Benutzungshandbuch

Turtle-Graphics-Simulator

Version 1.0 (30.09.2009)

Dietrich Boles

Universität Oldenburg

Inhaltsverzeichnis

1 Einleitung

Miniprogrammierwelten sind spezielle Entwicklungsumgebungen für Programmieranfänger. Sie bestehen aus einer überschaubaren Menge von Befehlen, die ein Programmieranfänger nutzen kann, um hiermit bestimmte ihm gestellte Aufgaben zu lösen. Die Ausführung der einzelnen Befehle bewirkt dabei jeweils eine bestimmte visuelle Änderung auf dem Bildschirm. Dadurch können Programmiernfänger sehr gut nachvollziehen, was ihr Programm tut. Fehler lassen sich sehr schnell entdecken und beheben.

1.1 Turtle-Graphics

Eine der ältesten Miniprogrammierwelten ist Turtle-Graphics, auf Deutsch „Schildkrötengraphik“. Mit Hilfe weniger vordefinierter Befehle wird eine virtuelle Schildkröte über den Bildschirm bewegt, die bei Bedarf eine farbige Linie hinter sich herziehen kann. Mit Hilfe der Schildkröte können also hübsche Zeichnungen erstellt werden.

Programmieranfängern werden nun bestimmte Aufgaben gestellt, die sie mit Hilfe der vordefinierten Befehle zu lösen haben. Die entstehenden Programme werden im Folgenden Turtle-Programme genannt.

In ihrer Ursprungsversion wurde die funktionale Programmiersprache Logo für die Turtle-Graphics genutzt. Logo hat aber heutzutage quasi keine Bedeutung mehr. Daher wurde beim vorliegenden Turtle-Simulator die Programmiersprache Java als Grundlage gewählt. Java – auch als „Sprache des Internet“ bezeichnet – ist eine moderne Programmiersprache, die sich in den letzten Jahren sowohl im Ausbildungsbereich als auch im industriellen Umfeld durchgesetzt hat.

Andere bekannte Miniprogrammierwelten sind übrigens das Hamster-Modell (www.java-hamster-modell.de) oder Kara, der Marienkäfer.

1.2 Der Turtle-Simulator

Bei den Miniprogrammierwelten steht nicht so sehr das "Learning-by-Listening" bzw. „Learning-by-Reading“ im Vordergrund, sondern vielmehr das „Learning-by-Doing“, also das praktische Üben. Genau das ist bezüglich der Turtle-Graphics mit dem Turtle-Graphics-Simulator, kurz Turtle-Simulator möglich. Dieser stellt eine Reihe von Werkzeugen zum Erstellen und Ausführen von Turtle-Programmen zur Verfügung: einen Editor zum Eingeben und Verwalten von Turtle-Programmen, einen Compiler zum Übersetzen von Turtle-Programmen, einen Territoriumsgestalter zum Gestalten und Verwalten von Turtle-Welten, einen Interpreter zum Ausführen von Turtle-Programmen und einen Debugger zum Testen von Turtle-Programmen. Der Turtle-

Simulator ist einfach zu bedienen, wurde aber funktional und bedienungsmäßig bewusst an professionelle Entwicklungsumgebungen für Java (z.B. Eclipse) angelehnt, um Programmieranfängern einen späteren Umstieg auf diese zu erleichtern.

1.3 Voraussetzungen

Zielgruppe der Turtle-Graphics sind Schüler oder Studierende ohne Programmiererfahrung, die die Grundlagen der (imperativen) Programmierung erlernen und dabei ein wenig Spaß haben wollen. Kenntnisse im Umgang mit Computern sind wünschenswert. Der Turtle-Simulator ist dabei kein Lehrbuch sondern ein Werkzeug, das das Erlernen der Programmierung unterstützt. Auf gute begeleitende Lehrbücher zum Erlernen der Programmierung wird in Kapitel 7 (Literatur zum Programmieren lernen) hingewiesen.

1.4 Anmerkungen

Der Turtle-Simulator wurde mit dem Tool „Solist“ erstellt. Solist ist eine spezielle Entwicklungsumgebung für Miniprogrammierwelt-Simulatoren. Solist ist ein Werkzeug der Werkzeugfamilie des „Programmier-Theaters“, eine Menge von Werkzeugen zum Erlernen der Programmierung. Metapher aller dieser Werkzeuge ist die Theaterwelt. Schauspieler (im Falle von Solist „Solisten“) agieren auf einer Bühne, auf der zusätzlich Requisiten platziert werden können. Eine Aufführung entspricht der Ausführung eines Programms.

Im Falle des Turtle-Simulators ist die Bühne die Turtle-Welt, in der die Schildkröte als Solist (es gibt nur eine Schildkröte) agiert. Wenn Ihnen also beim Umgang mit dem Turtle-Simulator der Begriff „Solist“ begegnet, kennen Sie nun hiermit den Hintergrund dieser Namenswahl.

Mehr Informationen zu Solist finden Sie im WWW unter www.programmierkurs-java.de/solist.

1.5 Aufbau des Benutzerhandbuch

Das Benutzungshandbuch ist in 7 Kapitel gegliedert. Nach dieser Einleitung wird in Kapitel 2 die Installation des Turtle-Simulators beschrieben. Kapitel 3 erläutert die Grundlagen der Turtle-Graphics. Wie Sie Ihr erstes Turtle-Programm zum Laufen bringen, erfahren Sie kurz und knapp in Kapitel 4. Dieses enthält eine knappe Einführung in die Elemente und die Bedienung des Turtle-Simulators. Sehr viel ausführlicher geht dann Kapitel 5 auf die einzelnen Elemente und die Bedienung des Turtle-Simulators ein. Kapitel 6 demonstriert an einigen Beispielprogrammen die Programmierung mit der Java-Schildkröte und gibt Ihnen einige Aufgaben an die

Hand, die Sie bzw. die Schildkröte zu lösen haben. Kapitel 7 enthält letztendlich Hinweise zu guter Begleitliteratur zum Erlernen der Programmierung mit Java.

2 Installation

2.1 Voraussetzungen

Voraussetzung zum Starten des Turtle-Simulators ist ein Java Development Kit SE (JDK) der Version 6 oder höher. Ein Java Runtime Environment SE (JRE) reicht nicht aus. Das jeweils aktuelle JDK kann über die Website http://java.sun.com/javase/downloads/index.jsp bezogen werden und muss anschließend installiert werden.

2.2 Download, Installation und Start

Der Turtle-Simulator kann von der Solist-Website http://www.programmierkurs-java.de/solist kostenlos herunter geladen werden. Er befindet sich in einer Datei namens *turtlesimulator-1.0.zip*. Diese muss zunächst entpackt werden. Es entsteht ein Ordner namens *turtlesimulator-1.0* (der so genannte *Simulator-Ordner*), in dem sich eine Datei *simulator.jar* befindet. Durch Doppelklick auf diese Datei wird der Turtle-Simulator gestartet. Alternativ lässt er sich auch durch Eingabe des Befehls `java –jar simulator.jar` in einer Console starten.

Beim ersten Start sucht der Turtle-Simulator nach der JDK-Installation auf Ihrem Computer. Sollte diese nicht gefunden werden, werden Sie aufgefordert, den entsprechenden Pfad einzugeben. Der Pfad wird anschließend in einer Datei namens *solist.properties* im Simulator-Ordner gespeichert, wo er später wieder geändert werden kann, wenn Sie bspw. eine aktuellere JDK-Version auf Ihrem Rechner installieren sollten. In der Property-Datei können Sie weiterhin die Sprache angeben, mit der der Turtle-Simulator arbeitet. In der Version 1.0 wird allerdings nur deutsch unterstützt.

3 Turtle-Graphics-Grundlagen

Computer können heutzutage zum Lösen vielfältiger Aufgaben genutzt werden. Die Arbeitsanleitungen zum Bearbeiten der Aufgaben werden ihnen in Form von Programmen mitgeteilt. Diese Programme, die von Programmierern entwickelt werden, bestehen aus einer Menge von Befehlen bzw. Anweisungen, die der Computer ausführen kann. Die Entwicklung solcher Programme bezeichnet man als Programmierung.

Turtle-Graphics ist ein spezielles didaktisches Modell zum Erlernen der Programmierung. Bei Turtle-Graphics nimmt eine virtuelle Schildkröte die Rolle des Computers ein. Dieser Schildkröte können ähnlich wie einem Computer Befehle erteilt werden, die diese ausführt.

Ihnen als Programmierer werden bestimmte Aufgaben gestellt, die sie durch die Steuerung der Schildkröte zu lösen haben. Derartige Aufgaben werden im Folgenden Turtle-Aufgaben genannt. Zu diesen Aufgaben müssen Sie in der Turtle-Sprache - eine Programmiersprache, die fast vollständig der Programmiersprache Java entspricht - Programme - Turtle-Programme genannt - entwickeln, die die Aufgaben korrekt und vollständig lösen.

Die Grundidee von Turtle-Graphics ist ausgesprochen einfach: Sie als Programmierer müssen eine (virtuelle) Schildkröte durch eine (virtuelle) Landschaft steuern und sie gegebene Aufgaben lösen lassen. Meist handelt es sich um das Anfertigen bestimmter Zeichnungen. Turtle-Graphics ist prädestiniert für zweidimensionale geometrische Zeichnungen und fraktale Kurven, wie die Drachenkurve oder die Hilbertkurve.

3.1 Schildkröte

Bei der vorliegenden Umsetzung der Turtle-Graphics existiert immer genau eine Schildkröte. Sie ist in der Mitte der Turtle-Welt platziert und schaut anfangs nach oben. Dies entspricht einem Winkel von 0 Grad. Abbildung 3.1 skizziert die Ausgangssituation der Schildkröte.

Abb. 3.1: Schildkröte

3.2 Befehle

Die Schildkröte kennt eine Reihe von Befehlen, durch deren Aufruf ein Programmierer sie steuern kann. Sie können sich die Schildkröte quasi als einen virtuellen Prozessor vorstellen, der im Gegensatz zu realen Computer-Prozessoren (zunächst) keine arithmetischen und logischen Operationen ausführen kann, sondern in der Lage ist, mit einem kleinen Grundvorrat an Befehlen Zeichnungen auf dem Bildschirm zu erstellen.

Die Befehle der Schildkröte sind:

- `void home();`: Die Schildkröte bewegt sich zur Mitte des Bildschirms mit Ausrichtung nach oben (0 Grad).
- `void clean();`:"Der Bildschirm wird gelöscht, die Position der Schildkröte ändert sich nicht.
- `void cleanscreen();`: Die Ausführung der Befehle `home` und `clean` zusammen.
- `void hideturtle();`: Die Schildkröte wird unsichtbar.
- `void showturtle();`: Die Schildkröte wird sichtbar.
- `void setpencolor(String color);`: Der Schildkröte wird eine Zeichenfarbe zugewiesen. Erlaubte Farben sind „red“, „black“, „blue“, „yellow“, „green“, „magenta“ und „white“.
- `void right(double degrees);`: Die Schildkröte dreht sich um einen bestimmten Winkel nach rechts.
- `void left(double degrees);`: Die Schildkröte dreht sich um einen bestimmten Winkel nach links.
- `void forward(double distance);`: Die Schildkröte bewegt sich um eine bestimmte Anzahl von Einheiten nach vorne.
- `void back(double distance);`: Die Schildkröte bewegt sich um eine bestimmte Anzahl von Einheiten nach hinten.

3.3 Turtle-Programme

Turtle-Programme werden in der so genannten Turtle-Sprache geschrieben. Diese ist fast deckungsgleich mit der Programmiersprache Java. Die wichtigsten Unterschiede sind:

- Anders als in Java bedarf es in der Turtle-Sprache keiner Klassen-Definition.
- In der Turtle-Sprache gibt es keine Methode mit der Signatur `public static void main(String[] args)`. Stattdessen gibt es die so genannte main-Prozedur mit der Signatur `void main()`.

- Turtle-Programme sind in der Regel imperative, keine objektorientierten Programme. Es brauchen also keine Objekte erzeugt zu werden. Trotzdem müssen Prozeduren und Funktionen nicht als `static` deklariert werden.
- Ein Turtle-Programm setzt sich aus der main-Prozedur sowie weiteren Prozeduren bzw. Funktionen zusammen, die vor oder nach der main-Prozedur definiert werden dürfen. Variablen können global, d.h. außerhalb von Prozeduren, oder lokal, d.h. innerhalb von Prozeduren, definiert werden. Der Gültigkeitsbereich von globalen Variablen erstreckt sich über das gesamte Programm, der Gültigkeitsbereich von lokalen Variablen ist auf den Prozedurrumpf beschränkt.
- Der Aufruf bzw. Start eines Turtle-Programms bewirkt die automatische Ausführung der main-Prozedur.

