



Einführung in die Programmierungsumgebung Eclipse

Vanessa Petrausch, Claudia Loitsch & Gerhard Jaworek

Projektpartner:



Gefördert durch:



Bundesministerium
für Arbeit und Soziales

aus Mitteln des Ausgleichsfonds
Förderkennzeichen: 01KM141108

Inhaltsverzeichnis

1	EINFÜHRUNG IN ECLIPSE	5
2	VORBEREITUNGEN	6
2.1	ECLIPSE UND JAVA INSTALLIEREN	6
2.2	JAVA ACCESS BRIDGE	6
2.3	ANSAGEN VON PUNKTATIONEN BEI SCREENREADERN	7
2.4	EINSTELLUNG VON ECLIPSE ZUR BRAILLEAUSGABE	8
3	ALLGEMEINER AUFBAU VON ECLIPSE	8
4	EINFÜHRUNG IN DIE PROGRAMMIERUNG MIT ECLIPSE	10
4.1	IMPORTIEREN EINES PROJEKTS	10
4.1.1	<i>Der Package Explorer</i>	10
4.1.2	<i>Der Editor</i>	11
4.1.3	<i>Das Outline</i>	11
4.2	DAS ERSTE EIGENE PROGRAMM	12
5	WEITERE PROGRAMMIERTECHNIKEN	14
5.1	WAHRNEHMUNG VON FEHLER UND WARNUNGEN	14
5.2	PLUG-IN INSTALLATION	15
5.2.1	<i>Installation über den „Install new Software“ Dialog</i>	15
5.2.2	<i>Installation über den Marketplace</i>	16
6	LISTE VON TASTATURKÜRZEL	17
6.1	ALLGEMEINE NAVIGATIONSELEMENTE	17
6.2	DATEIMANGEMENT	17
6.3	NAVIGATION IM EDITOR	17
6.4	PROGRAMMIERUNG	18
6.5	VERSIONSVERWALTUNG SVN	18
7	LITERATURVERZEICHNIS	19

1 Einführung in Eclipse

Eclipse ist eine integrierte Entwicklungsumgebung (kurz IDE Integrated Development Environment) zur Entwicklung von Software. Ursprünglich wurde Eclipse seit 1998 von IBM für die Entwicklung von Java genutzt. Seit 2001 ist es jedoch frei verfügbar und wird seit 2004 unter der Verantwortung der Eclipse Foundation weiterentwickelt. Obwohl Eclipse inzwischen verschiedene Programmiersprachen unterstützt, wird es immer noch bevorzugt von Java-Entwicklern genutzt. So nutzen circa 65% der Java-Entwickler Eclipse als Programmierumgebung. Jedes Jahr im Juni werden neue Versionen der Eclipse IDE herausgegeben. Die verwendete Version innerhalb dieses Dokuments ist Eclipse Neon (Version 4.6.), welche im Juni 2016 veröffentlicht wurde. Eclipse ist plattformunabhängig, sodass es auf jedem Betriebssystem genutzt werden kann.

Seit Eclipse Version 3.0 (2004) stellt Eclipse nur noch den Kern zur Verfügung. Die restlichen Komponenten werden mithilfe von Plug-Ins geladen. Plug-Ins stellen die vielfältige Verwendung von Eclipse sicher. So können z.B. weitere Programmiersprachen wie C, C++, Python oder Fortran bereitgestellt werden, Versionskontrollsysteme wie GIT oder SVN direkt integriert werden, GUI-Anwendungen erstellt oder zur Modellierung (EMF, GMF,...) verwendet werden. Eigene Plugins können ebenfalls erstellt und integriert werden, sodass inzwischen viele Projekte als Eclipse-Erweiterung realisiert werden [1].

Diese Modularisierung bietet den Vorteil, dass nach einer Einarbeitung in die Eclipse IDE viele verschiedene Programme und Tools verwendet werden können ohne sich neu in die Umgebung einarbeiten zu müssen. Das Projekt Cooperate, in welchem diese Schulung entstanden ist, entwickelt, basierend auf dem Plug-In Konzept von Eclipse, einen barrierefreien UML-Editor. Mithilfe dieses Editors können Menschen mit und ohne Sehschädigung einfach und effizient zusammen arbeiten und UML Diagramme modellieren. Mehr Informationen zu Cooperate werden auf der Homepage [2] des Projekts bereitgestellt.

Nachfolgend werden zuerst die Vorbereitungen erklärt, welche benötigt werden, um mit Eclipse arbeiten zu können. Enthalten ist dabei die Installation von Eclipse und Java und unterstützende Einstellungen für Screenreader und Braillezeilen. Danach wird der allgemeine Aufbau von Eclipse mithilfe einer Grafik, welche ebenfalls taktil aufbereitet wurde erklärt, um eine bessere Orientierung innerhalb des Programms zu bieten. Im nächsten Teil (Kapitel 4) werden dann die Grundlagen der Programmierung von Eclipse anhand eines Java-Beispiels erklärt. Dieses Kapitel richtet sich an Personen mit einer Sehbehinderung indem auch die Shortcuts der jeweiligen Aktionen eingeführt werden. In Kapitel 5 werden dann zwei weiter fortgeschrittene Programmier Techniken von Eclipse, die Erkennung von Fehlern und Warnungen und die Plug-In-Installation, erklärt. Diese beiden Techniken werden näher behandelt, da diese häufig genutzt werden und bisher für Personen mit Sehschädigung noch nicht optimal umgesetzt sind. Abschließend werden alle Shortcuts, auf welche innerhalb dieses Dokuments verwiesen werden, und einige mehr übersichtlich in einer Tabelle dargestellt.

