

# **Benutzungshandbuch**

## **TicTacToe-Simulator**

Version 1.0 (02.10.2009)

Dietrich Boles

Universität Oldenburg



## Inhaltsverzeichnis

1	Einleitung .....	7
1.1	TicTacToe .....	7
1.2	Der TicTacToe-Simulator .....	8
1.3	Voraussetzungen.....	8
1.4	Anmerkungen .....	8
1.5	Aufbau des Benutzerhandbuch .....	9
2	Installation.....	10
2.1	Voraussetzungen.....	10
2.2	Download, Installation und Start.....	10
3	Grundlagen der TicTacToe-Miniprogrammierwelt.....	11
3.1	Regeln .....	11
3.2	Befehle .....	11
3.3	TicTacToe-Programme.....	12
3.4	Beispielprogramm.....	13
3.5	Aufgaben .....	13
4	Ihr erstes TicTacToe-Programm .....	14
4.1	Eingeben eines TicTacToe-Programms .....	15
4.2	Compilieren eines TicTacToe-Programms .....	16
4.3	Ausführen eines TicTacToe-Programms .....	17

4.4	Debuggen eines TicTacToe-Programms.....	18
4.5	Zusammenfassung .....	20
5	Bedienung des TicTacToe-Simulators .....	21
5.1	Grundfunktionen .....	22
5.1.1	Anklicken .....	22
5.1.2	Tooltipps .....	22
5.1.3	Button .....	22
5.1.4	Menü.....	23
5.1.5	Toolbar.....	24
5.1.6	Popup-Menü .....	24
5.1.7	Eingabefeld.....	24
5.1.8	Dialogbox.....	24
5.1.9	Dateiauswahl-Dialogbox .....	25
5.1.10	Elementbaum .....	26
5.1.11	Split-Pane .....	27
5.2	Verwalten und Editieren von TicTacToe-Programmen.....	27
5.2.1	Schreiben eines neuen TicTacToe-Programms.....	29
5.2.2	Ändern des aktuellen TicTacToe-Programms.....	29
5.2.3	Löschen des aktuellen TicTacToe-Programm .....	29
5.2.4	Abspeichern des aktuellen TicTacToe-Programms .....	29
5.2.5	Öffnen eines einmal abgespeicherten TicTacToe-Programms .....	29
5.2.6	Drucken eines TicTacToe-Programms .....	29

5.2.7	Editier-Funktionen.....	30
5.3	Compilieren von TicTacToe-Programmen.....	31
5.3.1	Compilieren.....	31
5.3.2	Beseitigen von Fehlern .....	31
5.4	Ausführen von TicTacToe-Programmen.....	33
5.4.1	Starten eines TicTacToe-Programms .....	33
5.4.2	Stoppen eines TicTacToe-Programms .....	33
5.4.3	Pausieren eines TicTacToe-Programms.....	33
5.4.4	Während der Ausführung eines TicTacToe-Programms.....	33
5.4.5	Einstellen der Geschwindigkeit .....	34
5.4.6	Wiederherstellen einer Zeichenfläche.....	34
5.5	Debuggen von TicTacToe-Programmen .....	34
5.5.1	Beobachten der Programmausführung.....	35
5.5.2	Schrittweise Programmausführung.....	36
5.5.3	Breakpoints .....	36
5.5.4	Debugger-Fenster.....	37
5.6	Konsole .....	38
5.7	Befehlsfenster .....	39
5.7.1	Befehlsfenster.....	39
5.7.2	Parameter .....	39
5.7.3	Rückgabewerte von Funktionen.....	39
5.7.4	Anmerkung .....	40

6	Literatur zum Erlernen der Programmierung.....	41
---	--	----

# 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 Programmieranfänger sehr gut nachvollziehen, was ihr Programm tut. Fehler lassen sich sehr schnell entdecken und beheben.

## 1.1 TicTacToe

Tic Tac Toe (auch: Drei gewinnt, XXO, Kreis und Kreuz, Dodelschach oder engl. Noughts and Crosses) ist ein klassisches, einfaches Zweipersonen-Strategiespiel, dessen Geschichte sich bis ins 12. Jahrhundert v. Chr. zurückverfolgen lässt. Auf einem 3x3 Felder großen Spielfeld machen die beiden Spieler abwechselnd ihre Zeichen (ein Spieler Kreuze, der andere Kreise). Der Spieler, der als erstes drei seiner Zeichen in einer Reihe, Spalte oder einer der beiden Hauptdiagonalen setzen kann, gewinnt. (Quelle: Wikipedia)

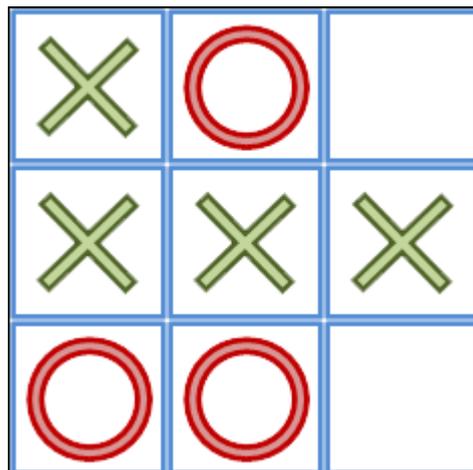


Abb. 1.1: TicTacToe

In der vorliegenden TicTacToe-Miniprogrammierwelt wird nun eine kleine Menge an Befehlen zur Verfügung gestellt, mit deren Hilfe Programmieranfänger versuchen sollen, Programme zu schreiben, die es ermöglichen, dass Menschen gegen Menschen, Menschen gegen das Programm oder das Programm gegen sich selbst TicTacToe spielt. Die Programme werden im Folgenden TicTacToe-Programme genannt. Als zugrunde liegende Programmiersprache wird Java (mit leichten Abänderungen) genutzt.