3.4 Beispielprogramm

In folgendem Programm zeichnet die Schildkröte die in Abbildung 3.2 skizzierte Zeichnung (das „Haus des Nikolaus“) auf den Bildschirm.

```
/*
Haus des Nikolaus
*/

void main() {
    int laenge = 100;
    right(90);
    forward(laenge);
    left(90);
    forward(laenge);
    left(30);
    forward(laenge);
    left(120);
    forward(laenge);
    left(30);
    forward(laenge);
    left(135);
    forward(Math.sqrt(2) * laenge);
    left(135);
    forward(laenge);
    left(135);
    forward(Math.sqrt(2) * laenge);
}
```

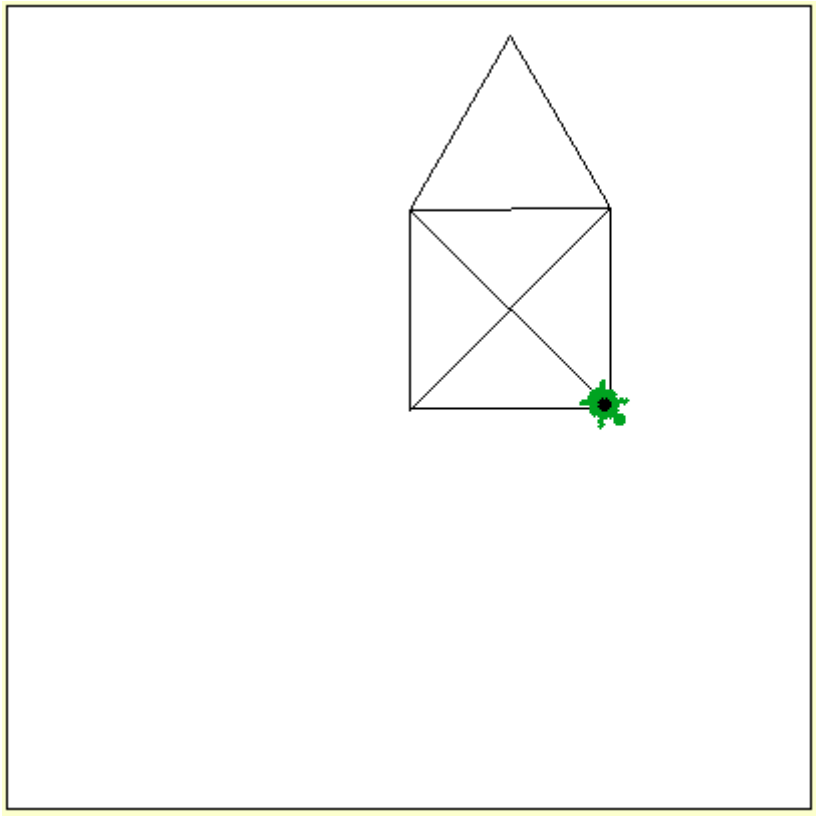


Abb. 3.2: Durch das Beispielprogramm erstellte Zeichnung

4 Ihr erstes Turtle-Programm

Nachdem Sie den Turtle-Simulator gestartet haben, öffnet sich ein Fenster, das in etwa dem in Abbildung 4.1 dargestellten Fenster entspricht. Ganz oben enthält es eine Menüleiste mit 5 Menüs, darunter eine Toolbar mit einer Reihe von Buttons und ganz unten einen Meldungsbereich, in dem Meldungen ausgegeben werden. Im linken Teil befindet sich der Editor-Bereich, im rechten Teil der Simulationsbereich. Im Editor-Bereich geben Sie Turtle-Programme ein und im Simulationsbereich führen Sie Turtle-Programme aus.

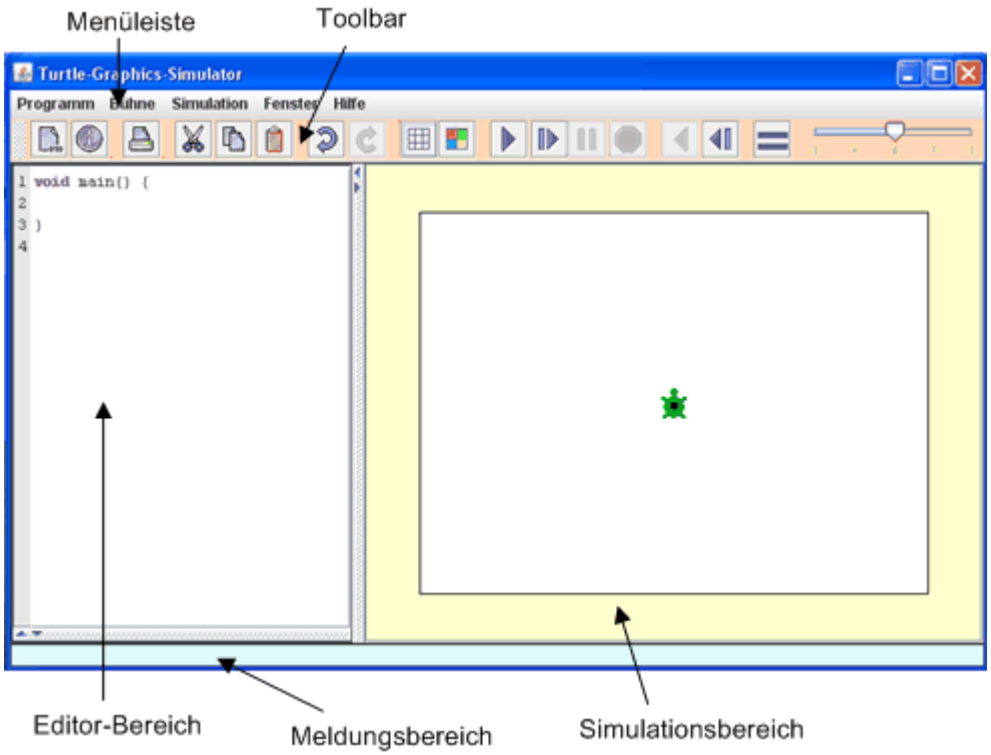


Abb. 4.1: Turtle-Simulator nach dem Öffnen

Um die Schildkröte ein wenig näher kennen zu lernen, epmpfehle ich Ihnen, als erstes zunächst die Turtle-Befehle einmal auszuprobieren. Wie das geht, wird in Abschnitt 4.1 erläutert. Anschließend wird in den darauf folgenden Abschnitten im Detail beschrieben, was Sie machen müssen, um Ihr erstes Turtle-Programm zu schreiben und auszuführen. Insgesamt müssen/können fünf Stationen durchlaufen werden:

- Gestaltung der Turtle-Welt
- Eingeben eines Turtle-Programms
- Compilieren eines Turtle-Programms

- Ausführen eines Turtle-Programms
- Debuggen eines Turtle-Programms

4.1 Ausprobieren der Turtle-Befehle

Klicken Sie im Menü „Fenster“ das Menü-Item „Befehlsfenster“ an. Es öffnet sich ein Fenster mit dem Titel „Befehlsfenster“. In diesem Fenster erscheinen alle Befehle, die die Schildkröte kennt.

Abb. 4.2: Befehlsfenster

Sie können die jeweiligen Befehle ausführen, indem Sie den Maus-Cursor auf den entsprechenden Button verschieben und diesen anklicken. Klicken Sie bspw. auf den Button „void hideturtle()“, dann wird die Schildkröte unsichtbar.

Besitzt ein Befehl Parameter, dann erscheint nach dem Anklicken des Befehls eine Dialogbox, in der die gewünschten Parameterwerte eingegeben werden können.

Wenn Sie Programme mit Prozeduren oder Funktionen schreiben und erfolgreich kompilieren, werden die Prozeduren und Funktionen übrigens auch im Befehlsfenster angezeigt und können interaktiv ausgeführt werden. Dabei werden keine Zwischenzustände im Simulationsbeereich ausgegeben, sondern immer der jeweilige Endzustand nach Abarbeitung der Funktion.

4.2 Gestaltung der Turtle-Welt

Zur Gestaltung der Turtle-Welt dienen die Buttons 9 und 10 (von links) in der so genannten *Toolbar*. Diese werden auch als *Gestaltungsbuttons* bezeichnet. Fahren Sie einfach mal mit der Maus über die einzelnen Buttons der Toolbar, dann erscheint jeweils ein Tooltipp, der beschreibt, wozu dieser Button dient.

Mit Hilfe des „Größe ändern“-Buttons (9. Toolbar-Button von links) können wir die Größe der Zeichenfläche ändern. Nach dem Anklicken des Buttons erscheint eine Dialogbox, in der Sie die gewünschte Anzahl an Reihen und Spalten (in Pixeln) eingeben können. Um die dort erscheinenden Werte ändern zu können, klicken Sie mit der Maus auf das entsprechende Eingabefeld. Anschließend können Sie den Wert mit der Tastatur eingeben. Nach der Eingabe der Werte klicken Sie bitte auf den OK-Button. Die Dialogbox schliesst sich und die sichtbare Zeichenfläche erscheint in der angegebenen Größe.

Das Ändern der Hintergrundfarbe der Zeichenfläche ist mit dem „Farbe ändern“-Button (10. Toolbar-Button von links) möglich. Nach dem Anklicken des Buttons erscheint eine spezielle Farbauswahl-Dialogbox, in der Sie die gewünschte Hintergrundfarbe auswählen können.

Um die Schildkröte auf der Zeichenfläche umzuplatzieren, klicken wir sie an und ziehen (man spricht auch von „draggen“) sie mit gedrückter Maustaste an die gewünschte Position. Dann lassen wir die Maustaste los. Eine Änderung der Blickrichtung der Schildkröte ist durch das Ausführen der Befehle left und right über das Befehlsfenster möglich.

Über das Menü „Bühne“ können Zeichenflächen auch in Dateien gespeichert und u.U. später wieder geladen werden.

4.3 Eingeben eines Turtle-Programms

Das Eingeben von Turtle-Programmen erfolgt im Editor-Bereich.

Unser erstes Programm soll bewirken, dass die Schildkröte wie in Abbildung 3.2 skizziert das „Haus des Nikolaus“ auf den Bildschirm malt. Wir klicken in den Editor-Bereich und tippen dort wie in einem normalen Editor bzw. Textverarbeitungsprogramm, wie Microsoft Word, die entsprechenden Turtle-Befehle ein, so dass letztlich folgendes im Eingabebereich steht:

```
void main() {
    int laenge = 100;
    right(90);
    forward(laenge);
    left(90);
```

```
    forward(laenge);
    left(30);
    forward(laenge);
    left(120);
    forward(laenge);
    left(30);
    forward(laenge);
    left(135);
    forward(Math.sqrt(2)*laenge);
    left(135);
    forward(laenge);
    left(135);
    forward(Math.sqrt(2)*laenge);
}
```

Das ist unser erstes Turtle-Programm.

Ihnen sicher von anderen Editoren bzw. Textverarbeitungsprogrammen bekannte Funktionen, wie „Ausschneiden“, „Kopieren“, „Einfügen“, „Rückgängig“ und „Wiederherstellen“ können Sie über das Menü „Programm“ bzw. die entsprechenden Buttons in der Toolbar ausführen (vierter bis achter Button von links).

Weiterhin gibt es im Menü „Programm“ zwei Menü-Items zum Speichern von Programmen in Dateien und zum wieder Laden von abgespeicherten Programmen. Bei ihrer Aktivierung erscheint eine Dateiauswahl-Dialogbox, in der Sie die entsprechende Auswahl der jeweiligen Datei vornehmen müssen. Achtung: Wenn Sie eine abgespeicherte Datei laden, geht der Programmtext, der sich aktuell im Editorbereich befindet, verloren (wenn er nicht zuvor in einer Datei abgespeichert wurde).

Im Menü „Programm“ finden Sie darüber hinaus weitere nützliche Funktionen (Schriftgröße ändern, Drucken, …).