2 Vorbereitungen

Dieses Kapitel erklärt, welche Komponenten für die Eclipse IDE Installation benötigt werden und wie diese heruntergeladen und eingerichtet werden. Die aktuellen Versionen zur Installation können entweder selbstständig über die bereitgestellten Web-Links heruntergeladen werden oder mithilfe der bereitgestellten Dateien, welche im Ordner Installationsdateien liegen, installiert werden. Abschließend wird beschrieben, wie bei verschiedenen Screenreadern das Ansagen von Punktationen eingestellt werden kann, da dies für die Programmierung wichtig ist. Nachfolgend werden Benennungen von Menüpunkten, Shortcuts oder Dateipfade immer in der Schriftart Consolas gesetzt, um diese von beschreibendem Text zu unterscheiden.

2.1 Eclipse und Java installieren

1. Java

- 1.1. JRE, besser JDK 8 herunterladen von der Webseite: [Download JDK8 von Oracle](#).
- 1.2. Mithilfe des Wizard Java installieren.
- 1.3. Für das Cooperate-Plugin ist JDK8 notwendig, sonst können auch ältere Versionen genutzt werden.
- 1.4. Falls eine Java-Version bereits installiert ist, sollte diese gegebenenfalls auf Version 8 aktualisiert werden.

2. Eclipse

- 2.1. Download von Eclipse Neon for Java Developers (Windows, Linux, MacOS): [Link zur Downloadseite](#)
- 2.2. Entpacken des zip-Ordners an einem beliebigem Ort
- 2.3. Es ist keine weitere Installation von Eclipse notwendig
- 2.4. Starten der Eclipse.exe

3. Mögliche Probleme

- 3.1. Sollte Eclipse, unter Windows, eine Fehlermeldung beim Starten anzeigen, dass Java nicht gefunden wird, muss man eine Pfadvariable zum System hinzufügen. Dies sollte bei einer Installation von Java automatisch geschehen. Falls nicht muss unter: „Systemsteuerung > System > Erweitert > Umgebungsvariablen“ bei der Variable „Path“ der Systemvariablen der Pfad zur Java-Installation gesetzt werden. Dazu bei der Variable auf „Bearbeiten“ wählen und mit einem Semikolon getrennt den Ort des „bin“- Ordners unter Java hinzufügen, z.B. „C:\Programme\jdk1.8.0\jre\bin“.

2.2 Java Access Bridge

Um Java-Programme barrierefrei programmieren und nutzen zu können, müssen die verschiedenen Komponenten von einem Screenreader oder einer Braille-Zeile erkannt werden. Eine Schnittstelle, um dies zu gewährleisten, bietet die Java Accessibility API (JAAPI). Die Java Access Bridge ist die Brücke zwischen den Unterstützungstechnologien und der API [3]. Seit der Java-Version 7 ist die Access Bridge bereits mitinstalliert, muss jedoch noch aktiviert werden. Es wird davon ausgegangen, dass der Anwender mindestens die Java-Version 7 (Aktuelle Version ist Java 8) verwendet, weshalb das Installieren unter der Version

6 nicht beschrieben wird. Es gibt mehrere Varianten, die Access Bridge ab Version 7 zu aktivieren. Nachfolgend werden drei Varianten zur Aktivierung vorgestellt:

1. Mithilfe der Menüführung

1.1. Windows 7 und 10

Start > Systemsteuerung > Center für erleichterte Bedienung > Computer ohne einen Bildschirm verwenden > Java Access Bridge aktivieren (ganz unten auf der Seite)

1.2. Windows 8.1

Zum Desktop wechseln und Rechtsklick auf Startmenü/Kontextmenü öffnen: Systemsteuerung > Center für erleichterte Bedienung > Computer ohne einen Bildschirm verwenden > Java Access Bridge aktivieren (ganz unten auf der Seite)

2. Kurzversion der Menüführung (funktioniert nicht bei Windows 8)

Windowstaste+U > Computer ohne einen Bildschirm verwenden > Java Access Bridge aktivieren

3. DOS-Kommando:

- %JRE_HOME% ist ein Platzhalter für den Pfad zur Java-Installation
- %JRE_HOME%\bin\jabswitch –enable (einschalten)
- %JRE_HOME%\bin\jabswitch –disable (ausschalten)

2.3 Ansagen von Punktationen bei Screenreadern

Viele Screenreader sagen Interpunktationen standardmäßig nicht an. Für die Programmierung ist dies jedoch sinnvoll. Nachfolgend wird beschrieben wie die Einstellungen für verschiedene Screenreader geändert werden können [1].

- JAWS
 - Optionen > Stimmen > Stimmeneinstellungen
 - Satzzeichen: Alle
 - Button Übernehmen
 - Mit OK bestätigen
- NVDA
 - NVDA Menü (Einfg+N) > Einstellungen > Stimmeneinstellungen
 - Zeichen- und Symbolebene: Alle
 - Mit OK bestätigen
- Orca
 - Terminal öffnen: orca -s eingeben
 - Speech > Punctuation level: All auswählen
 - Mit OK bestätigen

2.4 Einstellung von Eclipse zur Brailleanzeige

Vor allem für die Arbeit mit Screenreadern wie z.B. NVDA und einer Braillezeile ist es vorteilhaft wenn Einrückungen im Code als Leerzeichen dargestellt werden und nicht als TAB-Zeichen. Diese Einstellung kann innerhalb von Eclipse folgendermaßen geändert werden:

Window > Preferences > General expandieren > Editors expandieren > Text Editors

In diesem Fenster muss die Checkbox `Insert spaces for tabs` selektiert werden, um Tabs durch Leerzeichen zu ersetzen. Oberhalb dieser Checkbox ist ein Eingabefeld, welches die Anzahl Leerzeichen pro Tab angibt. Benannt ist dieses mit `Display tab width`.