## **1.2 Der TicTacToe-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 unterstützt der TicTacToe-Simulator. Dieser stellt eine Reihe von Werkzeugen zum Erstellen und Ausführen von TicTacToe-Programmen zur Verfügung: einen Editor zum Eingeben und Verwalten von TicTacToe-Programmen, einen Compiler zum Übersetzen von TicTacToe-Programmen, einen Territoriumsgestalter zum Gestalten und Verwalten von TicTacToe-Spielfeldern, einen Interpreter zum Ausführen von TicTacToe-Programmen und einen Debugger zum Testen von TicTacToe-Programmen. Der TicTacToe-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 von Miniprogrammierwelten sind Schüler oder Studierende ohne oder mit nur wenig Programmiererfahrung, die die Grundlagen der (imperativen) Programmierung erlernen und dabei ein wenig Spaß haben wollen. Kenntnisse im Umgang mit Computern sind wünschenswert. Der TicTacToe-Simulator ist dabei kein Lehrbuch sondern ein Werkzeug, das das Erlernen der Programmierung unterstützt. Auf gute begleitende Lehrbücher zum Erlernen der Programmierung wird in Kapitel 6 (Literatur zum Programmieren lernen) hingewiesen.

## **1.4 Anmerkungen**

Der TicTacToe-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 TicTacToe-Simulators ist die Bühne ein TicTacToe-Spielfeld, in der ein (unsichtbarer) Spieler – der Solist – agiert. Er platziert Spielfiguren in die Felder des Spielfelds. Wenn Ihnen also beim Umgang mit dem TicTacToe-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](http://www.programmierkurs-java.de/solist).

## **1.5 Aufbau des Benutzerhandbuch**

Das Benutzungshandbuch ist in 6 Kapitel gegliedert. Nach dieser Einleitung wird in Kapitel 2 die Installation des TicTacToe-Simulators beschrieben. Kapitel 3 erläutert die Grundlagen der TicTacToe-Miniprogrammierwelt. Wie Sie Ihr erstes TicTacToe-Programm zum Laufen bringen, erfahren Sie kurz und knapp in Kapitel 4. Dieses enthält eine knappe Einführung in die Elemente und die Bedienung des TicTacToe-Simulators. Sehr viel ausführlicher geht dann Kapitel 5 auf die einzelnen Elemente und die Bedienung des TicTacToe-Simulators ein. Kapitel 6 enthält letztendlich Hinweise zu guter Begleitliteratur zum Erlernen der Programmierung mit Java.

## 2 Installation

### 2.1 Voraussetzungen

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

Beim ersten Start sucht der TicTacToe-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 TicTacToe-Simulator arbeitet. In der Version 1.0 wird allerdings nur deutsch unterstützt.

### **3 Grundlagen der TicTacToe-Miniprogrammierwelt**

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.

Die TicTacToe-Miniprogrammierwelt ist ein spezielles didaktisches Modell zum Erlernen der Programmierung. In der TicTacToe-Miniprogrammierwelt nimmt ein virtueller (unsichtbarer) Spieler die Rolle des Computers ein. Diesem Spieler können ähnlich wie einem Computer Befehle erteilt werden, die dieser ausführt.

Ihnen als Programmierer wird nun die Aufgabe gestellt, Programme zu entwickeln, die es ermöglichen, dass Menschen gegen Menschen, Menschen gegen Ihr Programm oder Ihr Programm gegen sich selbst TicTacToe spielt, wobei Sie den virtuellen Spieler notwendige Aktionen, wie das Platzieren von Spielfiguren in Felder, ausführen lassen können. Ihre Programme entwickeln Sie in der TicTacToe-Sprache - eine Programmiersprache, die fast vollständig der Programmiersprache Java entspricht.

#### **3.1 Regeln**

Tic Tac Toe (auch: Drei gewinnt, XXO, Kreis und Kreuz, Dodenschach oder engl. Noughts and Crosses) ist ein klassisches, einfaches Zweipersonen-Strategiespiel, dessen Geschichte sich bis ins 12. Jahrhundert v. Chr. zurückverfolgen lässt. Auf einem 3x3 Felder großen Spielfeld machen die beiden Spieler abwechselnd ihre Zeichen (ein Spieler Kreuze, der andere Kreise). Der Spieler, der als erstes drei seiner Zeichen in einer Reihe, Spalte oder einer der beiden Hauptdiagonalen setzen kann, gewinnt. Es beginnt der Kreuz-Spieler. (Quelle: Wikipedia)

#### **3.2 Befehle**

Der virtuelle (unsichtbare) Spieler der TicTacToe-Welt kennt nur zwei Befehle. Sie können sich den Spieler quasi als einen virtuellen Prozessor vorstellen, der im Gegensatz zu realen Computer-Prozessoren in der Lage ist, mit einem kleinen Grundvorrat an Befehlen u.a. regelkonform Spielfiguren auf dem Spielfeld zu platzieren.

Die Befehle des TicTacToe-Spielers – im Folgenden TicTacToe-Befehle genannt – sind:

- `Spielzug warten(int figur)`: Auf den Spielzug eines Menschen warten. Genauer: Bei der Ausführung dieses Befehls wartet Ihr Programm darauf, dass ein Mensch das Feld des Spielfeldes mit der Maus anklickt, auf dem eine Spielfigur platziert werden soll. Der entsprechende Spielzug wird zurückgeliefert. Der Parameter gibt an, welche Spielfigur am Zug ist (Kreuz = 1, Kreis = 2). Es gibt einen Laufzeitfehler, wenn eine ungültige oder die falsche Spielfigur übergeben wird.
- `void ziehen(Spielzug zug)`: Der als Parameter übergeben Spielzug wird auf dem Spielfeld ausgeführt. Es gibt einen Laufzeitfehler, wenn der Spielzug ungültig ist.