4.4 Compilieren eines Turtle-Programms

Nachdem wir unser Turtle-Programm geschrieben haben, müssen wir es compilieren. Der Compiler überprüft den Sourcecode auf syntaktische Korrektheit und transformiert ihn – wenn er korrekt ist – in ein ausführbares Programm. Zum Compilieren drücken Sie einfach auf den „Compilieren“-Button (erster Toolbar-Button von links oder „Compilieren“-Menü-Item im „Programm“-Menü). Compiliert wird dann das Programm, das gerade im Eingabebereich sichtbar ist. Es wird zuvor automatisch in einer (temporären) Datei (namens Solist.java) abgespeichert.

Wenn das Programm korrekt ist, erscheint eine Dialogbox mit der Nachricht „Compilierung erfolgreich“. Zur Bestätigung müssen Sie anschließend noch den OK-

Button drücken. Das Programm kann nun ausgeführt werden. Merken Sie sich bitte: Immer, wenn Sie Änderungen am Sourcecode Ihres Programms vorgenommen haben, müssen Sie es zunächst neu kompilieren.

Wenn das Programm syntaktische Fehler enthält – wenn Sie sich bspw. bei der Eingabe des obigen Programms vertippt haben –, werden unter dem Editor-Bereich die Fehlermeldungen des Compilers eingeblendet. Diese erscheinen in englischer Sprache. Weiterhin wird die Zeile angegeben, in der der Fehler entdeckt wurde. Wenn Sie mit der Maus auf die Fehlermeldung klicken, springt der Maus-Cursor im Editor-Bereich automatisch in die angegebene Zeile (siehe Abbildung 4.3).

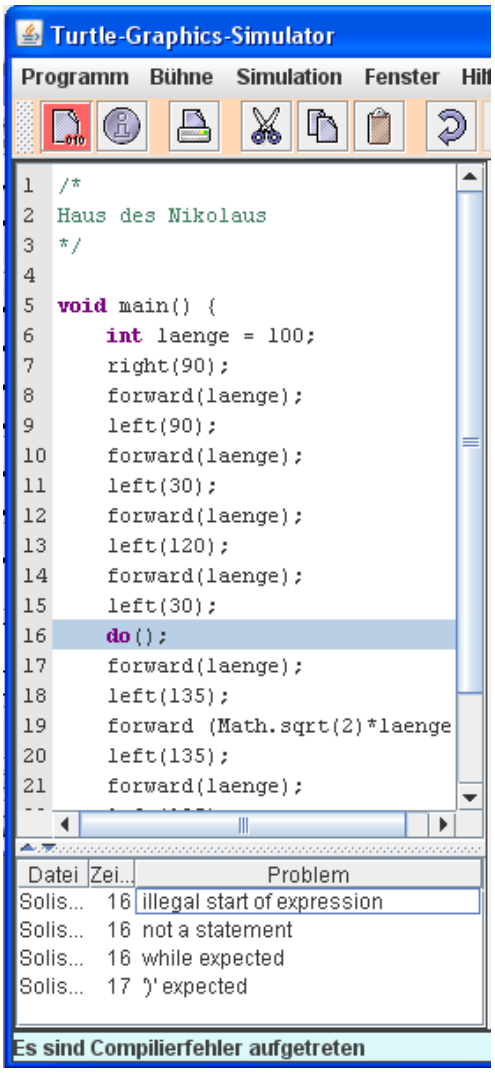


Abb. 4.3: Fehlermeldungen des Compilers

Vorsicht: Die Fehlermeldungen sowie die Zeilenangabe eines Compilers sind nicht immer wirklich exakt. Das Interpretieren der Meldungen ist für Programmieranfänger

häufig nicht einfach und bedarf einiger Erfahrungen. Deshalb machen Sie ruhig am Anfang mal absichtlich Fehler und versuchen Sie, die Meldungen des Compilers zu verstehen.

Tipp: Arbeiten Sie die Fehler, die der Compiler entdeckt hat, immer von oben nach unten ab. Wenn Sie eine Meldung dann überhaupt nicht verstehen, compilieren Sie ruhig erst mal erneut. Häufig ist es (leider) so, dass der Compiler für einen einzelnen Fehler mehrere Fehlermeldungen ausgibt, was Anfänger leicht verwirren kann.

Nachdem Sie die Fehler korrigiert haben, müssen Sie das Programm erneut compilieren. Wiederholen Sie dies so lange, bis der Compiler die Meldung „Compilierung erfolgreich“ ausgibt. Erst dann können Sie das Programm ausführen!

4.5 Ausführen eines Turtle-Programms

Nach dem erfolgreichen Compilieren ist es endlich soweit: Wir können die Schildkröte bei der Arbeit beobachten. Macht sie wirklich das, was wir ihr durch unser Programm beigebracht haben?

Zum Steuern der Programmausführung dienen die Buttons rechts in der Toolbar. Durch Anklicken des „Simulation starten“-Buttons (11. Button von links) starten wir das Programm. Wenn Sie bis hierhin alles richtig gemacht haben, sollte die Schildkröte loslaufen und wie im Programm beschrieben, das Haus des Nikolaus zeichnen. Herzlichen Glückwunsch zu Ihrem ersten Turtle-Programm!

Wollen Sie die Programmausführung anhalten, können Sie dies durch Anklicken des „Simulation pausieren“-Buttons (13. Button von links) erreichen. Die Schildkröte pausiert so lange, bis Sie wieder den „Simulation starten/fortsetzen“-Button (11. Button von links) anklicken. Dann fährt die Schildkröte mit ihrer Arbeit fort. Das Programm vorzeitig komplett abbrechen, können Sie mit Hilfe des „Simulation beenden“-Buttons (14. Button von links).

Wenn Sie ein Programm mehrmals hintereinander im gleichen Territorium ausführen wollen, können Sie mit dem „Rücksetzen“-Button (15. Button von links) den Zustand des Territoriums wieder herstellen, der vor dem letzten Ausführen des Programms Bestand hatte. Eine komplette Zurücksetzung des Territoriums auf den Zustand beim Öffnen des Turtle-Simulators ist mit dem „Komplett zurücksetzen“-Button möglich (15. Button von links).

Der Schieberegler ganz rechts in der Menüleiste dient zur Steuerung der Geschwindigkeit der Programmausführung. Je weiter Sie den Knopf nach links verschieben, umso langsamer erledigt die Schildkröte ihre Arbeit. Je weiter Sie ihn nach rechts verschieben, umso schneller flitzt die Schildkröte durchs Territorium.

Die Bedienelemente zum Steuern der Programmausführung finden Sie übrigens auch im Menü „Simulation“.

4.6 Debuggen eines Turtle-Programms

„Debuggen eines Programms“ eines Programms bedeutet, dass Sie bei der Ausführung eines Programms zusätzliche Möglichkeiten zur Steuerung besitzen und sich den Zustand des Programms (welche Zeile des Sourcecodes wird gerade ausgeführt, welche Werte besitzen aktuell die Variablen) in bestimmten Situationen anzeigen lassen können. Das ist ganz nützlich, wenn Ihr Programm nicht das tut, was es soll, und sie herausfinden wollen, woran der Fehler liegt.

Klicken Sie zum Debuggen zunächst den „Ablaufverfolgung aktivieren“-Button in der Toolbar an (17. Button von links). Es wird das so genannte Debugger-Fenster mit dem Titel „Debugger“ geöffnet. Im linken Bereich des Debugger-Fensters wird der Prozedur-Stack angezeigt, das ist die aktuelle Prozedur sowie die Prozeduren, aus denen die Prozedur heraus aufgerufen wurde. Im rechten Bereich werden – falls vorhanden – die gültigen Variablen und ihre aktuellen Werte dargestellt (siehe Abb. 4.4).

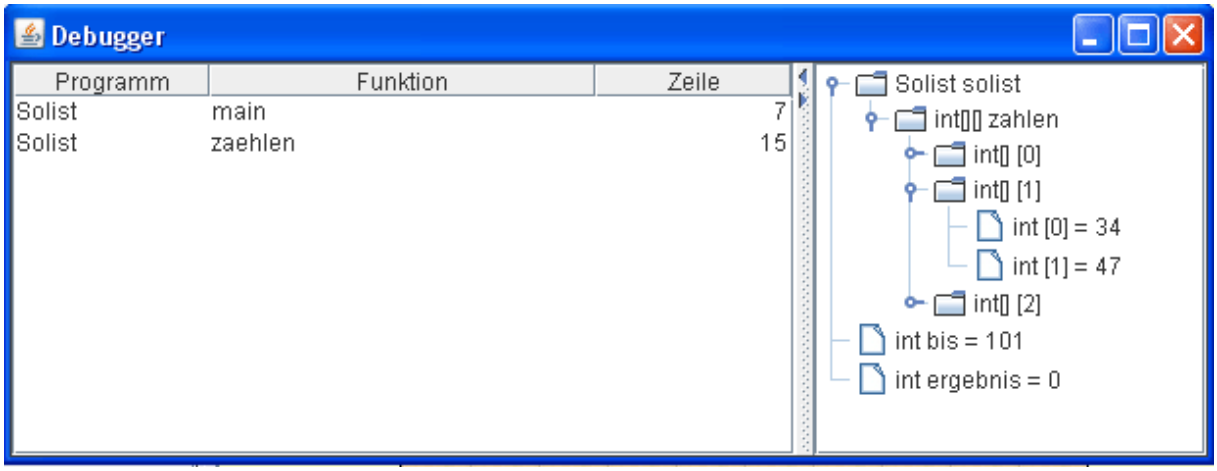


Abb. 4.4: Debugger-Fenster

Jetzt können Sie über den „Schritt“-Button (12. Toolbar-Button von links) jeden Befehl einzeln ausführen und bekommen im Editorbereich durch einen blauen Balken angezeigt, welcher Befehl bzw. welche Zeile als nächstes ausgeführt wird. Bei einem Prozeduraufruf wird dabei auch automatisch in die entsprechende Prozedur verzweigt. Im Debugger-Fenster wird zusätzlich der Prodzedur-Stack angezeigt, d.h. die aktuelle Prozedur sowie die Prozeduren, aus denen die Prozedur heraus aufgerufen wurde.

Sie können zunächst auch einfach das Programm durch Anklicken des „Starten“-Buttons starten und beobachten. Wenn Sie dann den „Pause“-Button drücken, haben Sie anschließend ebenfalls die Möglichkeit der schrittweisen Ausführung ab der aktuellen Position.

Die Ablaufverfolgung kann übrigens jederzeit durch erneutes Klicken des „Ablaufverfolgung“-Buttons deaktiviert bzw. reaktiviert werden.

Wenn Sie möchten, dass der Programmablauf beim Erreichen einer bestimmten Zeile automatisch in den Pausezustand gelangt, können Sie vor oder während des Programmablaufs in der entsprechenden Zeile einen Breakpoint setzen. Führen Sie dazu im Editorbereich auf der entsprechenden Zeilennummer einen Doppelklick mit der Maus aus. Breakpoints werden durch eine violette Hinterlegung der Zeilennummer kenntlich gemacht. Durch erneuten Doppelklick oder über ein Popup-Menü, das oberhalb der Zeilennummern aktiviert werden kann, kann ein Breakpoint oder auch alle Breakpoints wieder gelöscht werden. Breakpoints nutzt man häufig dazu, dass man ein Programm bis zu einer bestimmten Stelle normal ablaufen lässt und ab dort dann die Möglichkeit der zeilenweisen Ausführung nutzt.

4.7 Zusammenfassung

Herzlichen Glückwunsch! Wenn Sie bis hierhin gekommen sind, haben Sie Ihr erstes Turtle-Programm erstellt und ausgeführt. Sie sehen, die Bedienung des Turtle-Simulators ist gar nicht so kompliziert.

Der Turtle-Simulator bietet jedoch noch weitere Möglichkeiten. Diese können Sie nun durch einfaches Ausprobieren selbst erkunden oder im nächsten Abschnitt im Detail nachlesen.

5 Bedienung des Turtle-Simulators

Im letzten Abschnitt haben Sie eine kurze Einführung in die Funktionalität des Turtle-Simulators erhalten. In diesem Abschnitt werden die einzelnen Funktionen des Simulators nun im Detail vorgestellt. Dabei wird sich natürlich einiges auch wiederholen.

Wenn Sie den Turtle-Simulator starten, öffnen sich ein Fenster mit dem Titel „Turtle-Simulator“. Abbildung 5.1 skizziert die einzelnen Komponenten des Fensters.