3 Allgemeiner Aufbau von Eclipse

In diesem Kapitel wird der allgemeine Aufbau von Eclipse ohne Inhalte erklärt. Um Eclipse auszuführen muss im entpackten Ordner die `eclipse.exe` ausgeführt werden. Beim Starten wird nach dem Workspace gefragt. Der Workspace ist der Ort an dem alle Projekte mit den Daten gespeichert werden. Dieser Ordner kann mithilfe des Auswahlmenüs beliebig gewählt werden. Die Standardauswahl erfolgt für Windows unter „C:\Users\Benutzername\workspace“. Es kann jedoch jeder beliebige Ablageort verwendet werden. Im Dialog gibt es eine Checkbox „Als Default nutzen“. Wird diese gesetzt, wird nicht bei jedem Starten von Eclipse nach dem Workspace gefragt, sondern die Einstellungen übernommen. Der aktuelle Workspace kann zu jedem Zeitpunkt gewechselt werden, auch wenn ein Workspace als Default gesetzt wurde. Mithilfe des Menüs `File > Switch Workspace > Other` wird ein neuer Workspace ausgewählt.

Initial erscheint ein „Welcome“-Bildschirm auf welchem sich Übersichten, Tutorials, Beispiele und vieles mehr befinden. Dieser Bildschirm öffnet sich im Editorbereich und kann mithilfe des „X“ oder dem Menüeintrag `Workbench` geschlossen werden. Unten rechts, oder am Ende der Tab-Reihenfolge, ist eine Checkbox mit der eingestellt werden kann, ob dieser Bildschirm bei jedem Starten von Eclipse erscheinen soll oder nicht. Initial ist die Checkbox gesetzt, sodass bei jedem Starten von Eclipse der Welcome-Bildschirm erscheint.

Nach dem Welcome-Bildschirm wird der Startbildschirm von Eclipse angezeigt. Die Standardeinstellung unter Windows ist in Abbildung 1 abgebildet. Die Abbildung unterteilt den Startbildschirm in 6 Bereiche, welche farblich voneinander abgehoben und nummeriert sind. Verschiedene Bereiche werden in Eclipse als Views bezeichnet. Die Abbildung wurde ebenfalls taktil in A3 Format aufbereitet. Nachfolgend werden alle Bereiche einzeln erklärt. Die Aufzählung entspricht dabei der Nummerierung der Bereiche.

1. Lila Bereich oben über die gesamte Breite: Hier befindet sich das Menü und darunter eine Icon-Leiste, welche individuell gestaltet werden kann, um auf bestimmte Funktionen schneller zugreifen zu können.

Error! Use the Home tab to apply Überschrift 1 to the text that you want to appear here.

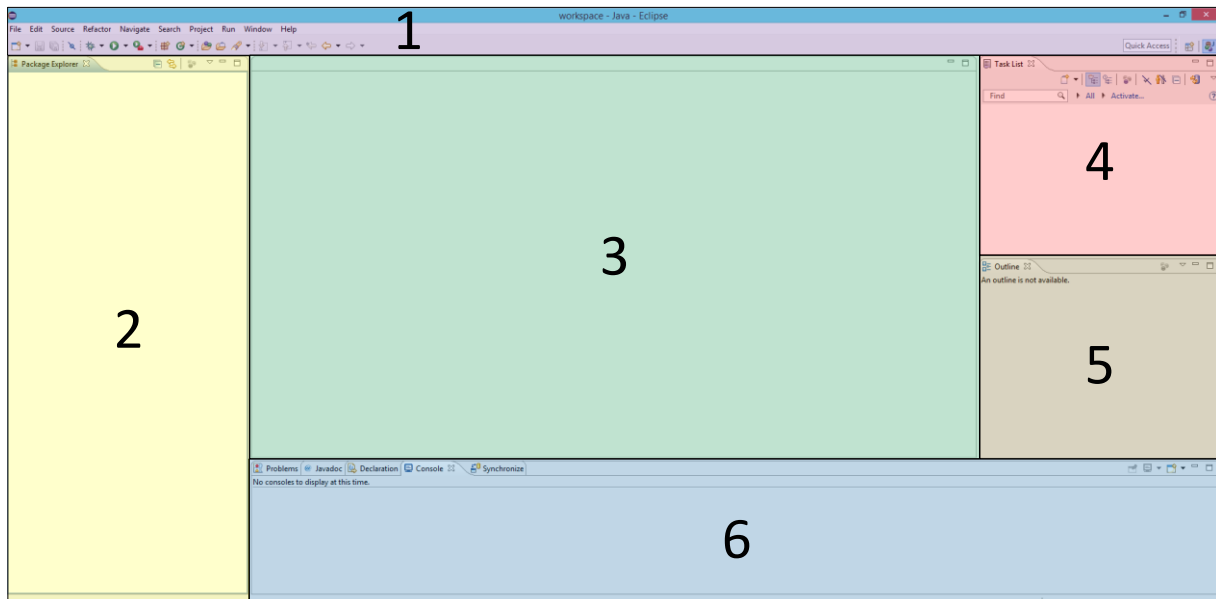


Abbildung 1: Startbildschirm von Eclipse unter Windows

2. Gelber Bereich als Spalte ganz links, gesamte Höhe des Bildschirms: Package Explorer. Hier wird ein Überblick über alle Projekte mit allen enthaltenen Dateien gegeben.
3. Grüner Bereich in der Mitte, Hauptbereich: Editor. Der Editor ist die Hauptansicht für die Quelldateien. Verschiedene Dateien werden nebeneinander in einer Tab-Liste mithilfe ihrer Namen angezeigt.
4. Roter Bereich rechts oben, ca. 1/3 der Höhe: Task List. Die Task List ist eine Aufgabenverwaltung, welche lokale oder global gespeicherte Aufgaben in einem Repository wie Bugzilla oder Trac anzeigt. Im weiteren Verlauf wird diese View nicht mehr näher erläutert, da sie nicht relevant für die folgenden Beispiele ist.
5. Brauner Bereich, rechts unter der Task List, ca. 1/3 der Höhe: Outline. Es bietet eine Übersicht über weitere Informationen. Bei einer Klasse z.B. die enthaltenen Attribute und Methoden, zu welchen man mit Enter zu der Definition des Codes im Editor springen kann.
6. Blauer Bereich unten gesamte Breite vom Package Explorer an: Zusatzinformationen. Hier werden verschiedenste Informationen, wie z.B. Konsolenausgaben, Kompilierungsfehler, Ausgabe des Programms, Javadoc, etc. angegeben [1].