Folgende Klassen sind für die Nutzung der Befehle notwendig bzw. nützlich:

```
class Spielzug implements java.io.Serializable {
    int figur; // KREUZ, KREIS, KEINE
    int spalte; // 0 .. 2
    int reihe; // 0 .. 2
}

class Figur {
    final static int KREUZ = 1;
    final static int KREIS = 2;
    final static int KEINE = 0;
}
```

### 3.3 *TicTacToe-Programme*

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

- Anders als in Java bedarf es in der TicTacToe-Sprache keiner Klassen-Definition.
- In der TicTacToe-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()`.
- TicTacToe-Programme sind imperative, keine objektorientierten Programme. Es brauchen also keine Objekte erzeugt zu werden. Trotzdem müssen Prozeduren und Funktionen nicht als `static` deklariert werden.
- Ein TicTacToe-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 TicTacToe-Programms bewirkt die automatische Ausführung der main-Prozedur.

### 3.4 Beispielprogramm

In folgendem Programm werden zwei legale Spielzüge ausgeführt:

```
void main() {
    ersterZug();
    zweiterZug();
}

void ersterZug() {
    Spielzug zug = new Spielzug();
    zug.figur = Figur.KREUZ;
    zug.reihe = 0;
    zug.spalte = 0;
    ziehen(zug);
}

void zweiterZug() {
    Spielzug zug = new Spielzug();
    zug.figur = Figur.KREIS;
    zug.reihe = 2;
    zug.spalte = 1;
    ziehen(zug);
}
```

### 3.5 Aufgaben

Entwickeln Sie mit den zur Verfügung stehenden TicTacToe-Befehlen Programme, die es ermöglichen, Tic-Tac-Toe zu spielen:

- Schreiben Sie ein Programm, das es ermöglicht, dass ein Mensch gegen einen anderen Menschen Tic-Tac-Toe spielt.
- Schreiben Sie ein Programm, das es ermöglicht, dass ein Mensch gegen Ihr Programm Tic-Tac-Toe spielt. Ihr Programm kann zunächst seine Spielzüge per Zufall berechnen.
- Schreiben Sie ein Programm, das es ermöglicht, dass ein Mensch gegen Ihr Programm Tic-Tac-Toe spielt. Überlegen Sie sich nun eine Spielstrategie, so dass Ihr Programm auch mal gegen den Menschen gewinnt.
- Schreiben Sie ein Programm, das es ermöglicht, dass zwei Programme gegeneinander Tic-Tac-Toe spielen. Beide Programme müssen Sie natürlich (mit unterschiedlichen Spielstrategien) als Teilprogramme in Ihr Programm einbauen und gegeneinander antreten lassen.

## 4 Ihr erstes TicTacToe-Programm

Nachdem Sie den TicTacToe-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 TicTacToe-Programme ein und im Simulationsbereich führen Sie TicTacToe-Programme aus.

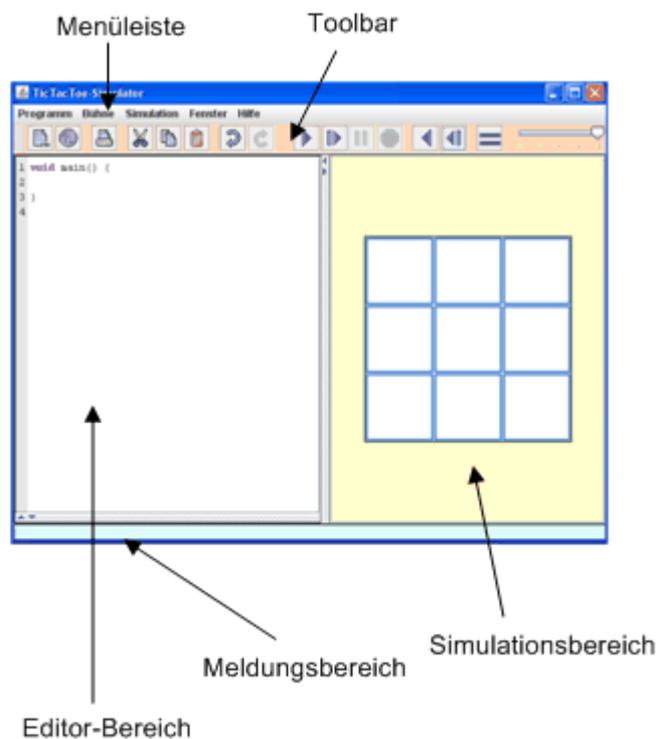


Abb. 4.1: TicTacToe-Simulator nach dem Öffnen

In den folgenden Abschnitten wird im Detail beschrieben, was Sie machen müssen, um Ihr erstes TicTacToe-Programm zu schreiben und auszuführen. Insgesamt müssen/können vier Stationen durchlaufen werden:

- Eingeben eines TicTacToe-Programms
- Compilieren eines TicTacToe-Programms
- Ausführen eines TicTacToe-Programms
- Debuggen eines TicTacToe-Programms

## 4.1 Eingeben eines TicTacToe-Programms

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

Unser erstes Programm soll folgendes bewirken: Zunächst wird ein Mensch aufgefordert, ein Feld anzuklicken, in das ein Kreuz gezeichnet werden soll. Anschließend platziert der Computer einen Kreis in ein anderes Feld. Wir klicken in den Editor-Bereich und tippen dort wie in einem normalen Editor bzw. Textverarbeitungsprogramm, wie Microsoft Word, die entsprechenden TicTacToe-Befehle ein, so dass letztlich folgendes im Eingabebereich steht:

```
void main() {
    // Mensch gibt erstes Kreuz ein
    Spielzug zug = warten(Figur.KREUZ);

    // Computerzug
    ersterComputerZug(zug);
}

void ersterComputerZug(Spielzug menschZug) {
    Spielzug zug = new Spielzug();
    zug.figur = Figur.KREIS;
    zug.reihe = 0;
    if (menschZug.spalte != 0) {
        zug.spalte = 0;
    } else {
        zug.spalte = 1;
    }
    ziehen(zug);
}
```

Das ist unser erstes TicTacToe-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, ...).