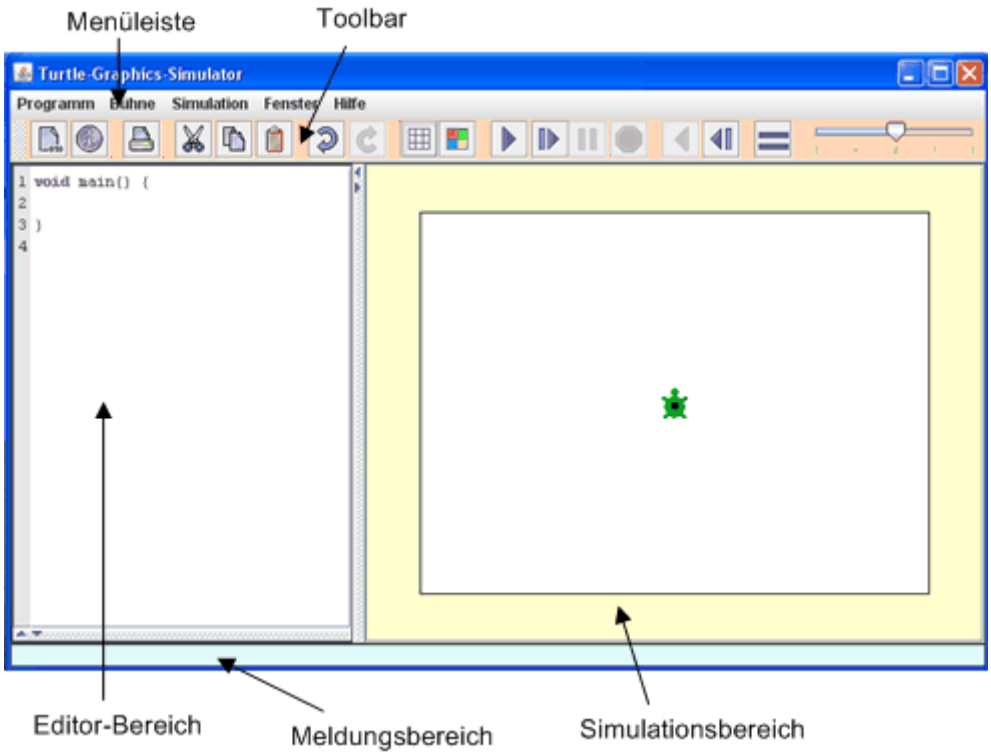


Abb. 5.1: Komponenten des Simulator-Fensters

Die Menüleiste oben im Fenster beinhaltet 5 Menüs. Darunter ist eine Toolbar mit Buttons platziert, über die die wichtigsten Funktionen der Menüs durch Anklicken eines Buttons schneller ausgeführt werden können. Ganz unten im Meldungsbereich werden wichtige Meldungen ausgegeben. Die Eingabe von Turtle-Programmen erfolgt im Editor-Bereich links und die Ausführung von Turtle-Programmen wird im Simulationsbereich (auch „Bühne“ genannt) visualisiert.

Als Hauptfunktionsbereiche des Turtle-Simulators lassen sich identifizieren:

- Verwalten und Editieren von Turtle-Programmen

- Compilieren von Turtle-Programmen
- Verwalten und Gestalten von Turtle-Welten
- Interaktives Ausführen von Turtle-Befehlen
- Ausführen von Turtle-Programmen
- Debuggen von Turtle-Programmen

Bevor im Folgenden anhand dieser Funktionsbereiche der Simulator im Detail vorgestellt wird, werden zuvor noch einige Grundfunktionen graphischer Benutzungsoberflächen erläutert.

5.1 Grundfunktionen

In diesem Unterabschnitt werden einige wichtige Grundfunktionalitäten graphischer Benutzungsoberflächen beschrieben. Der Abschnitt ist für diejenigen von Ihnen gedacht, die bisher kaum Erfahrungen mit Computern haben. Diejenigen von Ihnen, die schon längere Zeit einen Computer haben und ihn regelmäßig benutzen, können diesen Abschnitt ruhig überspringen.

5.1.1 Anklicken

Wenn im Folgenden von „Anklicken eines Objektes“ oder „Anklicken eines Objektes mit der Maus“ gesprochen wird, bedeutet das, dass Sie den Maus-Cursor auf dem Bildschirm durch Verschieben der Maus auf dem Tisch über das Objekt platzieren und dann die – im Allgemeinen linke – Maustaste drücken.

5.1.2 Tooltipps

Als *Tooltipps* werden kleine Rechtecke bezeichnet, die automatisch auf dem Bildschirm erscheinen, wenn man den Maus-Cursor auf entsprechende Objekte platziert (siehe Abbildung 5.2). In den Tooltipps werden bestimmte Informationen ausgegeben.

Abb. 5.2: Tooltipp

5.1.3 Button

Buttons sind Objekte der Benutzungsoberfläche, die man anklicken kann und die daraufhin eine bestimmte Aktion auslösen (siehe Abbildung 6.3). Buttons besitzen eine textuelle Beschreibung (z.B. „OK“) oder eine Graphik, die etwas über die Aktion

aussagen. Sie erkennen Buttons an der etwas hervorgehobenen Darstellung. Graphik-Buttons sind in der Regel Tooltipps zugeordnet, die die zugeordnete Aktion beschreiben.

Abb. 5.3: Buttons

5.1.4 Menü

Menüs befinden sich ganz oben in einem Fenster in der so genannten *Menüleiste* (siehe Abbildung 5.4). Sie werden durch einen Text beschrieben (Programm, Bühne, Simulation, ...). Klickt man die Texte an, öffnet sich eine Box mit so genannten *Menü-Items*. Diese bestehen wiederum aus Texten, die man anklicken kann. Durch Anklicken von Menü-Items werden genauso wie bei Buttons Aktionen ausgelöst, die im Allgemeinen durch die Texte beschrieben werden (Speichern, Kopieren, ...). Nach dem Anklicken eines Menü-Items wird die Aktion gestartet und die Box schließt sich automatisch wieder. Klickt man irgendwo außerhalb der Box ins Fenster, schließt sich die Box ebenfalls und es wird keine Aktion ausgelöst.

Abb. 5.4: Menüleiste und Menü

Häufig steht hinter den Menü-Items ein weiterer Text, wie z.B. „Strg-O“ oder „Alt-N“. Diese Texte kennzeichnen Tastenkombinationen. Drückt man die entsprechenden Tasten auf der Tastatur, wird dieselbe Aktion ausgelöst, die man auch durch Anklicken des Menü-Items auslösen würde.

Manchmal erscheinen bestimmte Menü-Items etwas heller. Man sagt auch, sie sind ausgegraut. In diesem Fall kann man das Menü-Item nicht anklicken und die zugeordnete Aktion nicht auslösen. Das Programm befindet sich in einem Zustand, in dem die Aktion keinen Sinn machen würde.

5.1.5 Toolbar

Direkt unterhalb der Menüleiste ist die so genannte *Toolbar* angeordnet (siehe Abbildung 5.5). Sie besteht aus einer Menge an Graphik-Buttons, die Alternativen zu den am häufigsten benutzten Menü-Items der Menüs darstellen.

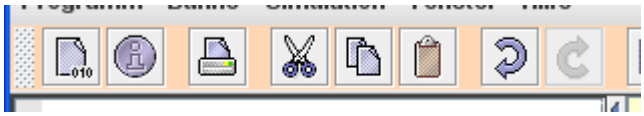


Abb. 5.5: Toolbar

5.1.6 Popup-Menü

Popup-Menüs sind spezielle Menüs, die bestimmten Elementen auf dem Bildschirm zugeordnet sind (siehe Abbildung 5.6). Man öffnet sie dadurch, dass man den Maus-Cursor auf das entsprechende Element verschiebt und danach die rechte Maustaste drückt. Genauso wie bei normalen Menüs erscheint dann eine Box mit Menü-Items.

Abb. 5.6: Popup-Menü

5.1.7 Eingabefeld

Eingabefelder dienen zur Eingabe von Zeichen (siehe Abbildung 5.7). Positionieren Sie dazu den Maus-Cursor auf das Eingabefeld und klicken Sie die Maus. Anschließend können Sie über die Tastatur Zeichen eingeben, die im Eingabefeld erscheinen.

5.1.8 Dialogbox

Beim Auslösen bestimmter Aktionen erscheinen so genannte *Dialogboxen* auf dem Bildschirm (siehe Abbildung 5.7). Sie enthalten in der Regel eine Menge von graphischen Objekten, wie textuelle Informationen, Eingabefelder und Buttons. Wenn eine Dialogbox auf dem Bildschirm erscheint, sind alle anderen Fenster des Programms für Texteingaben oder Mausklicks gesperrt. Zum Schließen einer Dialogbox, d.h. um die Dialoxbox wieder vom Bildschirm verschwinden zu lassen, dienen in der Regel eine Menge an Buttons, die unten in der Dialogbox angeordnet sind. Durch Anklicken eines „OK-Buttons“ wird dabei die der Dialogbox zugeordnete

Aktion ausgelöst. Durch Anklicken des „Abbrechen-Buttons“ wird eine Dialogbox geschlossen, ohne dass irgendwelche Aktionen ausgelöst werden.

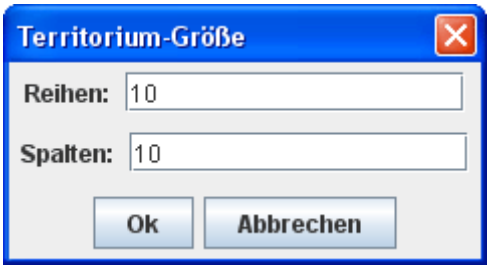


Abb. 5.7: Dialogbox mit Eingabefeldern

5.1.9 Dateiauswahl-Dialogbox

Dateiauswahl-Dialogboxen sind spezielle Dialogboxen, die zum Speichern und Öffnen von Dateien benutzt werden (siehe Abbildung 5.8). Sie spiegeln im Prinzip das Dateisystem wider und enthalten Funktionalitäten zum Verwalten von Dateien und Ordnern.

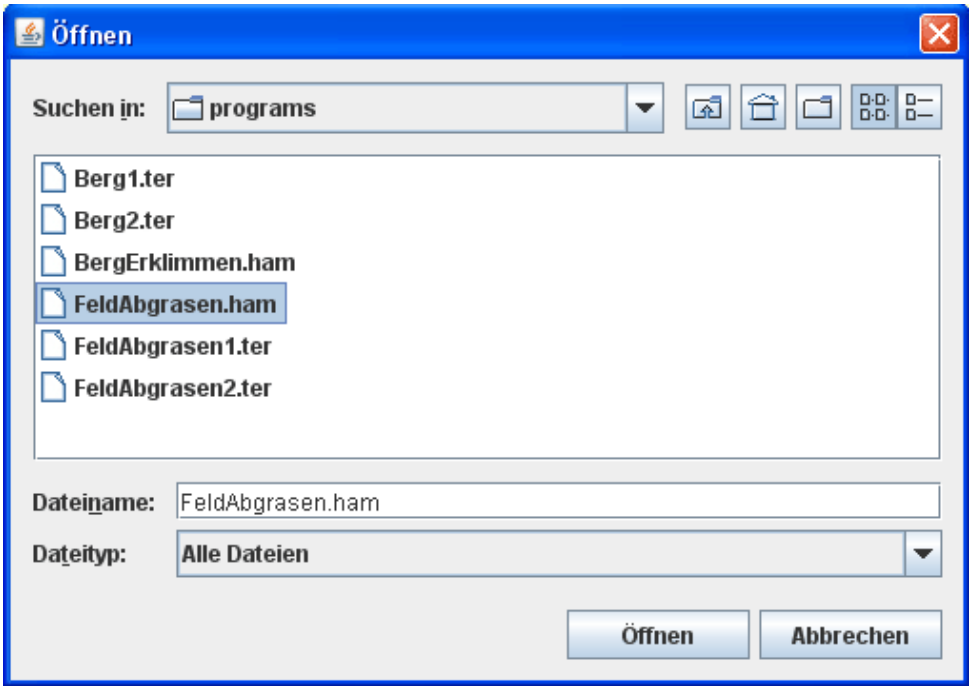


Abb. 5.8: Dateiauswahl-Dialogbox

Im mittleren Bereich einer Dateiauswahl-Dialogbox erscheinen alle Dateien und Unterordner des aktuellen Ordners. Sie sind durch unterschiedliche Symbole repräsentiert. Der eigentliche Zweck von Dateiauswahl-Dialogboxen ist – wie der

Name schon sagt – die Auswahl einer Datei. Klickt man auf eine Datei, erscheint der Name automatisch im Eingabefeld „Dateiname“. Dort kann man auch über die Tastatur einen Dateinamen eingeben. Anschließend wird nach Drücken des OK-Buttons die entsprechende Datei geöffnet bzw. gespeichert.