Per Menüeintrag **Window > Show View** können weitere Views angezeigt werden. Mithilfe des Shortcuts **ALT+SHIFT+Q** und einem weiteren spezifischen Buchstaben danach, können viele verschiedenen Standardviews direkt ausgewählt werden, ohne dass diese vorher explizit geöffnet werden müssen. So kann beispielsweise mit **ALT+SHIFT+Q,O** das Outline angezeigt werden. Mittels **Strg+F7** kann durch die verschiedenen Views navigiert werden, Mittels **Strg+Shift+F7** kann rückwärts navigiert werden. Die Steuerungstaste muss dabei gedrückt gehalten werden, um zwischen den Views zu rotieren. Views können frei angeordnet werden, z.B. den Editor ganz links oder als Vollbild ohne andere Views. Die Anordnung und der Umfang aller angezeigten Views wird Perspektive genannt. Persönlich konfigurierte Perspektiven können gespeichert werden. Dies erfolgt über den Menüeintrag **Window > Perspective > Save Perspective as...** Im daraufhin geöffneten Wizzard kann ein eigener Name vergeben werden oder bereits vordefinierte Perspektiven überschrieben werden. Es gibt einige vordefinierte Perspektiven, z.B. für Debugging, ein Repository für GIT oder zum Testen.

Oben rechts über der Task List wird die aktuell ausgewählte Perspektive als Icon angezeigt und andere können direkt ausgewählt werden. Mehr Details zu den Inhalten der einzelnen Views werden im nächsten Kapitel anhand eines konkreten Beispiels erklärt.

4 Einführung in die Programmierung mit Eclipse

Dieses Kapitel richtet sich an Personen mit einer Seheinschränkung. Die ersten Programmierschritte in Eclipse werden jeweils unter Verwendung der Maus und/oder der Tastatur erklärt. Zuerst wird dabei ein existierendes Programm (der Anleitung beigelegt) importiert und anhand dessen die Bedeutungen der wichtigsten Views erklärt. Danach wird die Erstellung eines ersten eigenen Programms beschrieben. Um eine effizientere Arbeitsweise zu ermöglichen werden, sofern vorhanden, Shortcuts für Aktionen genannt, welche den Arbeitsfluss beim späteren Arbeiten erleichtern.

4.1 Importieren eines Projekts

1. Import Dialog öffnen
 - a. Per Maus: Mauscursor in den Package Explorer (Bereich 2) setzen und über das Kontextmenü (Rechtsklick oder Shift+ F10) die Funktion `Import` auswählen
 - b. Per Tastatur: Über das Menü `File` die Funktion `Import` auswählen.
2. Im sich öffnenden Dialog den Ordner `General > Existing Projects into Workspace` auswählen und mit `Next` fortfahren
3. Über die Funktion `Browse` zu dem gespeicherten Ordner `Projekt` navigieren und diesen auswählen (nicht die darin enthaltenen Ordner).
4. Mit `Finish` den Dialog beenden
5. Das Projekt mit allen enthaltenen Dateien erscheint nun im Package Explorer (Bereich 2). Um die Dateien ansehen zu können, muss der Projektordner expandiert werden.

4.1.1 Der Package Explorer

Im Package Explorer (Bereich 2) erscheint das Projekt. Die Dateien des Projekts sind in einer Baumstruktur angeordnet. Das importierte Beispiel weist mit einem vollständig expandierten Dateibaum, bis auf die Java Bibliotheken, folgende Struktur auf:

- Projekt
 - src
 - paket
 - AndereKlasse.java
 - AndereKlasse
 - `Main(String[]):void`
 - MeineErsteKlasse.java
 - MeineErsteKlasse
 - `main(String[]):void`
 - `multiply(int, int):int`
 - JRE System Library [JavaSE-1.8]

Error! Use the Home tab to apply Überschrift 1 to the text that you want to appear here.

Im Ordner Source (src) liegen alle relevanten Projektdateien. In diesem Fall gibt es nur ein Package mit dem Namen paket. Darin enthalten sind alphabetisch angeordnet alle enthaltenen Java-Klassen, welche dem Paket zugeordnet sind. Unter anderem ist eine Klasse mit dem Namen `MeineErsteKlasse.java` enthalten. Per Konvention werden Paketnamen klein und Klassennamen groß geschrieben. Innerhalb der Java-Klasse sind die definierten Methoden aufgelistet. Im Beispiel sind dies die Methoden `multiply` und `main`. Die Typen der übergebenen Parameter werden in Klammern gefolgt von einem Doppelpunkt und dem Rückgabotyp angegeben. Die Methoden werden alphabetisch angeordnet. Der Eintrag `JRE System Library` zeigt in eckigen Klammern die verwendete Java-Version an. In unserem Beispiel ist Java 8 eingebunden.

4.1.2 Der Editor

Um die Java Klassen `AndereKlasse.java` oder `MeineErsteKlasse.java` im Editor (Bereich 3) zu öffnen führen sie einen Doppelklick mit der Maus aus oder drücken Enter auf der Tastatur. Der Sourcecode von `MeineErsteKlasse.java` ist nachfolgend abgebildet:

```
package paket;

public class MeineErsteKlasse {

    public static int multiply(int a, int b){
        return a*b;
    }

    public static void main(String[] args) {
        System.out.print(multiply(3,6));
    }
}
```

Die Standardsprache der benutzten Eclipseversion ist Java, sodass Syntax-Highlighting von Keywords in Java enthalten ist. Keywords werden als standardmäßig violett hinterlegt. Sind mehrere Klassen in einem Editor geöffnet, wird dies durch eine Tab-Ansicht dargestellt. Der aktive Tab ist dabei in einer anderen Farbe als die inaktiven Tabs hinterlegt.

