NetBeans — Kurzanleitung

Karl Stroetmann

12. Juni 2006

Die Software *NetBeans* ist eine *Java* IDE. Die Abkürzung IDE steht für *integrated development environment*, *NetBeans* ist also eine komplette Entwicklungs-Umgebung für *Java*. Für den Anfänger ist der Gebrauch einer solchen Entwicklungs-Umgebung nicht unbedingt empfehlenswert, da das Erlernen der Bedienung einer solchen Umgebung mit erheblichem Zeitaufwand verbunden ist. Am Anfang ist es besser, diese Zeit in das Erlernen der Programmiersprache *Java* zu investieren. Kein Programmierer kommt allerdings mittelfristig um die Verwendung eines Debuggers herum. Ziel dieser Kurzanleitung ist es daher, Sie mit den wichtigsten Features des Debuggers vertraut zu machen, der in *NetBeans* integriert ist.

1 Start der Entwicklungs-Umgebung

Im Übungsraum ist *NetBeans* unter dem Pfad

```
/opt/netbeans-4.1/bin/netbeans
```

abgelegt. Am einfachsten ist es, wenn Sie am Ende der Datei `~/.bashrc` die folgende Zeile einfügen:

```
export PATH=/opt/netbeans-4.1/bin/:$PATH
```

Dann können Sie die Entwicklungs-Umgebung mit dem Befehl

```
netbeans
```

starten.

2 Laden eines Programmes

Wie in der Einleitung schon erwähnt macht es wenig Sinn, wenn Sie sich schon gleich zu Anfang mit dem in *NetBeans* integrierten Editor auseinander setzen. Es ist einfacher, wenn Sie die *Java*-Programme zunächst weiter mit einem ihnen vertrauten Editor wie `xemacs`, `emacs`, `kate` oder `vi` erstellen. Dann müssen Sie aber wissen, wie Sie ein *Java*-Programm in die Entwicklungs-Umgebung laden können. Ich nehme im folgenden an, dass Sie irgendwo ein Verzeichnis `Java-Examples` haben und dass Sie diesem Verzeichnis für jedes Projekt, dass Sie bearbeiten, einen Unterordner anlegen. Konkret nehmen wir an, dass das Verzeichnis `Java-Examples` eine Unterordner `Calculator` enthält und dass sich in diesem Unterordner die *Java*-Dateien befinden, deren Funktionalität sie testen wollen. Konkret wollen wir annehmen, dass Sie dort die Dateien aus dem Tape-Archive

```
http://www.ba-stuttgart.de/~stroetma/Java/calculator.tar
```

installiert haben. Es handelt sich hierbei um das in der Vorlesung besprochene Programm zur Auswertung arithmetischer Ausdrücke. Das Starten des Debuggers verläuft nun wie folgt:

1. Öffnen Sie eine Shell und starten Sie *NetBeans* mit dem Kommando `netbeans`. Falls das nicht funktioniert, müssen Sie den vollen Pfad angeben:

   ```
   /opt/netbeans-4.1/bin/netbeans
   ```

2. Öffnen Sie das Menü "`File` und wählen Sie dort den Eintrag "`New Project...`". Alternativ können Sie auch die Taste `Ctrl+Shift-N` betätigen.

3. Es erscheint nun ein Fenster, indem Sie unter der Überschrift `Categories` die Kategorie des Projektes festlegen können. Voreingestellt ist hier `General` und das ist für unsere Zwecke gerade richtig.

   Außerdem können Sie unter der Überschrift `Projects` die Art des Projektes festlegen. Hier wählen Sie

   `Java Project with Existing Sources`

   und klicken anschließend auf den Button "`Next>`".

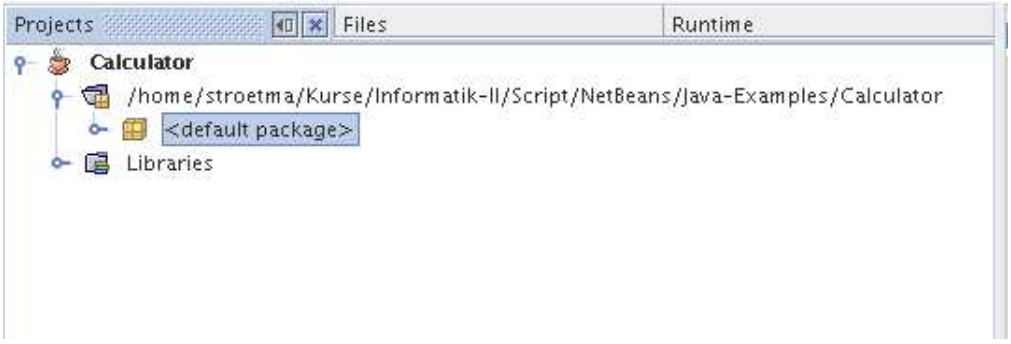
4. Es erscheint ein neues Fenster, in dem Sie zunächst einen Namen für das Projekt festlegen. Wir wählen den Namen "`Calulator`". Gleichzeitig können Sie auf dieser Seite noch der Name eines Verzeichnisses angeben, in dem *NetBeans* Daten zu dem neuen Projekt ablegt. Die Voreinstellung ist, dass der Projekt-Ordner in ihrem Home-Verzeichnis abgelegt wird und den selben Namen hat wir das Projekt. Da es nicht sehr sinnvoll ist, das Home-Verzeichnis mit solchen Projekt-Ordnern zuzumüllen, empfiehlt es sich, in dem Home-Verzeichnis einen Ordner "`NetBeans-Projects`" anzulegen und als Projekt-Ordner dann den Ordner "`~/NetBeans-Projects/Calculator`" anzugeben. Anschließend klicken Sie wieder auf den Button "`Next>`".