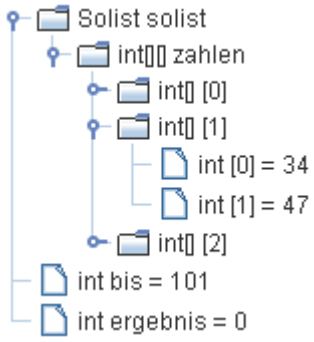
Dateiauswahl-Dialogboxen stellen jedoch noch zusätzliche Funktionalitäten bereit. Durch Doppelklick auf einen Ordner kann man in den entsprechenden Ordner wechseln. Es werden dann anschließend die Dateien und Unterordner dieses Ordners im mittleren Bereich angezeigt. Um zu einem übergeordneten Ordner zurück zu gelangen, bedient man sich des Menüs „Suchen in“, in dem man den entsprechenden Ordner auswählen kann.

Neben dem „Suchen in“-Menü sind noch fünf Graphik-Buttons angeordnet. Durch Anklicken des linken Buttons kommt man im Ordnerbaum eine Ebene höher. Durch Anklicken des zweiten Buttons von links gelangt man zur Wurzel des Ordnerbaumes. Mit dem mittleren Button kann man im aktuellen Ordner einen neuen Unterordner anlegen. Mit den beiden rechten Buttons kann man die Darstellung im mittleren Bereich verändern.

Möchte man einen Ordner oder eine Datei umbenennen, muss man im mittleren Bereich der Dateiauswahl-Dialogbox zweimal – mit Pause zwischendurch – auf den Namen des Ordners oder der Datei klicken. Die textuelle Darstellung des Namens wird dann zu einem Eingabefeld, in der man über die Tastatur den Namen verändern kann.

5.1.10 Elementbaum

Ein *Elementbaum* repräsentiert Elemente und strukturelle Beziehungen zwischen ihnen, bspw. die Ordner und Dateien des Dateisystems (siehe Abbildung 5.9).



Abbi. 5.9: Elementbaum mit Verzeichnissen und Dateien

Unterschiedliche Elementtypen werden dabei durch unterschiedliche Symbole dargestellt, hinter denen entsprechende Bezeichnungen erscheinen. Durch Anklicken

der Symbole auf der linken Seite kann man Strukturen öffnen und schließen, d.h. Unterstrukuren sichtbar bzw. unsichtbar machen.

Den Ordnern und Dateien sind Popup-Menüs zugeordnet. Um diese zu öffnen, muss man zunächst den Ordner bzw. die Datei mit der Maus anklicken. Der Name wird dann durch einen blauen Balken hinterlegt. Anschließend muss man die rechte Maustaste drücken. Dann öffnet sich das Popup-Menü. Die Popup-Menüs enthalten bspw. Menü-Items zum Löschen und Umbenennen des entsprechenden Ordners bzw. der entsprechenden Datei.

5.1.11 Split-Pane

Eine Split-Pane ist ein Element, das aus zwei Bereichen und einem Balken besteht. (siehe Abbildung 5.10). Die beiden Bereiche können dabei links und rechts oder oberhalb und unterhalb des Balkens liegen. Wenn Sie den Balken mit der Maus anklicken und bei gedrückter Maustaste nach links oder rechts (bzw. oben oder unten) verschieben, vergrößert sich einer der beiden Bereiche und der andere verkleinert sich. Durch Klicken auf einen der beiden Pfeile auf dem Balken können Sie einen der beiden Bereiche auch ganz verschwinden lassen.

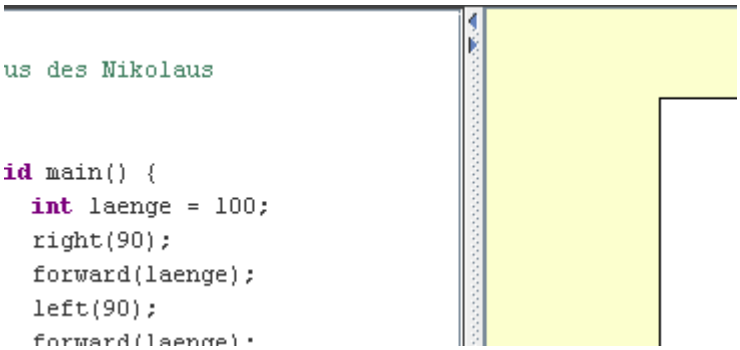


Abb. 5.10: Split-Pane

5.2 Verwalten und Editieren von Turtle-Programmen

Das Schreiben von Programmen bzw. genauer gesagt das Schreiben des Sourcecodes von Programmen bezeichnet man als *Editieren*. Im Turtle-Simulator dient der Editor-Bereich zum Editieren von Turtle-Programmen (siehe Abbildung 5.11)

```
1  /*
2  Haus des Nikolaus
3  */
4
5  void main() {
6      int laenge = 100;
7      right(90);
8      forward(laenge);
9      left(90);
10     forward(laenge);
11     left(30);
12     forward(laenge);
13     left(120);
14     forward(laenge);
15     left(30);
16     forward(laenge);
17     left(135);
18     forward (Math.sqrt(2)*laenge);
19     left(135);
20     forward(laenge);
21     left(135);
22     forward (Math.sqrt(2)*laenge);
23 }
24
```

Abb. 5.11: Editor-Bereich des Turtle-Simulators

Im Editor-Bereich können Sie Programme eintippen. Für das Verwalten und Editieren von Programmen ist das Menü „Programm“ wichtig. Unterhalb der Menüleiste ist eine spezielle Toolbar zu sehen, über die Sie die wichtigsten Funktionen der Menüs auch schneller erreichen und ausführen können. Schieben Sie einfach mal die Maus über die Buttons. Dann erscheint jeweils ein Tooltipp, der die Funktionalität des Buttons anzeigt. Die für das Editieren von Programmen wichtigen Buttons der Toolbar werden in Abbildung 5.12 skizziert.

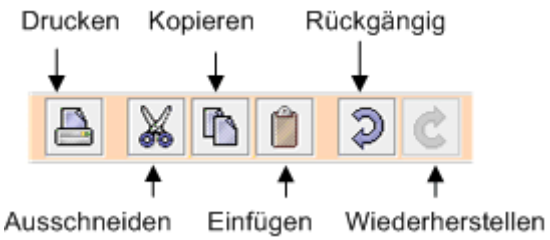


Abb. 5.12: Editor-Buttons der Toolbar

5.2.1 Schreiben eines neuen Turtle-Programms

Das Schreiben eines neuen Turtle-Programms ist durch das entsprechende Eintippen des Sourcecodes im Editor-Bereich möglich.

5.2.2 Ändern des aktuellen Turtle-Programms

Möchten Sie Teile des aktuellen Turtle-Programms ändern, klicken Sie im Editor-Bereich einfach an die entsprechende Stelle und fügen dort die entsprechenden Wörter ein oder löschen sie.

5.2.3 Löschen des aktuellen Turtle-Programm

Komplett löschen können Sie das aktuelle Turtle-Programm des Editor-Bereichs, indem Sie den kompletten Sourcecode mit der Maus selektieren und dann in der Toolbar den „Ausschneiden“-Button (4. Button von links) drücken.

5.2.4 Abspeichern des aktuellen Turtle-Programms

Normalerweise müssen Sie sich nicht um das Speichern des aktuellen Programms kümmern. Es wird automatisch vor dem Compilieren in einer internen Datei abgespeichert. Wenn Sie jedoch ein Programm explizit abspeichern möchten, können Sie im „Programm“-Menü das „Speichern unter…“-Menü-Item anklicken. Es öffnet sich eine Dateiauswahl-Dialogbox, über die Sie die gewünschte Datei auswählen können.

5.2.5 Öffnen eines einmal abgespeicherten Turtle-Programms

Möchten Sie ein einmal abgespeichertes Turtle-Programm wieder in den Editorbereich laden, können Sie dies über das Menü-Item „Laden…“ des „Programm“-Menüs tun. Nach dem Anklicken des Items erscheint eine Dateiauswahl-Dialogbox, über die Sie die gewünschte Datei auswählen können.

Achtung: Beim Laden einer abgespeicherten Datei geht der aktuelle Inhalt des Editor-Bereichs verloren! Sie müssen ihn also gegebenenfalls vorher in einer Datei abspeichern.

5.2.6 Drucken eines Turtle-Programms

Über den „Drucken“-Button (3. Toolbar-Button von links) können Sie das aktuelle Programm des Editor-Bereichs drucken. Es öffnet sich eine Dialogbox, in der Sie die entsprechenden Druckeinstellungen vornehmen und den Druck starten können.

5.2.7 Editier-Funktionen

Im Editor-Bereich können Sie – wie bei anderen Editoren auch – über die Tastatur Zeichen eingeben bzw. wieder löschen. Darüber hinaus stellt der Editor ein paar weitere Funktionalitäten zur Verfügung, die über das „Programm“-Menü bzw. die entsprechenden Editor-Buttons in der Toolbar aktiviert werden können.

- „Ausschneiden“-Button (4. Toolbar-Button von links): Hiermit können Sie komplette Passagen des Editor-Bereichs in einem Schritt löschen. Markieren Sie die zu löschende Passage mit der Maus und klicken Sie dann den Button an. Der markierte Text verschwindet.
- „Kopieren“-Button (5. Toolbar-Button von links): Hiermit können Sie komplette Passagen des Editor-Bereichs in einen Zwischenpuffer kopieren. Markieren Sie die zu kopierende Passage mit der Maus und klicken Sie dann den Button an.
- „Einfügen“-Button (5. Toolbar-Button von links): Hiermit können Sie den Inhalt des Zwischenpuffers an die aktuelle Cursorposition einfügen. Wählen Sie zunächst die entsprechende Position aus und klicken Sie dann den Button an. Der Text des Zwischenpuffers wird eingefügt.
- „Rückgängig“-Button (7. Toolbar-Button von links): Wenn Sie durchgeführte Änderungen des Sourcecode – aus welchem Grund auch immer – wieder rückgängig machen wollen, können Sie dies durch Anklicken des Buttons bewirken.
- „Wiederherstellen“-Button (8. Toolbar-Button von links): Rückgängig gemachte Änderungen können Sie mit Hilfe dieses Buttons wieder herstellen.

Die Funktionalitäten „Kopieren“ und „Einfügen“ funktionieren übrigens auch über einzelne Programme hinaus. Es ist sogar möglich, mit Hilfe der Betriebssystem-Kopieren-Funktion Text aus anderen Programmen (bspw. Microsoft Word) zu kopieren und hier einzufügen.

Die gerade aufgelisteten Funktionen finden Sie auch im „Programm“-Menü. Als zusätzliche Funktionalitäten werden dort angeboten:

- Einrückung: Durch Anklicken des Menü-Items wird der Einrück-Modus aktiviert bzw. deaktiviert. Ist der Einrück-Modus aktiviert, werden beim Eingeben von Text im Editor-Bereich automatisch Einrückungen an die entsprechende Spalte vorgenommen, wenn Sie die Enter-Taste drücken.
- Schriftgröße: Hierüber können Sie die Schriftgröße des Editor-Bereichs anpassen.

5.3 Compilieren von Turtle-Programmen

Beim Compilieren werden Programme – genauer gesagt der Sourcecode – auf ihre (syntaktische) Korrektheit überprüft und im Erfolgsfall ausführbare Programme erzeugt. Zum Compilieren von Programmen dient im Turtle-Simulator der „Compilieren“-Button (erster Button der Toolbar von links) bzw. das Menü-Item „Compilieren“ im „Programm“-Menü (siehe Abbildung 5.13).

An der Farbe des Compilier-Buttons können Sie erkennen, ob Compilieren aktuell notwendig ist oder nicht. Immer wenn Sie Änderungen im Editor-Bereich vorgenommen haben, erscheint der Button rot, und die Änderungen werden erst dann berücksichtigt, wenn Sie (erneut) compiliert haben. Erscheint der Button in einer neutralen Farbe, ist kein Compilieren notwendig.