Hinweis für die nicht-visuelle Benutzung: die farblichen Markierungen sind mit dem Screenreader nicht zugänglich.

Mithilfe der Tastenkombination `Strg+F6` kann zwischen den geöffneten Tabs/Dateien im Editor gewechselt werden. Die Tastenkombination `Strg+E` zeigt eine Liste aller geöffneten Dateien im Editor an. Mithilfe der Pfeiltasten kann durch diese navigiert werden und mittels der Taste `Enter` eine Datei zur Anzeige ausgewählt werden.

4.1.3 Das Outline

Das Outline kann mit `Alt+Shift+Q` und dann `O` oder mithilfe des Menüs `Window > Show View > Outline` geöffnet werden. Mittels `Strg+F7` (`Strg` gedrückt halten und mit `F7` rotieren) kann der Fokus auf das Outline gesetzt werden, sofern die View bereits geöffnet ist.

Im Outline sieht man alle enthaltenen Klassen und Methoden mit Parametern und Rückgabewert der geöffneten Datei im Editor. In unserem Beispiel ist im Package `paket` die Klasse `MeineErsteKlasse` geöffnet. Diese Klasse enthält bisher nur zwei Methoden: `multiply (int, int): int` und `main(String []): void`. Die Methoden werden dabei in der Reihenfolge ihrer Definition im Sourcecode angezeigt. Da die Methode `multiply` vor der `Main`-Methode definiert wird, wird diese auch zuerst aufgelistet. Es gibt im Menü des

Error! Use the Home tab to apply Überschrift 1 to the text that you want to appear here.

Outlines jedoch die Option die Methoden alphabetisch anzuordnen. Das Outline sieht standardmäßig für die Klasse `MeineErsteKlasse` folgendermaßen aus:

- paket
 - MeineErsteKlasse
 - `multiply(int, int):int`
 - `main(String[]):void`

4.1.3.1 Hinweis für die visuelle Benutzung:

Wird eine Methode markiert oder ausgewählt, werden im Editorfenster die Bereiche der Deklaration und alle Aufrufe markiert. Im Beispiel aus Abbildung 2 ist die Methode `multiply` markiert. Die Deklaration ist dabei durch einen blauen Balken am linken Rand im Editorfenster markiert, wie in Abbildung 2 zu sehen ist. Der Name der Methode wird außerdem blau hinterlegt. Die Aufrufe der Methode werden grau hinterlegt. Betätigt man die Enter-Taste im Outline wird der Fokus auf die Deklaration der Methode im Editor gesetzt. Der Cursor wird dabei direkt hinter dem Methodennamen gesetzt. Es ist zu beachten, dass der Methodennamen dabei markiert ist und es daher leicht passieren kann, dass der Methodennamen gelöscht wird. Die farbliche Markierung und die Veränderung der Cursorposition durch das Drücken der Enter-Taste im Outline sollen helfen, rasch eine bestimmte Methode im Editor zu finden.

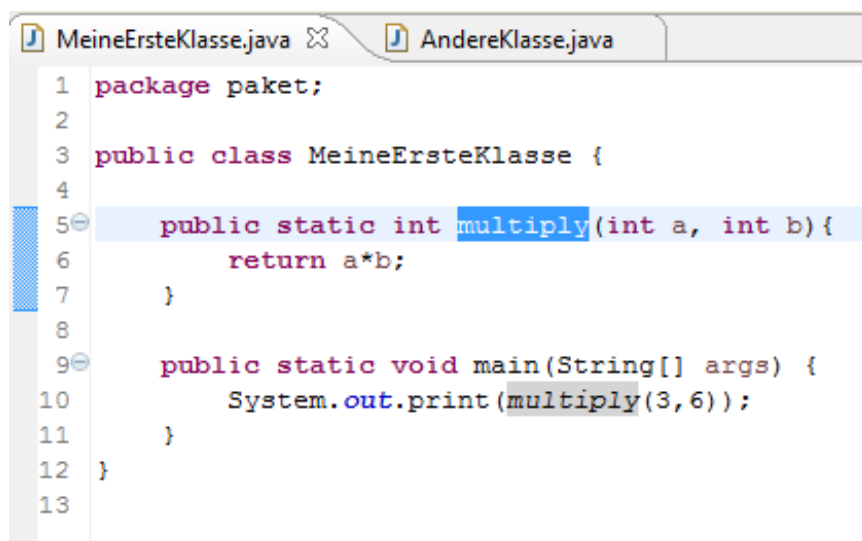


Abbildung 2: Hervorhebung von Methodenaufrufen

4.1.3.2 Hinweis für die nicht-visuelle Benutzung:

Betätigt man die Enter-Taste im Outline wird der Fokus auf die Deklaration der Methode im Editor gesetzt. Der Cursor wird dabei direkt hinter dem Methodennamen gesetzt. Es ist zu beachten, dass der Methodennamen dabei markiert ist und es daher leicht passieren kann, dass der Methodennamen gelöscht wird.

4.2 Das erste eigene Programm

Nachdem die einzelnen Views von Eclipse nun bekannt sind, wird nachfolgend erklärt, wie das erste eigene Programm angelegt und ausgeführt werden kann.

- Ein neues Projekt anlegen: Zuerst wird ein neues Projekt angelegt. Projektnamen können beliebig vergeben werden. Folgende Möglichkeiten gibt es für das Anlegen eines Projektes:

1.1. Per Maus:

- Ein Rechtsklick im Package Explorer (Bereich 2) New > Java Project
- Mithilfe des Icons ganz links in der Icon Leiste (Bereich 1): Dropdown-Menü öffnen (kleiner Pfeil rechts vom Icon): Java Project.
- Mithilfe des Icons ganz links in der Icon Leiste (Bereich 1): Ein Klick auf das gesamte Icon öffnet einen Wizard. Im Wizard General > Project auswählen.