5. Nun erscheint ein Fenster, indem Sie das Verzeichnis angeben können, indem sich die *Java*-Dateien befinden. Betätigen Sie hierzu die Schaltfläche `Add Folder` neben dem Fenster mit dem Label `Source Package Folders`. Anschließend geben Sie das Verzeichnis

   "`Java-Examples/Calculator`"

   an. Danach betätigen Sie die Schaltfläche `Finish`.

6. Nun sind die *Java*-Dateien geladen und es erscheint ein Fenster, dessen linke obere Ecke die in Abbildung 6 gezeigte Form hat. Klicken Sie auf den blauen Schalter links neben dem



   Text "`<default package>`", so werden die Namen der *Java*-Dateien angezeigt, die sich in dem Verzeichnis befinden. Die Datei, die die Methode `main`() enthält, ist durch ein kleines grünes Dreieck markiert. Anschließend kann durch einen Doppel-Klick eine dieser Dateien ausgewählt werden. Wir wählen die Datei `Calulator.java` aus. Diese Datei wird nun im rechten Teil des Debuggers angezeigt.

3 Verwendung des Debuggers

Bevor wir den Debugger laufen lassen, sollten wir Halte-Punkte (*break points*) setzen. Wir gehen dazu in die Zeile, in der wir den Haltepunkt setzen wollen. In unserem Fall wählen wir in dem Konstruktor `Calulator` die Zeile mit der ersten `while`-Schleife und drücken

`Ctrl-F8`

Die Zeile, in der sich der Cursor befindet, wird dann rot unterlegt um anzuzeigen, dass ein Haltepunkt gesetzt wurde. Eine nochmalige Betätigung der Taste `Ctrl-F8` würde den Haltepunkt

wieder ausschalten. Um nun den Debugger zu starten, betätigen wir die Taste

F5.

Es erscheint ein Fenster, indem wir nach der "*Main Class*" gefragt werden. Das ist die Klasse, deren Methode `main`() wir laufen lassen wollen. Wir wählen dort die Klasse `Calculator` aus. Danach läuft der Debugger los. Da das Programm eine Eingabe (nämlich den arithmetischen Ausdruck, der berechnet werden soll) erwartet, öffnet sich in der linken unteren Ecke der *NetBeans*-Oberfläche ein Textfeld mit dem Label `Input`. Nachdem wir dort den Text

```
1 + 2 * 3 - 4
```

eingegeben haben, schließen wir die Eingabe ab indem wir ein Return eingeben und anschließend die rechts von diesem Text-Feld befindliche Schaltfläche `Close Input` betätigen. Das Programm läuft nun bis zu dem Haltepunkt, den wir gerade gesetzt haben. Wenn wir das Programm jetzt weiterlaufen lassen, dann wollen wir sowohl die lokalen Variablen als auch die Member-Variablen beobachten. Dazu suchen wir zunächst im Editor die Member-Variablen `mTokens`, `mArguments` und `mOperators`. Wir bewegen den Cursor an den Anfang der Variablen und setzen jeweils durch drücken der Taste

```
Ctrl-Shift-F7
```

einen Beobachtungs-Punkt (*watch point*). Um das Fenster, das die Beobachtungs-Punkte anzeigt, sichtbar zu machen, wählen wir in dem Menü `Window` den Eintrag `Debugging`→`Watches` aus. Alternativ können wir auch die Taste

```
Ctrl-Shift-2
```

betätigen. Da das Fenster mit den Beobachtungs-Punkten jetzt das Fenster mit den lokalen Variablen verdeckt, schieben wir es über das Fenster mit dem Titel "`Navigator - Calculator`", dass sich mittig am linken Rand der *NetBeans*-Oberfläche befindet. Das Fenster mit den Beobachtungs-Punkten enthält eine Tabelle, in der neben den Namen der Variablen noch die Typen und Werte angegeben sind. Weder die Typen noch die Werte sind für uns interessant, was wir sehen wollen ist eine textuelle Darstellung der Variablen. Dazu klicken wir auf die Schaltfläche über dem Scroll-Bar am rechten Rand des Fensters mit dem Titel "`Watches`". Es erscheint ein Fenster in dem wir `toString()` auswählen und `Type` and `Value` abwählen. Das selbe machen wir in dem Fenster, das die lokalen Variablen anzeigt. Dieses Fenster befindet sich rechts unten in der *NetBeans*-Oberfläche.