## 4.2 Compilieren eines TicTacToe-Programms

Nachdem wir unser TicTacToe-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 4.2).

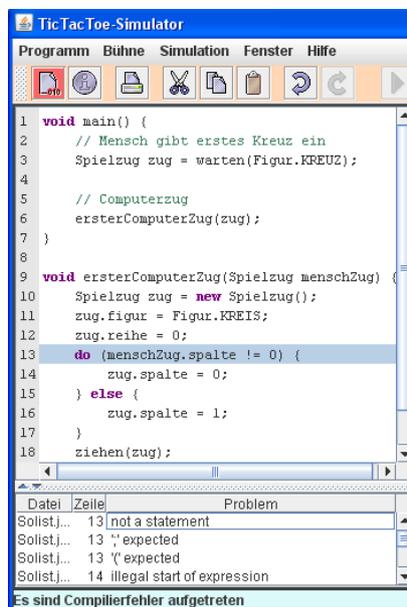


Abb. 4.2: Fehlermeldungen des Compilers

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!

### 4.3 Ausführen eines TicTacToe-Programms

Nach dem erfolgreichen Compilieren ist es endlich soweit: Wir können beobachten, wie die Zahlen im Spielfeld platziert werden. Macht der virtuelle Spieler wirklich das, was wir ihm durch unser Programm beigebracht haben?

Zum Steuern der Programmausführung dienen die Buttons rechts in der Toolbar. Durch Anklicken des „Simulation starten“-Buttons (9. Button von links) starten wir das eben eingegebene und compilierte Programm. Wenn Sie bis hierhin alles richtig gemacht haben, sollte Sie der Simulator auffordern, ein Feld anzuklicken. In diesem erscheint dann ein Kreuz und entweder in Feld (0/0) oder Feld (0/1) erscheint anschließend ein Kreis (der Computerzug) (siehe auch Abbildung 4.3). Herzlichen Glückwunsch zu Ihrem ersten TicTacToe-Programm, auch wenn es natürlich noch unvollständig ist.

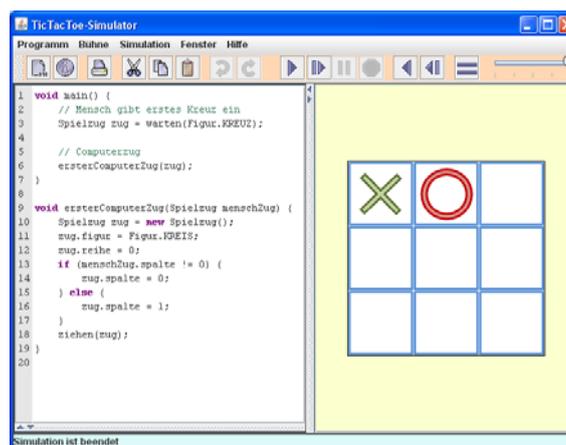


Abb. 4.3: Endzustand des Beispielprogramms

Wollen Sie die Programmausführung anhalten, können Sie dies durch Anklicken des „Simulation pausieren“-Buttons (11. Button von links) erreichen. Der virtuelle Spieler pausiert so lange, bis Sie wieder den „Simulation starten/fortsetzen“-Button (9. Button von links) anklicken. Dann fährt der Spieler mit seiner Arbeit fort. Das Programm vorzeitig komplett abbrechen, können Sie mit Hilfe des „Simulation beenden“-Buttons (12. Button von links).

Das Anklicken des „Rücksetzen“-Buttons (13. Button von links) bzw. „Komplett zurücksetzen“-Button (14. Button von links) bewirkt das Leeren des Spielfelds. Anmerkung: Die Buttons werden automatisch von Solist generiert, haben aber im TicTacToe-Simulator keine wirkliche Bedeutung. Denn: Beim Start eines Programms wird das Spielfeld eh immer komplett geleert.

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 der virtuelle Spieler Arbeit. Je weiter Sie ihn nach rechts verschieben, umso schneller führt er Ihre Anweisungen aus.

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

#### **4.4 Debuggen eines TicTacToe-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 (15. 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 sind 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).

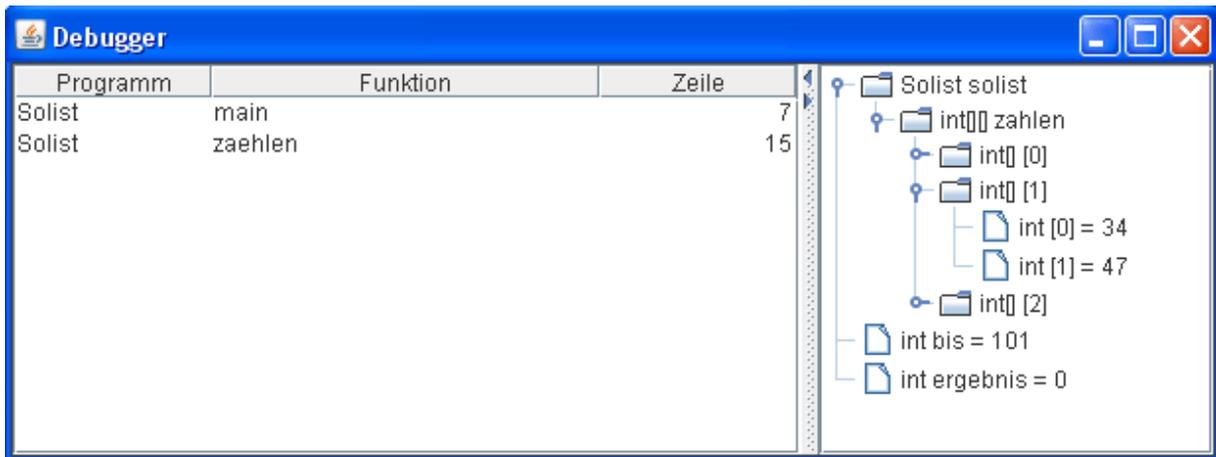


Abb. 4.4: Debugger-Fenster

Jetzt können Sie über den „Schritt“-Button (10. 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 Prozedur-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.