1.2. Per Tastatur:

- Mithilfe des Menüs File > New > Java Project.
- Sofern der Fokus auf dem Package Explorer liegt kann mithilfe des Shortcuts SHIFT+F10 das Kontextmenü öffnen und über New ebenfalls ein neues Java Project auswählen kann.

In allen Fällen öffnet sich der „New Java Project“ Dialog. Zunächst muss ein Name für das Projekt angegeben werden, z.B. „FirstProject“. Mit Finish wird der Dialog beendet.

2. Paket anlegen

Innerhalb des neu angelegten Projektes muss ein Paket angelegt werden. Dies kann identisch zu den fünf Schritten aus Punkt 1 angelegt werden mit dem Unterschied, dass ein Package ausgewählt werden muss: New > Package.

- Namen von Packages fangen mit Kleinbuschstaben an, z.B. „myPackage“.
- Im „New Package“ Dialog wird in unserem Fall automatisch das richtige Projekt im Eingabefeld Source Folder gesetzt, da kein weiteres Projekt existiert. Über die Schaltfläche Browse kann man manuell ein Projektordner auswählen falls notwendig.
- Nachdem der Name des Packages eingegeben wurde, kann der Dialog mit Finish beendet werden.

3. Klasse anlegen

- Als letztes muss eine Klasse angelegt werden. Dies geschieht wieder über die bereits beschriebenen Möglichkeiten aus Punkt 1: New > Class.
- Es öffnet sich der Dialog „New Java Class“ der zahlreiche Einstellmöglichkeiten bietet. Der Source Folder für das Projekt und das Paket sollten automatisch ausgewählt sein. Über die Schaltfläche Browse können diese bei Bedarf geändert werden.
- In dem Feld Name muss der Klassenname eingetragen werden, z.B. „MyClass“. Per Konvention fangen Klassen mit Großbuchstaben an.

Error! Use the Home tab to apply Überschrift 1 to the text that you want to appear here.

- Tipp: Setzen Sie aktivieren die Optionen `public static void main (String[] args)` damit die Methode automatisch als Grundgerüst erzeugt wird. Die restlichen Optionen werden hier nicht weiter betrachtet.
- Beenden Sie den Dialog mit **Finish**. Dadurch wird der Editor mit der neu angelegten Klasse geöffnet. Der Fokus ist dabei in der ersten Zeile an erster Stelle.

4. Eine Ausgabe erzeugen

Die sogenannte Main-Methode `public static void main(String[] args)` ist der Einstiegspunkt in jedem Java-Programm und muss definiert werden, da sonst das Programm nicht ausgeführt werden kann.

4.1. In den Rumpf der Main-Methode (innerhalb der geschweiften Klammern) schreiben wir die folgende Zeile: `System.out.print("Hello World");`

4.2. Mithilfe von **STRG+S** oder mit dem Menü **File > Save** die Datei speichern.

4.3. Das Programm kann durch folgende Optionen ausgeführt werden:

4.3.1. Per Maus

- Mit dem Icon mit dem grünen Kreis mit weißer Play-Taste (weißes Dreieck, welches mit der Spitze nach rechts zeigt)
- Über das Menü **Run > Run**

4.3.2. Per Tastatur

- Mit dem Shortcut **Strg+F11**
- Über das Menü **Run > Run**

4.4. In der Konsolenausgabe (Bereich 6) wird die Ausgabe des Programms angezeigt. Mithilfe eines Mausclicks, **STRG+F7** (**Strg** gedrückt halten und mit **F7** rotieren) oder **ALT+SHIFT+Q,C** kann der Fokus auf den Konsolenbereich gesetzt werden.

5 Weitere Programmiertechniken

In diesem Kapitel werden zwei Programmiertechniken erklärt, welche bisher noch nicht optimal für Personen mit Sehschädigung umgesetzt sind. Als erstes wird erklärt, wie man Fehler und Warnungen erkennen und automatisiert beheben kann und als zweites die Installation von Plug-Ins, welche vor der aktuellen Eclipse-Version nicht möglich war für Personen mit Blindheit.

5.1 Wahrnehmung von Fehler und Warnungen

Fehler und Warnungen können in Eclipse nicht direkt gemeldet werden, wie beispielsweise in Word. Um über alle Fehler und Warnungen benachrichtigt zu werden, muss das aktuelle Dokument gespeichert werden. Visuell werden Fehler mit einem roten Kreis und einem weißen X in der Seitenleiste des Editors angezeigt. Warnungen dagegen werden als Glühbirne mit einem gelben Dreieck mit schwarzen Ausrufezeichen ebenfalls in der Seitenleiste angezeigt. Um Fehler und Warnungen zu beheben gibt es mehrere Möglichkeiten.

1. Im Editor mithilfe von Shortcuts

1.1. Das Dokument mit **Strg+S** speichern. Fehler werden unterringelt dargestellt.

Error! Use the Home tab to apply Überschrift 1 to the text that you want to appear here.

- 1.2. Mit der Tastenkombination Strg+Punkt können alle Fehler und Warnungen vorwärts durchlaufen werden, mit Strg+Komma rückwärts.
 - 1.3. Die Fehler und Warnungen werden beim Durchlaufen automatisch selektiert.
 - 1.4. Mit Strg+1 kann ein Quickfix für den aktuellen Fehler angezeigt werden, mit den Pfeiltasten kann durch diese navigiert werden und mit Enter ein Quickfix angewendet.
2. Die Problems-View
- 2.1. Die Problems-View gibt detaillierte Informationen über alle enthaltenen Fehler und Warnungen an einem Ort aus. Alle Fehler und Warnungen werden dabei nach ihrer Art gruppiert.
 - 2.2. Mithilfe von Strg+F7 (Strg gedrückt halten und mit F7 rotieren) oder Alt+Shift+Q,X kann die View „Problems“ geöffnet werden.
 - 2.3. Mithilfe der Pfeiltasten können die Fehler und Warnungen durchlaufen werden. Bei allen Fehlern und Warnungen werden die Ressource, der Dateipfad und die Zeile angegeben.
 - 2.4. Mithilfe von Enter kann direkt zum ausgewählten Fehler oder Warnung gesprungen werden.
 - 2.5. Mit Strg+1 kann ein Quickfix angezeigt und mit Enter angewendet werden.
- Achtung: Einige Fehler werden in der Problems-View richtig erkannt und Lösungen angezeigt, welche im Quickfix nicht angezeigt werden, wie z.B. das Fehlen eines Semikolons am Ende einer Zeile.