Jetzt können wir mit dem eigentlichen Debuggen beginnen. Hierzu können wir die folgenden Tasten verwenden:

1. F7 führt eine Zeile des Programms aus. Falls diese Zeile den Aufruf einer Methode enthält, so springt die Kontrolle zu der Methode.
2. F8 führt ebenfalls eine Zeile des Programms aus. Im Unterschied zu F7 springt F8 aber über den Aufruf einer Methode hinweg, das heißt dass ein Methoden-Aufruf in einem einzigen Schritt durchgeführt wird.
3. Alt-Shift-F7 beendet die Abarbeitung der gerade abgearbeiteten Methode und springt zur aufrufenden Methode zurück.
4. Ctrl-F5 läßt den Debugger bis zum nächsten Haltepunkt laufen.
5. Shift-F5 stoppt den Debugger.

Damit haben wir die wichtigsten zum Debuggen benötigten Funktionalitäten vorgestellt.

4 Der Editor

Um Fehler, die beim Debuggen gefunden werden, schnell korrigieren zu können ist es sinnvoll, sich zumindest einen groben Überblick über den Editor zu verschaffen. Der Editor unterscheidet sich nicht wesentlich von anderen Editoren. Wir stellen die allerwichtigsten Funktionen des Editors vor:

1. Das Abspeichern einer Datei geschieht wie bei den meisten Editoren mit `Ctrl-S`.

2. `Ctrl-F` startet die Suche.

3. `Ctrl-Z` macht die letzte Änderung rückgängig.

4. `Ctrl-Leertaste` startet die *Java*-Vervollständigung. Diese ist ein sehr nützliches Feature, dass wir jetzt näher kennenlernen wollen. Wir gehen dazu an das Ende einer beliebigen Methode und geben dort am Anfang einer neuen Zeile den Text "`BigI`" ein. Anschließend betätigen wir die Taste `Ctrl-Leertaste`. Nun vervollständigt der Editor den Text automatisch zu "`BigInteger`". Wir deklarieren eine Variable `a`, schreiben in der nächsten Zeile den Text "`a.`" und drücken wieder `Ctrl-Leertaste`. Dann erscheint über dem Cursor ein neues Fenster, in dem sämtliche Methoden der Klasse `BigInteger` aufgelistet sind. Mit den Tasten "↓" und "↑" können wir hier die verschiedenen Methoden aktivieren. Es wird die Dokumentation der gerade aktivierten Methode angezeigt. Haben wir die gewünschte Methode gefunden, so können wir Sie mit *Return* auswählen.

   Sollte uns das Fenster mit den Methoden aus irgendeinem Grunde stören, so können wir es durch Betätigen des `Escape`-Taste schließen.

5. `Ctrl-K` und `Ctrl-L` aktiviert das sogenannte *Word Matching*. Nehmen Sie an, dass Sie irgendwo oberhalb der Stelle, an der Sie gerade editieren, das Wort

   "`mDonauDampfschiffFahrtsGesellschaft`"

   im Text stehen haben. Jetzt wollen Sie dieses Wort wieder eingeben. Dann reicht es aus, die ersten paar Buchstaben (beispielsweise "`mDo`") dieses Wortes einzugeben und anschließend die Taste `Ctrl-K` oder `Ctrl-L` zu betätigen. Die Taste `Ctrl-K` sucht dann rückwärts im Text nach einem Wort, dass mit dem Text "`mDo`" anfängt und vervollständigt den bisher eingegebenen Text entsprechend. Analog sucht die Taste `Ctrl-L` vorwärts im Text. Gibt es mehrere Wörter, die mit dem Text "`mDo`" anfangen, so können diese der Reihe nach durch nochmaliges Drücken der Tasten `Ctrl-K` oder `Ctrl-L` ausgewählt werden.

6. Schließlich sind in dem *NetBeans*-Editor eine Reihe Abkürzungen integriert. Wollen Sie beispielsweise den Text

   ```
   public static final String
   ```

   eingeben, so reicht es aus, den Text "`psfs`" einzugeben und anschließend die Leertaste zu betätigen. Dann wird der Text "`psfs`" entsprechend expandiert. Sollten Sie tatsächlich einmal den Text "`psfs `" wörtlich eingeben wollen, so müssen Sie anstelle der Leertaste die Taste `Shift-Leertaste` betätigen.

   Sie können sich jederzeit die Liste der Abkürzungen ansehen, indem Sie die Taste `Alt-H`+`K` betätigen. Dann wird eine PDF-Datei angezeigt. Auf der ersten Seite finden Sie die Keyboard-Shortcuts, auf der zweiten Seite sind die Abkürzungen aufgelistet.