## **4.5 Zusammenfassung**

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

Der TicTacToe-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 TicTacToe-Simulators

Im letzten Abschnitt haben Sie eine kurze Einführung in die Funktionalität des TicTacToe-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 TicTacToe-Simulator starten, öffnen sich ein Fenster mit dem Titel „TicTacToe-Simulator“. Abbildung 5.1 skizziert die einzelnen Komponenten des Fensters.

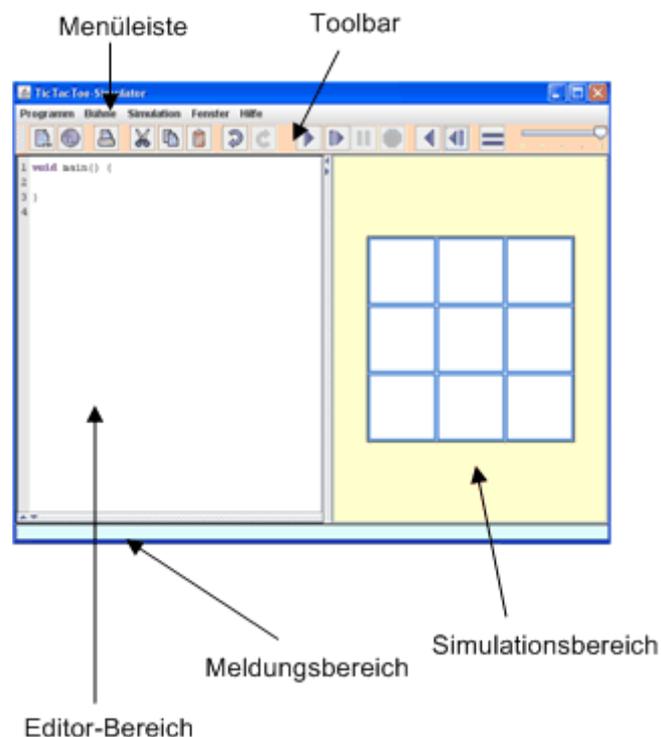


Abb. 5.1: Komponenten des Simulator-Fensters

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 TicTacToe-Programmen erfolgt im Editor-Bereich links und die Ausführung von TicTacToe-Programmen wird im Simulationsbereich (auch „Bühne“ genannt) visualisiert.

Als Hauptfunktionsbereiche des TicTacToe-Simulators lassen sich identifizieren:

- Verwalten und Editieren von TicTacToe-Programmen
- Compilieren von TicTacToe-Programmen
- Ausführen von TicTacToe-Programmen
- Debuggen von TicTacToe-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 Tooltips

Als *Tooltips* 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 Tooltips werden bestimmte Informationen ausgegeben.



Abb. 5.2: Tooltip

### 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 Tooltips 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 Dialogbox 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.

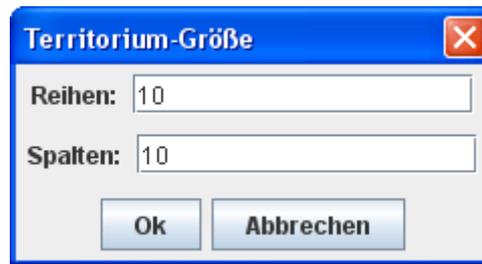


Abb. 5.7: Dialogbox mit Eingabefeldern

### 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.

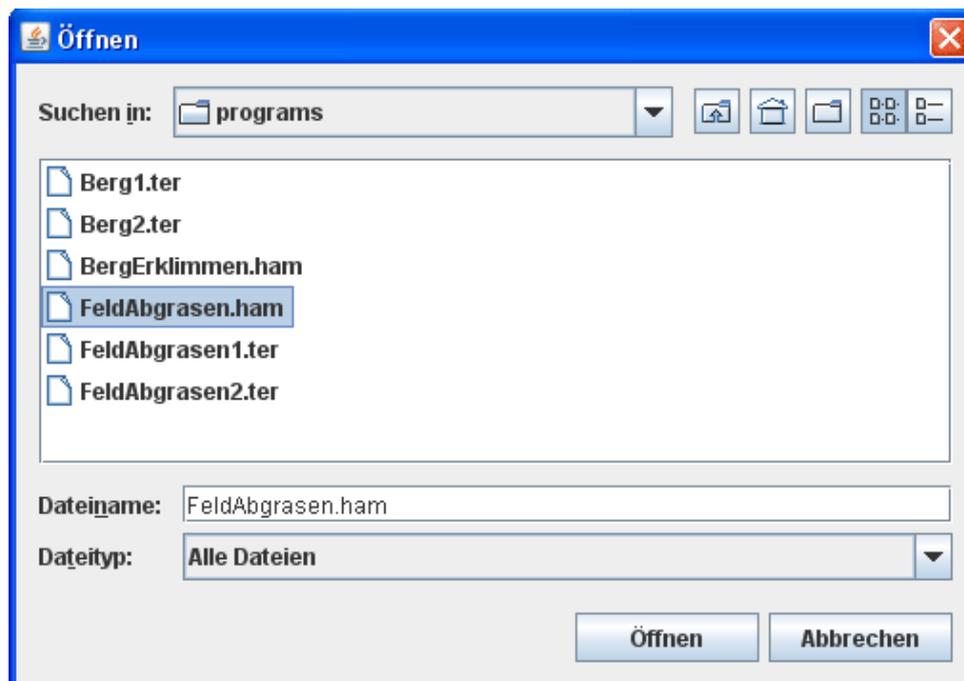


Abb. 5.8: Dateiauswahl-Dialogbox

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.