5.2 Plug-In Installation

Vor der Eclipse-Version 4.6, Eclipse Neon, war die Installation von Plug-Ins für Personen mit Sehbehinderung kaum möglich. Viele Felder konnten nicht angesteuert werden oder waren nicht benannt, sodass eine Installation ohne sehende Hilfe schwierig war. Seit der Veröffentlichung von Neon ist die Plug-In Installation jedoch auch für Personen mit Sehbehinderung ohne sehende Hilfe möglich. Zwei Möglichkeiten werden nachfolgend erläutert: Installation von Plug-Ins über den „Install new Software“ Dialog und über den „Marketplace“. Rein funktionell ermöglichen beide Varianten letztlich die Installation neuer Features und Plug-Ins. Der Eclipse Marketplace bietet, wie der Name schon sagt, eine Möglichkeit die zahlreichen Community Erweiterungen zu entdecken und zu installieren. Der Marketplace stellt somit eine zentrale Update Seite für Eclipse-basierte Lösungen dar. Der „Install new Software“ dialog ist die klassische Variante zum Installieren und Aktualisieren von Erweiterungen. Hier muss man, im Gegensatz zum Marketplace, die URL einer Erweiterung kennen um die Installation anzustoßen. Das heißt also, der Marketplace verbessert das Entdecken und Finden von Erweiterungen. Beide Möglichkeiten führen letztlich aber zum selben Ziel. Weiterhin ist zu beachten, dass nicht alle Plug-Ins bei beiden Varianten verfügbar sind. Es lohnt sich demnach immer die andere Variante auszuprobieren, falls keine Ergebnisse verfügbar sind.

5.2.1 Installation über den „Install new Software“ Dialog

1. „Install new Software“ Dialog öffnen mithilfe des Menüs Help > Install New Software...

Error! Use the Home tab to apply Überschrift 1 to the text that you want to appear here.

2. Der Fokus ist in dem Eingabefeld „Work with“, in welchem eine Updateseite ausgewählt werden muss. Es kann eine URL eingegeben werden oder eine vorhandene Updateseite aus dem Dropdown-Menü ausgewählt werden, z.B. --All Available Sites--. Mithilfe des Add... Button können neue Seiten hinzugefügt werden.
3. Als nächstes gibt es ein Eingabefeld für die Filterung von Plug-Ins. Es handelt sich um eine inkrementelle Suche, d.h. beim Eintippen startet schon die Suche. Dies kann dazu führen, dass der Dialog zeitweise nicht mehr bedienbar ist, eine Rückmeldung hierrüber liefert NVDA nicht. Die Suche kann ebenfalls durch Betätigung der Enter-Taste gestartet werden.
4. Die Darstellung, ob die Suche beendet ist, ist leider immer noch nicht optimal. Bei NVDA ist es der Fall, dass der Fokus automatisch auf das erste expandierte Suchergebnis springt, wenn Ergebnisse vorhanden sind. Die Ergebnisse werden als Baumansicht dargestellt. Alternativ gelangt man mithilfe von TAB in die Baumansicht der Ergebnisse.
5. Mittels TAB können zusätzliche Optionen ausgewählt werden.
6. Mit Next fortfahren
7. Ein Wizzard mit den Installationsdetails wird angezeigt. Mit Next fortfahren.
8. Die Lizenzvereinbarung muss akzeptiert werden, um die Installation fortzufahren. Als Defaultwert ist „nicht akzeptiert“ markiert. Mithilfe der Pfeiltasten kann die akzeptierende Option gewählt werden.
9. Mit Finish den Dialog beenden und die Installation starten.

5.2.2 Installation über den Marketplace

1. Marketplace öffnen mithilfe des Menüs Help > Eclipse Marketplace...
2. Nach dem Laden ist der Fokus im Suchfeld, sodass direkt ein Suchbegriff eingegeben und mit Enter die Suche gestartet werden kann.
3. Die Suche kann nach „Märkten“ oder „Kategorien“ eingeschränkt werden. Dazu einfach mit TAB weiter navigieren um zum kombinierten Eingabefeld „All Marktes“ oder „All Categories“ zu gelangen. Mit den Pfeiltasten die gewünschte Auswahl treffen.
4. Danach mittels TAB weiter zur Schaltfläche GO navigieren um die Suche zu starten. Die Liste der Suchergebnisse erreicht man mit TAB. Das erste Suchergebnis wird mit „Feld“ angegeben. Bei weiterer Navigation mit Tab werden die Details eines Suchergebnisses angegeben. Alle Suchergebnisse enthalten folgende Informationen
 - 4.1. Einen Titel, welcher jedoch nicht von NVDA vorgelesen wird.
 - 4.2. Einen Beschreibungstext. Dies ist das erste Element, welches von NVDA fokussiert wird.
 - 4.3. Der Entwickler der Erweiterung
 - 4.4. Schlüsselwörter
 - 4.5. Ein Favoriten-Button
 - 4.6. Ein Share-Button
 - 4.7. Die Größe der Installation

Error! Use the Home tab to apply Überschrift 1 to the text that you want to appear here.

4.8. Ein Installier-Button: Install

5. Mittels des Install Buttons startet die Installation des gewählten Plug-Ins direkt.
6. Sollte ein Fehler auftreten öffnet sich ein Fenster, in welchem mehrere Optionen zur Behebung ausgewählt werden können.