5.3.1 Compilieren

Wenn Sie den „Compilieren“-Button anklicken, wird das Programm, das gerade im Editor-Bereich sichtbar ist, in einer internen Datei (mit dem Namen „Solist.java“) abgespeichert und kompiliert.

Wenn Ihr Programm syntaktisch korrekt ist, erscheint nach ein paar Sekunden eine Dialogbox mit einer entsprechenden Meldung. Es wurde ein (neues) ausführbares Programm erzeugt.

5.3.2 Beseitigen von Fehlern

Wenn Ihr Programm Fehler enthält, öffnet sich unterhalb des Editor-Bereichs in einer Scroll-Pane ein neuer Bereich, der die Fehlermeldungen des Compilers anzeigt (siehe Abbildung 5.13). Es wurde kein (neues) ausführbares Programm erzeugt! Jede Fehlermeldung erscheint in einer eigenen Zeile. Jede Zeile enthält eine Beschreibung des (wahrscheinlichen) Fehlers sowie die entsprechende Zeile der Anweisung im Programm. Wenn Sie eine Fehlermeldung anklicken, wird die entsprechende Anweisung im Eingabebereich blau markiert und der Maus-Cursor an die entsprechende Stelle gesetzt. Sie müssen nun die einzelnen Fehler beseitigen und dann erneut speichern und compilieren, bis Ihr Programm keine Fehler mehr enthält. Der Fehlermeldungsbereich schließt sich dann automatisch wieder.

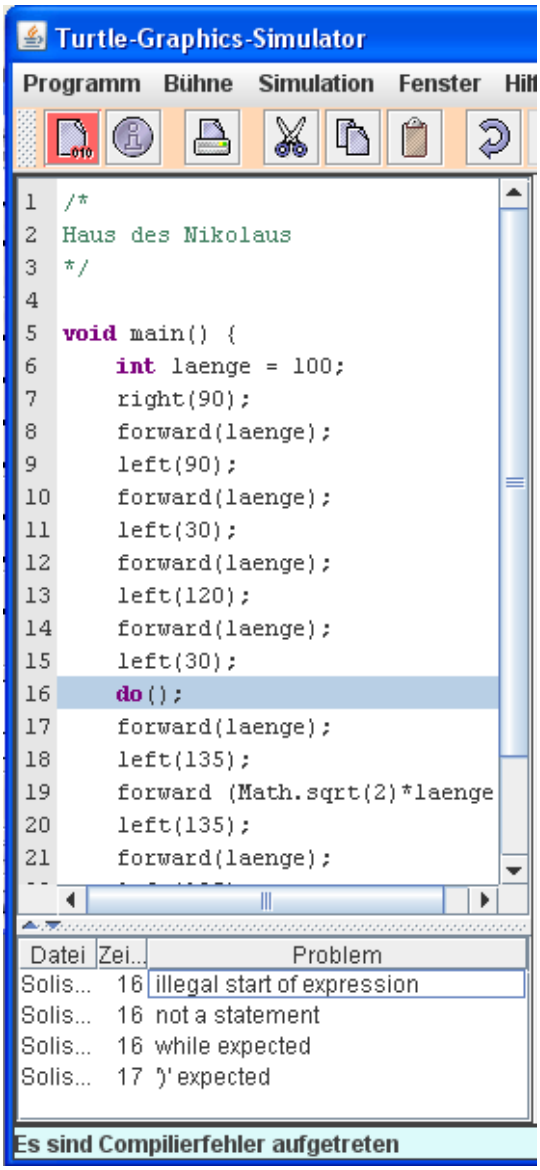


Abb. 5.13: Fehlermeldungen des Compilers

Achtung: Die Interpretation von Fehlermeldungen, die der Compiler ausgibt, ist nicht trivial. Die Meldungen sind nicht immer besonders präzise und oft auch irreführend. Häufig gibt der Compiler mehrere Fehlermeldungen aus, obwohl es sich nur um einen einzelnen Fehler handelt. Deshalb beherzigen Sie gerade am Anfang folgende Hinweise: Arbeiten Sie die Fehlermeldungen immer von oben nach unten ab. Wenn der Compiler eine große Menge von Fehlermeldungen liefert, korrigieren Sie zunächst nur eine Teilmenge und kompilieren Sie danach erneut. Bauen Sie – gerade als Programmieranfänger – auch mal absichtlich Fehler in Ihre Programme ein und schauen Sie sich dann die Fehlermeldungen des Compilers an.

5.4 Verwalten und Gestalten von Turtle-Welten

Die Turtle-Welt, d.h. die Zeichenfläche bzw. das Territorium der Schildkröte hat standardmäßig ein Höhe von 800 und Breite von 600 Pixeln. Die Schildkröte steht in der Mitte und schaut nach oben (0 Grad).

In der Toolbar dienen die Buttons 9 und 10 von links zum Gestalten der Turtle-Welt. Über das Menü „Bühne“ können Turtle-Welten verwaltet werden (siehe auch Abbildung 5.14).

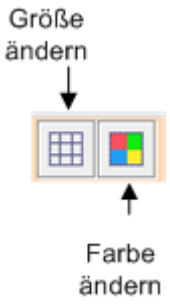


Abb. 5.14: Territorium-Buttons der Toolbar

Normalerweise sollten Sie ein Territorium vor der Ausführung eines Programms gestalten. Es ist jedoch auch möglich, während der Programmausführung noch Umgestaltungen vorzunehmen.

5.4.1 Verändern der Größe der Zeichenfläche

Durch Anklicken des „Größe ändern“-Buttons (9. Toolbar-Button von links) können Sie die Größe des Territoriums verändern. Es öffnet sich eine Dialogbox mit zwei Eingabefelder, in denen Sie die gewünschte Reihen- und Spaltenanzahl eingeben können. Nach Drücken des OK-Buttons schließt sich die Dialogbox und das Territorium erscheint in der angegebenen Größe.

Achtung: Beim Ändern der Größe der Zeichenfläche wird die Zeichenfläche gelöscht und die Schildkröte in der Mitte der neuen Zeichenfläche platziert.

5.4.2 Umplatzieren der Schildkröte auf der Zeichenfläche

Um die Schildkröte auf der Zeichenfläche an eine andere Position zu platzieren, klicken Sie sie mit der Maus an und ziehen sie sie bei gedrückter Maustaste an die gewünschte Position.

5.4.3 Setzen der Blickrichtung der Schildkröte

Um die Blickrichtung der Schildkröte zu ändern, öffnen Sie bitte über das „Fenster“-Menü das Befehlsfenster und klicken dort den left- oder right-Befehl an. Er erscheint eine Dialogbox, in der sie den Winkel, um den sich die Schildkröte drehen soll, eingegeben werden muss.

5.4.4 Ändern der Hintergrundfarbe der Zeichenfläche

Durch Anklicken des „Farbe ändern“-Buttons (10. Toolbar-Button von links) können Sie die Hintergrundfarbe der Zeichenfläche verändern. Es öffnet sich eine Farbauswahl-Dialogbox, in der Sie die gewünschte Frabe auswählen können. Nach Drücken des OK-Buttons schließt sich die Dialogbox und das Territorium erscheint in der angegebenen Hintergrundfarbe.

5.4.5 Löschen der Zeichenfläche

Um die Zeichenfläche zu löschen, öffnen Sie bitte über das „Fenster“-Menü das Befehlsfenster und klicken dort den clean-Befehl an.

5.4.6 Abspeichern der Zeichenfläche

Sie können die aktuelle Zeichenfläche in einer Datei abspeichern und später wieder laden. Zum Abspeichern der aktuellen Zeichenfläche aktivieren Sie im Menü „Bühne“ das Menü-Item „Speichern unter…“. Es öffnet sich eine Dateiauswahl-Dialogbox. Hierin können Sie den Ordner auswählen und den Namen einer Datei eingeben, in die die aktuelle Zeichenfläche gespeichert werden soll.

Weiterhin ist es möglich, die aktuelle Zeichenfläche als Bild (gif- oder png-Datei) abzuspeichern. Eine entsprechende Funktion findet sich im „Bühne“-Menü.

5.4.7 Wiederherstellen einer abgespeicherten Zeichenfläche

Abgespeicherte Zeichenflächen können mit dem „Laden“-Menü-Item des „Bühne“-Menüs wieder geladen werden. Es erscheint eine Dateiauswahl-Dialogbox, in der Sie die zu ladende Datei auswählen können. Nach dem Anklicken des OK-Buttons schließt sich die Dialogbox und die entsprechende Zeichenfläche ist wiederhergestellt.

Achtung: Der Zustand der Zeichenfläche, der vor dem Ausführen der Laden-Funktion Gültigkeit hatte, geht dabei verloren. Speichern Sie ihn daher gegebenenfalls vorher ab.

5.5 Interaktives Ausführen von Turtle-Befehlen

Sie können einzelne Turtle-Befehle oder selbst definierte Funktionen und Prozeduren bspw. zu Testzwecken auch interaktiv ausführen. Aktivieren Sie dazu im Menü „Fenster“ den Eintrag „Befehlsfenster“. Es öffnet sich das so genannte Befehlsfenster (siehe Abbildung 5.15).

Abb. 5.15: Befehlsfenster

5.5.1 Befehlsfenster

Im Befehlsfenster werden die Turtle-Befehle (mit grünlichem Hintergrund) sowie die Prozeduren und Funktionen dargestellt, die im aktuellen Turtle-Programm beim letztmaligen erfolgreichen Compilieren definiert waren (mit orangenem Hintergrund).

Beim Anklicken mit der Maus werden die entsprechenden Befehle jeweils ausgeführt. Die Ausführung erfolgt dabei ohne Anzeige von Zwischenzuständen. D.h. wird bspw. die main-Prozedur angeklickt, wird das entsprechende Programm ohne Visualisierung der einzelnene Befehle „in einem Rutsch“, also sehr schnell ausgeführt.

Dauert die Ausführung eines Befehls mehr als 5 Sekunden (vermutlich enthält dann die Funktion eine Endlosschleife), wird der Befehl abgebrochen und der Turtle-Simulator wird komplett auf seinen Startzustand zurückgesetzt.

5.5.2 Parameter

Besitzt eine Prozedur oder Funktion Parameter, erscheint nach ihrer Aktivierung im Befehlsfenster eine Dialogbox, in der die entsprechenden aktuellen Parameterwerte eingegeben werden müssen. Hinweis: Aktuell wird nur die Eingabe von Werten der Java-Standard-Datentypen (`int`, `boolean`, `float`, ...) sowie die Eingabe von Zeichenketten (Strings) unterstützt.

5.5.3 Rückgabewerte von Funktionen

Bei der Ausführung von Funktionen wird der jeweils gelieferte Wert in einem Dialogfenster dargestellt.

5.5.4 Befehls-Popup-Menü

Alternativ zur Benutzung des Befehlsfensters ist es auch möglich, über ein Popup-Menü die Turtle-Befehle interaktiv auszuführen. Sie können dieses Popup-Menü durch Drücken der rechten Maustaste oberhalb der Schildkröte im Territorium aktivieren (siehe Abbildung 5.16).

Abb. 5.16: Befehls-Popup-Menü

5.6 Ausführen von Turtle-Programmen

Ausgeführt werden können (erfolgreich compilierte) Turtle-Programme mit Hilfe der in Abbildung 5.17 skizzierten Buttons der Toolbar. Alle Funktionen sind darüber hinaus auch über das Menü „Simulation“ aufrufbar.

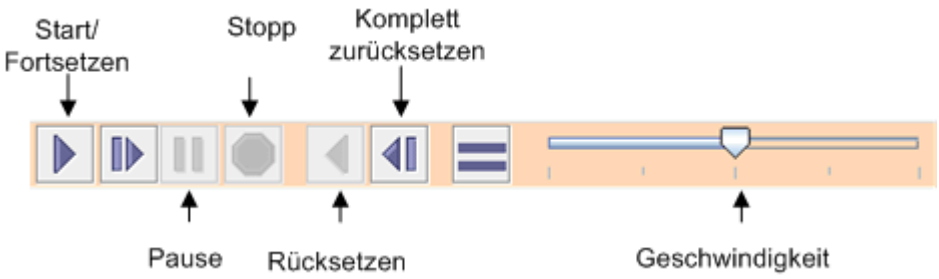


Abb. 5.17: Simulationsbuttons der Toolbar

5.6.1 Starten eines Turtle-Programms

Bevor ein Turtle-Programm ausgeführt werden kann, muss es erfolgreich compiliert worden sein. Gestartet werden kann das aktuelle Turtle-Programm dann durch Anklicken des „Start/Fortsetzen“-Buttons (11. Toolbar-Button von links).