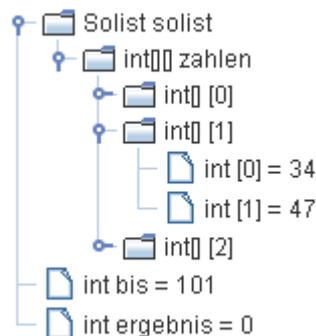
Dateiauswahl-Dialogboxen stellen jedoch noch zusätzliche Funktionalitäten bereit. Durch Doppelklick auf einen Ordner kann man in den entsprechenden Ordner wechseln. Es werden dann anschließend die Dateien und Unterordner dieses Ordners im mittleren Bereich angezeigt. Um zu einem übergeordneten Ordner 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. Unterstrukturen 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.

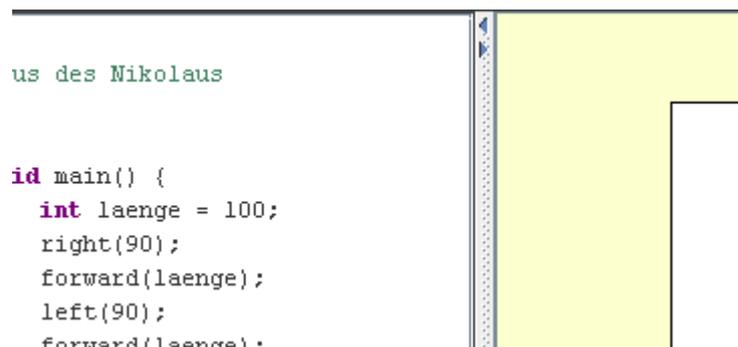


Abb. 5.10: Split-Pane

## 5.2 Verwalten und Editieren von TicTacToe-Programmen

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

```

1 void main() {
2     // Mensch gibt erstes Kreuz ein
3     Spielzug zug = warten(Figur.KREUZ);
4
5     // Computerzug
6     ersterComputerZug(zug);
7 }
8
9 void ersterComputerZug(Spielzug menschZug) {
10    Spielzug zug = new Spielzug();
11    zug.figur = Figur.KREIS;
12    zug.reihe = 0;
13    if (menschZug.spalte != 0) {
14        zug.spalte = 0;
15    } else {
16        zug.spalte = 1;
17    }
18    ziehen(zug);
19 }
20

```

Abb. 5.11: Editor-Bereich des TicTacToe-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 Tooltip, der die Funktionalität des Buttons anzeigt. Die für das Editieren von Programmen wichtigen Buttons der Toolbar werden in Abbildung 5.12 skizziert.

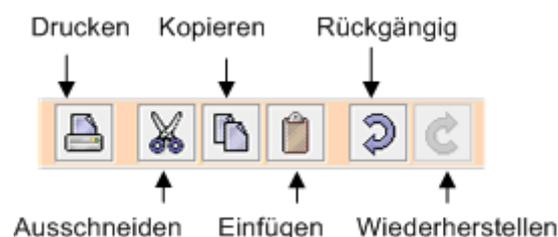


Abb. 5.12: Editor-Buttons der Toolbar

### **5.2.1 Schreiben eines neuen TicTacToe-Programms**

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

### **5.2.2 Ändern des aktuellen TicTacToe-Programms**

Möchten Sie Teile des aktuellen TicTacToe-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 TicTacToe-Programm**

Komplett löschen können Sie das aktuelle TicTacToe-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 TicTacToe-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 TicTacToe-Programms**

Möchten Sie ein einmal abgespeichertes TicTacToe-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 TicTacToe-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 TicTacToe-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 TicTacToe-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 Compiler-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) kompiliert 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 Fehlermeldungs-bereich schließt sich dann automatisch wieder.