6 Liste von Tastaturkürzel

Die folgenden Tabellen bieten eine Übersicht über verschiedene Shortcuts (Tastaturkürzel). In der ersten Spalte findet sich die Beschreibung und in der zweiten Spalte das Tastaturkürzel. Es sind nicht alle Shortcuts enthalten, welche Eclipse zur Verfügung stellt, sondern nur die wichtigsten, um Eclipse ohne Maus bedienen zu können und einige darüber hinaus gehende nützliche Shortcuts um die alltäglichen Programmierung effizient gestalten zu können [4], [5]. Die Shortcuts sind thematisch geordnet, z.B. Shortcuts für „Allgemeine Navigationselemente“ oder „Programmierung“.

6.1 Allgemeine Navigationselemente

Liste aller aktiven Shortcuts in Eclipse anzeigen	Strg + Shift + L
Wechsel zwischen geöffneten Tabs/Dateien im Editor	Strg + F6
Offene Sichten (Views) vorwärts durchlaufen	Strg + F7
Offene Sichten (Views) rückwärts durchlaufen	Strg + Shift + F7
Offene Perspektiven wechseln	Strg + F8
View öffnen (auch nicht sichtbare)	Alt + Shift + Q
Schließen geöffneter Dialoge	Escape
Fokus auf das Menü setzen	F10
Zeigt Kontextmenü (rechter Mausklick)	Shift + F10

6.2 Dateimangement

Öffnet einen Wizzard um neue Dateien zu erstellen	Strg + N
Öffnet das Kontextmenü um eine neue Datei zu erstellen	Alt+ Shift + N
Speichert die aktuelle Datei	Strg + S
Speichert alle Dateien	Strg + Shift + S
Schließt die aktuell aktive Datei	Strg + W
Schließt alle geöffneten Dateien im Editor	Strg + Shift + W
Öffnet die Dateieigenschaften	Alt + Enter

6.3 Navigation im Editor

Setze den Fokus auf den Editor	F12
Liste aller geöffneten Dateien im Editor anzeigen, Navigation mit Pfeiltasten, Datei auswählen mit Enter	Strg + E
Maximieren des Editors (oder andere aktive View), erneute Betätigung minimiert das Fenster wieder	Strg + M
Öffnet das Editor-Fenster-Options-Menü	Alt + Minus
Zeigt Anzeigemenü von verfügbaren Features für den linken, vertikalen Bereich; Breakpoints, Bookmarks, Zeilennummern, etc.)	Strg + F10
Springt zur nächsten Methode	Strg + Shift + Pfeil hoch/runter

Error! Use the Home tab to apply Überschrift 1 to the text that you want to appear here.

Springt zur eingegeben Zeilennummer im sich öffnenden Fenster. Um Zeilennummern ein-/auszublenden Strg + F10 und „Show Line Numbers“ wählen	Strg + L
Springt zur zuletzt editierten Position	Strg + Q
Springt zur öffnenden/schließenden Klammer einer selektierten Klammer (Klammer ist markiert oder Cursor ist direkt davor oder dahinter positioniert)	Strg + Shift + P
Expandiert oder minimiert aktuelle Methode	Strg + NumPad Plus/Minus

6.4 Programmierung

Zeigt Code Outline an	Strg + O
Öffnet Informationen zu Klassen, Methoden oder Variablen als ToOLTIPtext	F2
Öffnet Deklaration: Springt zur Deklaration der selektierten Klasse, Methode oder Parameter	F3
Öffnet das Typhierarchie Fenster des selektierten Items	F4
Fehler vorwärts durchlaufen	Strg + Punkt
Fehler rückwärts durchlaufen	Strg + Komma
Quick-Fix: Fehlerbehebung aufrufen, Pfeiltasten für Navigation	Strg + 1
Nach Typ suchen (Klasse, Interface, ...)	Strg + Shift + T
Umbenennung des selektierten Elements und aller Referenzen	Alt + Shift + R
Speichert und führt Applikation aus (Run)	Strg + F11
Debug	F11
Im Debugger: In Funktion reingehen	F5
Im Debugger: Nächster Schritt (Zeile für Zeile)	F6
Im Debugger: Aus Funktion rausgehen	F7
Im Debugger: Zum nächsten Haltepunkt springen	F8
Öffnet den Inhaltsassistenten, z.B. zeigt verfügbare Methoden oder Feldnamen	Strg + Leertaste

6.5 Versionsverwaltung SVN

Mit Repository synchronisieren	Strg + Alt + S
Eigenschaften anzeigen	Strg + Alt + T
Update	Strg + Alt + U
Update zu Revision	Strg + Alt + D
Commit	Strg + Alt + C
Merge	Strg + Alt + E
Zu svn:ignore hinzufügen	Strg + Alt + I

7 Literaturverzeichnis

- [1] Eclipse Foundation, „Eclipse IDE User Guide,“ [Online]. Available:
<http://help.eclipse.org/neon/index.jsp?nav=%2F0>. [Zugriff am 15 September 2016].
- [2] V. Petrausch, „Cooperate Projekthomepage,“ [Online]. Available:
<http://www.cooperate-project.de>. [Zugriff am 15 September 2016].
- [3] Oracle, „Java SE Documentation: Java Accessibility Guide,“ [Online]. Available:
<http://docs.oracle.com/javase/7/docs/technotes/guides/access/index.html>. [Zugriff am 15 September 2016].
- [4] H. Folkerts, „Blog von Heiko Folkerts,“ [Online]. Available:
<http://www.hfolkerts.de/2011/05/10/plant-uml-hilft-beim-design/>. [Zugriff am 2016].
- [5] „ShortcutWorld.com project,“ [Online]. Available:
<https://www.shortcutworld.com/en/win/Eclipse.html>. [Zugriff am 15 September 2016].