Nach dem Starten eines Turtle-Programms wird die Schildkröte auf der Zeichenfläche aktiv und tut das, was das Programm ihr vorgibt. Während des Ausführens eines Turtle-Programms wird der Editor-Bereich ausgegraut, d.h. es können während der Ausführung eines Programms keine Änderungen am Sourcecode durchgeführt werden.

5.6.2 Stoppen eines Turtle-Programms

Die Ausführung eines Turtle-Programms kann durch Anklicken des „Stopp“-Buttons (14. Button der Toolbar von links) jederzeit abgebrochen werden.

5.6.3 Pausieren eines Turtle-Programms

Möchten Sie ein in Ausführung befindliches Programm (kurzfristig) anhalten, können Sie dies durch Anklicken des „Pause“-Buttons (13. Button der Toolbar von links) tun. Wenn Sie anschließend auf den „Start/Fortsetzen“-Button klicken, wird die Programmausführung fortgesetzt.

5.6.4 Während der Ausführung eines Turtle-Programms

Treten bei der Ausführung eines Programms Laufzeitfehler auf, wird eine Dialogbox geöffnet, die eine entsprechende Fehlermeldung enthält. Nach dem Anklicken des OK-Buttons in der Dialogbox wird das Turtle-Programm beendet. Weiterhin öffnet sich das Konsolen-Fenster, in dem ebenfalls die Fehlermeldung ausgegeben wird.

5.6.5 Einstellen der Geschwindigkeit

Mit dem Schieberegler ganz rechts in der Toolbar können Sie die Geschwindigkeit der Programmausführung beeinflussen. Je weiter links der Regler steht, desto langsamer wird das Programm ausgeführt. Je weiter Sie den Regler nach rechts verschieben, umso schneller flitzt die Schildkröte über den Bildschirm.

5.6.6 Wiederherstellen einer Zeichenfläche

Beim Testen eines Programms recht hilfreich ist der „Rücksetzen“-Button (15. Button der Toolbar von links). Sein Anklicken bewirkt, dass die Zeichenfläche in den Zustand zurückversetzt wird, den sie vor dem letztmaligen Start eines Programms inne hatte.

Über den „Komplett Zurücksetzen“-Button (16. Button der Toolbar von links) ist eine Rücksetzung der Zeichenfläche in den Zustand möglich, der beim Öffnen des Turtle-Simulators gültig war. Sollte es irgendwann einmal bei der Benutzung des Turtle-Simulators zu unerklärlichen Fehlern können, ist es mit Hilfe dieses Buttons möglich, den Turtle-Simulator zu reinitialisieren.

5.7 Debuggen von Turtle-Programmen

Debugger sind Hilfsmittel zum Testen von Programmen. Sie erlauben es, während der Programmausführung den Zustand des Programms zu beobachten und gegebenenfalls sogar interaktiv zu ändern. Damit sind Debugger sehr hilfreich, wenn es um das Entdecken von Laufzeitfehlern und logischen Programmfehlern geht.

Der Debugger des Turtle-Simulators ermöglicht während der Ausführung eines Turtle-Programms das Beobachten des Programmzustands. Sie können sich während der Ausführung eines Turtle-Programms anzeigen lassen, welche Anweisung des Sourcecodes gerade ausgeführt wird und welche Werte die Variablen aktuell speichern. Die interaktive Änderung von Variablenwerten wird aktuell nicht unterstützt.

Die Funktionen des Debuggers sind eng mit den Funktionen zur Programmausführung verknüpft. Sie finden die Funktionen im Menü „Simulation“. Es bietet sich jedoch an, die entsprechenden Buttons der Toolbar zu verwenden. Neben dem „Start/Fortsetzen“-, dem „Pause“- und dem „Stopp“-Button gehören die zwei Buttons „Schrittweise Ausführung“ und „Ablaufverfolgung“ zu den Debugger-Funktionen (siehe auch Abbildung 5.18).

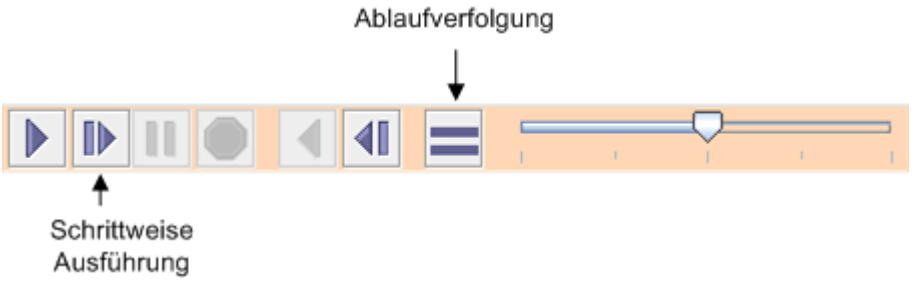


Abb. 5.18: Debugging-Buttons der Toolbar

5.7.1 Beobachten der Programmausführung

Durch Anklicken des Buttons „Ablaufverfolgung“ (17. Button der Toolbar von links) aktivieren bzw. (bei erneuten Anklicken) deaktivieren Sie die Ablaufverfolgung des Debuggers. Bei der Aktivierung öffnet sich dazu das Debugger-Fenster (siehe Abbildung 5.19). Dieses können Sie auch über das Menü „Fenster“ sichtbar bzw. unsichtbar machen.

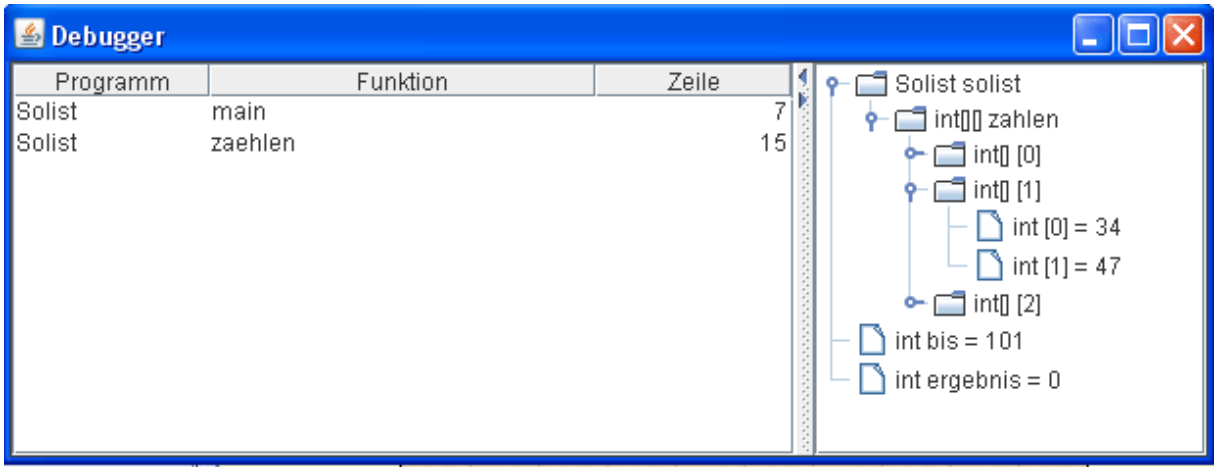


Abb. 5.19: Debugger-Fenster

Ist die Ablaufverfolgung aktiviert, wird bei der Ausführung des Programms im Editor-Bereich der Befehl (bzw. die entsprechende Zeile), der als nächstes ausgeführt wird, blau markiert. Bei einem Prozedur- bzw. Funktionsaufruf wird in die entsprechende Funktion gesprungen. Weiterhin werden im Debugger-Fenster der aktuelle Stack der Funktionsaufrufe (Name der Funktion und aktuelle Position der Ausführung der Funktion) sowie die aktuelle Belegung der Variablen dargestellt.

Im linken Bereich des Debugger-Fensters werden Informationen zu den aktiven Funktionen angezeigt, und zwar jeweils der Name der Funktion und die aktuelle Position der Ausführung der Funktion. Ganz unten erscheint die aktuell aktive

Funktion, darüber gegebenenfalls die Funktion, die diese Funktion aufgerufen hat, usw. Ganz oben steht also immer die `main`-Funktion.

Im rechten Bereich des Debugger-Fensters werden die aktiven Variablen und ihre aktuellen Werte angezeigt. Die Darstellung erfolgt dabei in einem Elementbaum, d.h. bei komplexen Variablen, wie Arrays, können Sie durch Anklicken des Symbols vor dem Variablennamen die Komponenten einsehen.

Auch während die Ablaufverfolgung aktiviert ist, können Sie die Programmausführung durch Anklicken des „Pause“-Buttons anhalten und durch anschließendes Anklicken des „Start/Fortsetzen“-Buttons wieder fortfahren lassen. Auch die Geschwindigkeit der Programmausführung lässt sich mit dem Schieberegler anpassen.

5.7.2 Schrittweise Programmausführung

Mit dem Toolbar-Button „Schrittweise Ausführung“ (12. Button der Toolbar von links) ist es möglich, ein Programm schrittweise, d.h. Anweisung für Anweisung auszuführen. Immer, wenn Sie den Button anklicken, wird die nächste Anweisung (bzw. Zeile) ausgeführt.

Sie können das Programm mit der schrittweisen Ausführung starten. Sie können jedoch auch zunächst das Programm normal starten, dann pausieren und ab der aktuellen Anweisung schrittweise ausführen. Eine normale Programmweiterführung ist jederzeit durch Klicken des „Start“-Buttons möglich.

5.7.3 Breakpoints

Wenn Sie das Programm an einer bestimmten Stelle anhalten möchten, können Sie in der entsprechenden Zeile einen so genannten Breakpoint setzen. Führen Sie dazu vor oder während der Programmausführung im Editor-Bereich mit der Maus einen Doppelklick auf die entsprechende Zeilennummer aus. Breakpoints werden durch violett hinterlegte Zeilennummern dargestellt (siehe Abbildung 5.20). Ein Doppelklick auf einen Breakpoint löscht den Breakpoint wieder. Über der Spalte mit den Zeilennummern lässt sich auch durch Klicken der rechten Maustaste ein Popup-Menü aktivieren, das als Funktionen das Setzen bzw. Entfernen von Breakpoints bzw. das gleichzeitige Löschen aller Breakpoints bietet.

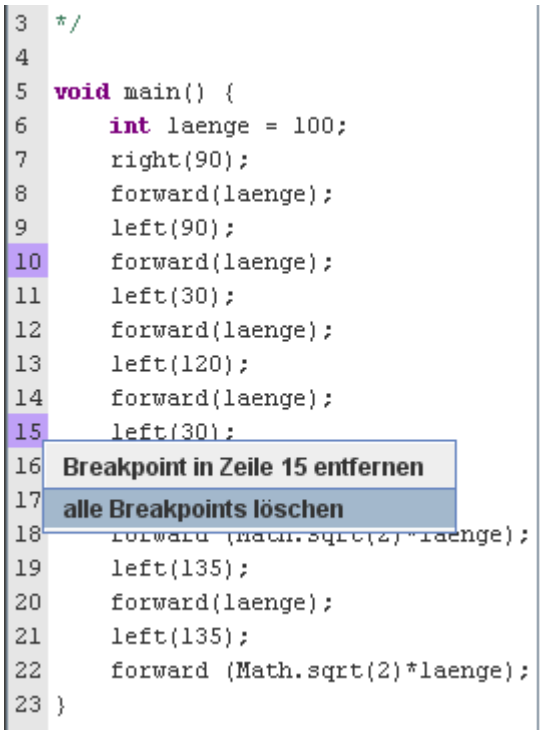


Abb. 5.20: Breakpoints

Wenn Sie ein Programm mit Breakpoints ausführen, wird die Ausführung jedesmal pausiert, wenn eine Zeile mit einem Breakpoint erreicht wird (unabhängig davon, ob die Ablaufverfolgung eingeschaltet ist). Durch Drücken des „Start/Fortsetzen“-Buttons kann die Ausführung des Programms fortgesetzt werden.