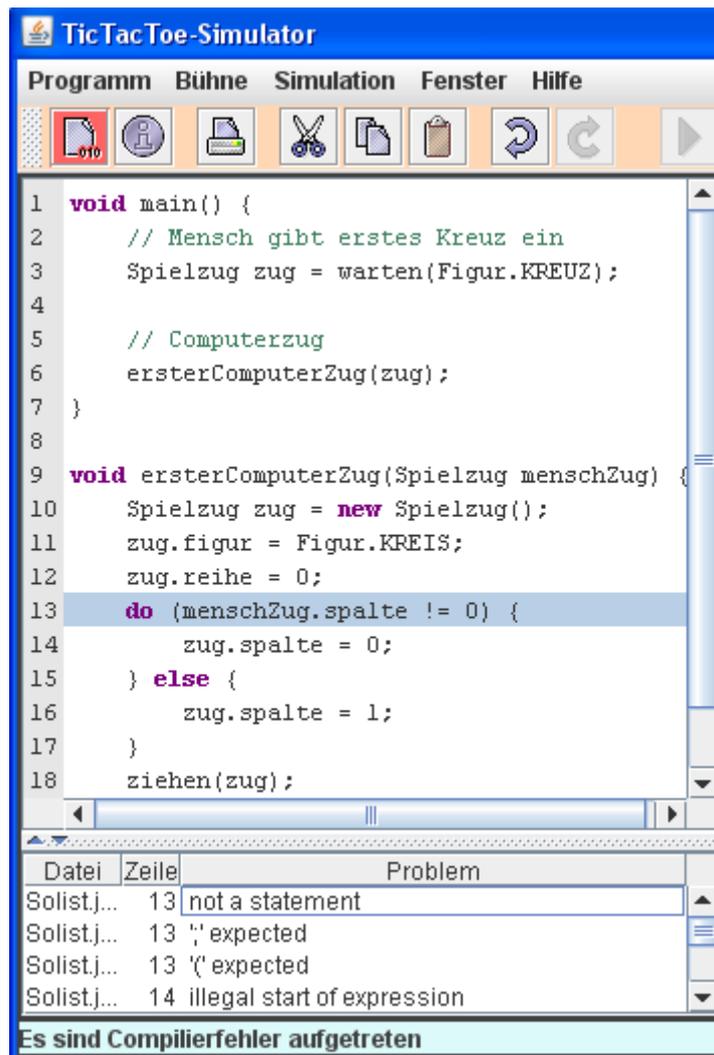


Abb. 5.13: Fehlermeldungen des Compilers

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 Ausführen von TicTacToe-Programmen

Ausgeführt werden können (erfolgreich compilierte) TicTacToe-Programme mit Hilfe der in Abbildung 5.14 skizzierten Buttons der Toolbar. Alle Funktionen sind darüber hinaus auch über das Menü „Simulation“ aufrufbar.

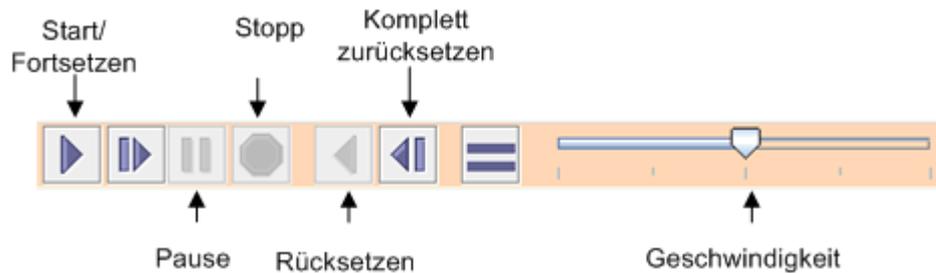


Abb. 5.14: Simulationsbuttons der Toolbar

### 5.4.1 Starten eines TicTacToe-Programms

Bevor ein TicTacToe-Programm ausgeführt werden kann, muss es erfolgreich compiliert worden sein. Gestartet werden kann das aktuelle TicTacToe-Programm dann durch Anklicken des „Start/Fortsetzen“-Buttons (9. Toolbar-Button von links).

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

### 5.4.2 Stoppen eines TicTacToe-Programms

Die Ausführung eines TicTacToe-Programms kann durch Anklicken des „Stopp“-Buttons (12. Button der Toolbar von links) jederzeit abgebrochen werden.

### 5.4.3 Pausieren eines TicTacToe-Programms

Möchten Sie ein in Ausführung befindliches Programm (kurzfristig) anhalten, können Sie dies durch Anklicken des „Pause“-Buttons (11. Button der Toolbar von links) tun. Wenn Sie anschließend auf den „Start/Fortsetzen“-Button klicken, wird die Programmausführung fortgesetzt.

### 5.4.4 Während der Ausführung eines TicTacToe-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 TicTacToe-Programm beendet. Weiterhin öffnet sich das Konsolen-Fenster, in dem ebenfalls die Fehlermeldung ausgegeben wird.

#### **5.4.5 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 agiert der virtuelle Spieler.

#### **5.4.6 Wiederherstellen einer Zeichenfläche**

Beim Testen eines Programms recht hilfreich ist der „Rücksetzen“-Button (13. Button der Toolbar von links). Sein Anklicken bewirkt, dass das Spielfeld in den Zustand zurückversetzt wird, den es vor dem letztmaligen Start eines Programms inne hatte.

Über den „Komplett Zurücksetzen“-Button (14. Button der Toolbar von links) ist eine Rücksetzung des Spielfelds in den Zustand möglich, der beim Öffnen des TicTacToe-Simulators gültig war. Sollte es irgendwann einmal bei der Benutzung des TicTacToe-Simulators zu unerklärlichen Fehlern können, ist es mit Hilfe dieses Buttons möglich, den TicTacToe-Simulator zu reinitialisieren.

Anmerkung: Die beiden Buttons werden automatisch von Solist generiert. Sie haben im TicTacToe-Simulator keine wirkliche Bedeutung, denn beim Starten eines Programms wird eh das Spielfeld zunächst immer gelöscht.

### **5.5 Debuggen von TicTacToe-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 TicTacToe-Simulators ermöglicht während der Ausführung eines TicTacToe-Programms das Beobachten des Programmzustands. Sie können sich während der Ausführung eines TicTacToe-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 5.15).

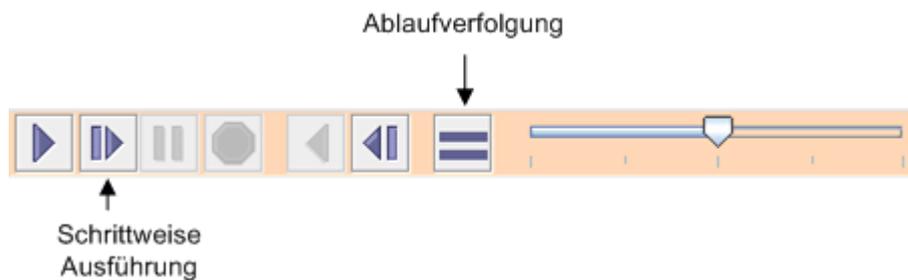


Abb. 5.15: Debugging-Buttons der Toolbar

### 5.5.1 Beobachten der Programmausführung

Durch Anklicken des Buttons „Ablaufverfolgung“ (15. 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 5.16). Dieses können Sie auch über das Menü „Fenster“ sichtbar bzw. unsichtbar machen.