5.7.4 Debugger-Fenster

Das Debugger-Fenster wird automatisch bei Aktivierung der Ablaufverfolgung sichtbar gemacht. Über das Menü „Fenster“ lässt es sich jedoch auch explizit sichtbar bzw. unsichtbar machen. Das Fenster besteht aus einem rechten und einem linken Teil innerhalb einer Split-Pane. Der Inhalt des Debugger-Fensters wird nur dann aktualisiert, wenn die Ablaufverfolgung aktiviert ist.

Im linken Bereich des Fensters werden die aktuell aufgerufenen Funktionen und die jeweilige Zeilennummer dargestellt, in der sich die Programmausführung innerhalb der Funktion gerade befindet. Ganz oben steht dabei immer die main-Funktion, ganz unten die zuletzt aufgerufene Funktion.

Im rechten Bereich werden Variablen und deren aktuelle Werte dargestellt. Im Normalfall sind das genau die Variablen, die in der zuletzt aufgerufenen Funktion

(also der im linken Bereich ganz unten stehenden Funktion) gültig sind. Die Darstellung erfolgt dabei in einem Elementbaum. Bei strukturierten Variablen kann durch Mausklick auf das Symbol vor dem Variablennamen die Struktur weiter geöffnet (bzw. wieder geschlossen) werden.

Im pausierten Zustand ist es möglich, sich auch die Werte lokaler Variablen anderer Funktionsinkarnationen anzuschauen. Das ist möglich, indem man im linken Bereich auf den entsprechenden Funktionsnamen klickt.

5.8 Konsole

Die Konsole ist ein zusätzliches Fenster, das bei bestimmten Aktionen automatisch geöffnet wird, sich aber über das Menü „Fenster“ auch explizit öffnen bzw. schließen lässt (siehe Abbildung 5.20).

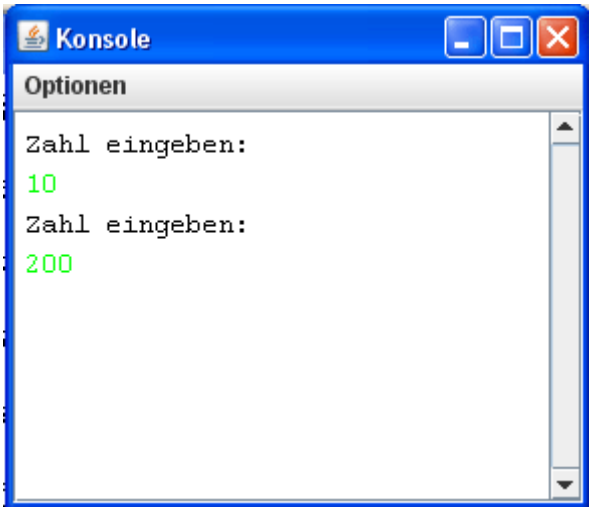


Abb. 5.20: Konsole

Die Konsole ist für die Java-Eingabe mittels `System.in` und die Ausgabe mittels `System.out` und `System.err` zuständig. Wird in Ihrem Programm bspw. der Befehl `System.out.println(„hallo“);` ausgeführt, wird die Zeichenkette `hallo` in der Konsole ausgegeben. `System.out` und `System.err` unterscheiden sich durch eine schwarze (out) bzw. eine rote (err) Ausgabe des entsprechenden Textes. Auch Fehlermeldungen des Turtle-Simulators erscheinen in der Konsole.

Ein Eingabebefehl via `System.in` blockiert das Turtle-Programm so lange, bis der Nutzer eine entsprechende Eingabe in der Konsole getätigt und im Allgemeinen durch Drücken der Enter-Taste abgeschlossen hat. Die folgende Funktion `readInt`

erwartet bspw. vom Nutzer die Eingabe einer Zahl in der Konsole und liefert den entsprechenden Wert an das Turtle-Programm. Wenn der Nutzer in der Konsole den Wert 10 eingibt, bewegt sich die Schildkröte 10 Einheiten nach vorne.

```
void main() {
    System.out.println("Zahl eingeben: ");
    double zahl = readDouble();
    if (zahl > 0) {
        forward(zahl);
    }
}

// Einlesen eines double-Wertes
double readDouble() {
    try {
        java.io.BufferedReader input =
            new java.io.BufferedReader(
                new java.io.InputStreamReader(System.in));
            String eingabe = input.readLine();
       return new Double(eingabe);
      } catch (Throwable exc) {
       return 0;
      }
}
```

Die Konsole enthält in der Menüleiste ein Menü namens „Optionen“. Hierin finden sich drei Menü-Items, über die es möglich ist, den aktuellen Inhalt der Konsole zu löschen, den aktuellen Inhalt der Konsole in einer Datei abzuspeichern bzw. die Konsole zu schließen.

6 Beispielprogramme und Aufgaben

Dem Turtle-Simulator sind sechs Beispielprogramme beigefügt, die Sie über das Menü „Programm“ laden können (Menü-Item „Laden…“). Diese werden in den beiden folgenden Abschnitten erläutert. Der dritte Abschnitt enthält einige Anregungen, sich selbstständig Aufgaben zu überlegen, die die Schildkröte erledigen könnte.

6.1 Haus des Nikolaus

Die Schildkröte zeichnet das Haus des Nikolaus.

Dateiname: nikolaus.turtle

```
/*
Haus des Nikolaus
*/

void main() {
    int laenge = 100;
    right(90);
    forward(laenge);
    left(90);
    forward(laenge);
    left(30);
    forward(laenge);
    left(120);
    forward(laenge);
    left(30);
    forward(laenge);
    left(135);
    forward (Math.sqrt(2)*laenge);
    left(135);
    forward(laenge);
    left(135);
    forward (Math.sqrt(2)*laenge);
}
```

6.2 Koch-Kurve

Die Schildkröte zeichnet die so genannte Koch-Kurve (siehe auch http://mathworld.wolfram.com/KochSnowflake.html).

Dateiname: koch.turtle

```
/*
Koch-Kurve
*/
void main() {
    for (int i=0; i<4; i++) {
        koch(4, 200);
```

```
        left(90);
    }
}

void koch(int t, int s) {
    if (t == 0) {
        forward(s);
    } else {
        koch(t-1, s/3);
        right(60);
        koch(t-1, s/3);
        left(120);
        koch(t-1, s/3);
        right(60);
        koch(t-1, s/3);
    }
}
```

6.3 mn_eck

Die Schildkröte zeichnet ein kreisförmiges Gebilde auf den Bildschirm (siehe auch http://de.wikibooks.org/wiki/Logo:_Beispiele).

Dateiname: mn_eck.turtle

```
/*
m-n-Eck
*/

void main() {
    mn_eck(36, 20);
}

void n_eck(int anzahl, int laenge) {
    for (int i=0; i<anzahl; i++) {
        right(360.0 / anzahl);
        forward(laenge);
     }
}

void mn_eck(int anzahl, int laenge) {
    for (int i=0; i<anzahl; i++) {
        right(360.0 / anzahl);
        n_eck(anzahl, laenge);
    }
}
```

6.4 Pythagoras-Baum

Die Schildkröte zeichnet den Pythagoras-Baum auf den Bildschirm (siehe auch http://www.pohlig.de/Unterricht/Inf2002/Tag18/PythagorasBaum.htm).

Dateiname: pythagoras.turtle

```
/*
Pythagoras-Baum
*/

void main() {
    back(200);
    baum(15, 150);
}

void baum(int stufe, double laenge) {
  if (stufe > 0) {
    forward(laenge);
    left(60);
    baum(stufe-1, 0.7*laenge);
    right(120);
    baum(stufe-1, 0.7*laenge);
    left(60);
    left(180);
    forward(laenge);
    left(180);
  }
}
```

6.5 Spirale

Die Schildkröte zeichnet eine Spirale auf den Bildschirm.

Dateiname: spirale.turtle

```
/*
Spirale
*/

void main() {
    spiral(0, 91);
}

void spiral(int size, int angle) {
    if (size <= 800) {
        forward(size);
        right(angle);
        spiral(size + 2, angle);
    }
}
```

6.6 Spirale (objektorientierte Variante)

Die Schildkröte hat übrigens auch einen Namen, nämlich `oscar`. Darüber ist es möglich, Befehle an die Schildkröte auch in einer objektorientierten Notation zu verwenden:

```
oscar.clean();
oscar.left(90.0);
```

```
oscar.pendown();
```

Die objektorientierte Notation kann auch bei selbst definierten Prozeduren bzw. Funktionen verwendet werden:

```
void linksUm() {
    oscar.left(90.0);
}

void main() {
    oscar.clean();
    oscar.linksUm();
}
```

Lösen wir das fünfte Beispielprogramm nun einmal in dieser objektorientierten Variante.

Dateiname: OOspirale.turtle

```
/*
Spirale (OO-Variante)
*/

void main() {
    oscar.spiral(0, 91);
}

void spiral(int size, int angle) {
    if (size <= 800) {
        oscar.forward(size);
        oscar.right(angle);
        oscar.spiral(size + 2, angle);
    }
}
```

6.7 Aufgaben

Im WWW finden sich viele Beispiele und auch Aufgaben im Umfeld der Turtle-Graphics. Schauen Sie am besten mal rein. Einge nützliche URLs sind:

- http://de.wikipedia.org/wiki/Logo_(Programmiersprache)
- http://de.wikibooks.org/wiki/Logo:_Turtle-Grafik
- http://edu.kde.org/kturtle/
- http://pagesperso-orange.fr/logoplus/dlogo.htm
- http://www.home.hs-karlsruhe.de/~pach0003/informatik_1/aufgaben/turtle.html
- http://wiki.zum.de/Java/Turtle-Grafik
- http://www.oberstufeninformatik.de/info11/turtle/turtle.html

7 Literatur zum Erlernen der Programmierung

Der Turtle-Simulator ist ein Werkzeug, das Programmierern beim praktischen Erlernen der (imperativen) Programmierung hilft. Es macht einfach Spaß, mit der Schildkröte hübsche Zeichnungen zu erstellen. Der Turtle-Simulator ist jedoch kein Lehrbuch. Um mit dem Turtle-Simulator programmieren zu lernen, sollten Sie sich ein begleitendes Lehrbuch anschaffen.

Wie bei der Turtle-Graphics handelt es sich auch beim so genannten Java-Hamster-Modell um eine Miniprogrammierwelt. Zu dem Hamster-Modell gibt es ein Lehrbuch: „**Programmieren spielend gelernt mit dem Java-Hamster-Modell**“ von Dietrich Boles, erschienen im Vieweg+Teubner-Verlag. In diesem Buch werden die grundlegenden Konzepte der Programmierung anhand des Java-Hamster-Modells vorgestellt. Das Buch geht langsam und schrittweise vor. Es enthält viele Beispiele und auch eine Reihe von Aufgaben. Dieses Buch kann uneingeschränkt auch als Begleitbuch für Turtle-Programmierer empfohlen werden. Das Buch ist dabei insbesondere für solche Programmieranfänger zu empfehlen, die sich beim Erlernen der Programmierung schwer tun. Mehr Informationen und auch Links zu Leseproben gibt es auf der Java-Hamster-Website www.java-hamster-modell.de.

Es gibt heutzutage unzählige Lehrbücher zu Java und es ist schwer zu sagen, für wen welches Buch das geeignetste ist. Viele Bücher stehen bei Amazon oder Google-Books zumindest auszugsweise online zur Verfügung und ich kann nur empfehlen, über diese Online-Angebote selbst einmal in die Bücher hineinzuschnuppern. Neben dem Java-Hamster-Buch kann ich die beiden weiteren Bücher insbesondere für Programmieranfänger empfehlen:

- Ratz, D., Scheffler, J., Seese, D. und Wiesenberger, J.: „Grundkurs Programmieren in Java, Band 1“, Hanser-Verlag.
- Heinisch, C., Müller, F. und Goll, J..: „Java als erste Programmiersprache“, Vieweg+Teubner.

Wenn Sie noch überhaupt keine Kenntnisse der Programmierung haben, empfehle ich Ihnen, sich Informationen zu den allgemeinen Grundlagen der Programmierung auf der folgenden Website durchzulesen (Leseprobe des Java-Hamster-Buches): http://www-is.informatik.uni-oldenburg.de/~dibo/hamster/leseprobe/node3.html.