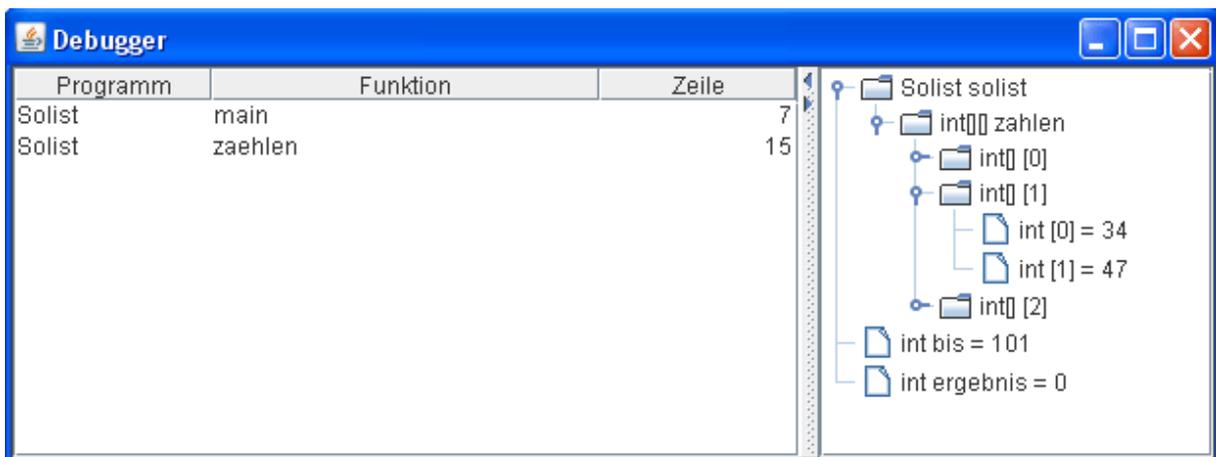


Abb. 5.16: Debugger-Fenster

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.

### **5.5.2 Schrittweise Programmausführung**

Mit dem Toolbar-Button „Schrittweise Ausführung“ (10. 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.

### **5.5.3 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 5.17). 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.

```

2 // Mensch gibt erstes Kreuz ein
3 Spielzug zug = warten(Figur.KREUZ);
4
5 // Computerzug
6 ersterComputerZug(zug);
7 }
8
9 void ersterComputerZug(Spielzug menschZug) {
10     Spielzug zug = new Spielzug();
11     zug.figur = Figur.KREIS;
12     zug.reihe = 0;
13     if (menschZug.spalte != 0) {
14         zug.spalte = 1;
15     }
16     ziehen(zug);
17 }
18
19 }

```



Abb. 5.17: 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.

#### 5.5.4 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.

## 5.6 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 5.18).

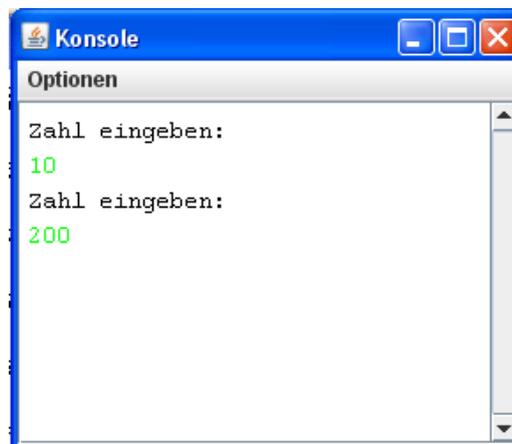


Abb. 5.18: Konsole

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 TicTacToe-Simulators erscheinen in der Konsole.

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

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.

## 5.7 Befehlsfenster

Solist generiert ein Befehlsfenster, über das einzelne Befehle oder selbst definierte Funktionen und Prozeduren bspw. zu Testzwecken auch interaktiv ausgeführt werden können. Sie können das Befehlsfenster über das Menü „Fenster“ durch Anklicken des Eintrags „Befehlsfenster“ öffnen bzw. wieder schließen (siehe Abbildung 5.19).

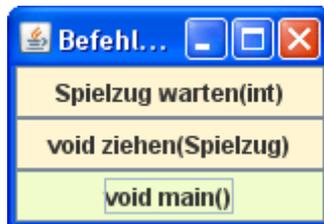


Abb. 5.19: Befehlsfenster

### 5.7.1 Befehlsfenster

Im Befehlsfenster werden die TicTacToe-Befehle (mit grünlichem Hintergrund) sowie die Prozeduren und Funktionen dargestellt, die im aktuellen TicTacToe-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.

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

### 5.7.2 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.

### 5.7.3 Rückgabewerte von Funktionen

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

#### **5.7.4 Anmerkung**

Leider können die beiden TicTacToe-Befehle nicht interaktiv ausgeführt werden. Das Warten auf einen Mensch-Spielzug funktioniert im interaktiven Modus nicht und der setzen-Befehl kann nicht ausgeführt werden, weil er einen Parameter vom Typ Spielzug erwartet, der aber kein Standarddatentyp ist.

## 6 Literatur zum Erlernen der Programmierung

Der TicTacToe-Simulator ist ein Werkzeug, das Programmierern beim praktischen Erlernen der (imperativen) Programmierung hilft. Der Befehlssatz ist minimal und es macht einfach Spaß visualisiert zu bekommen, was ein Programm bewirkt. Der TicTacToe-Simulator ist jedoch kein Lehrbuch. Um mit dem TicTacToe-Simulator programmieren zu lernen, sollten Sie sich ein begleitendes Lehrbuch anschaffen.

Wie bei der TicTacToe-Miniprogrammierwelt 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 TicTacToe-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](http://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>.