2021

PROSEMINAR
SOFTWAREENTWICKLUNG IN DER WISSENSCHAFT

WJATCHESLAW SANDER

Unter Betreuung von: Dr. Hermann-J. Lenhart

Inhaltsverzeichnis

1. Einleitung

Was ist Eclipse?

Eclipse ist eine Open-Source-Software zur Entwicklung verschiedener Projekte. Es ist eine integrierte Entwicklungsumgebung für Java und weiteren Programmiersprachen, wie Python. Eclipse besteht aus einem umfangreichen Texteditor mit einer großen Anzahl an vorinstallierten Werkzeugen und Funktionen, der noch nach den eigenen Bedürfnissen angepasst und erweitert werden kann. In der Standardversion unterstützt Eclipse nur die Programmiersprache Java, jedoch können weitere Programmiersprachen sowie Werkzeuge zur Projektverwaltung durch das Herunterladen über den integrierten Marktplatz hinzugefügt werden. Dadurch ermöglicht Eclipse die Erstellung vielseitiger Projekte, die die Stärken verschiedener Programmiersprachen gleichzeitig nutzen kann.

2. GUI-Erläuterung

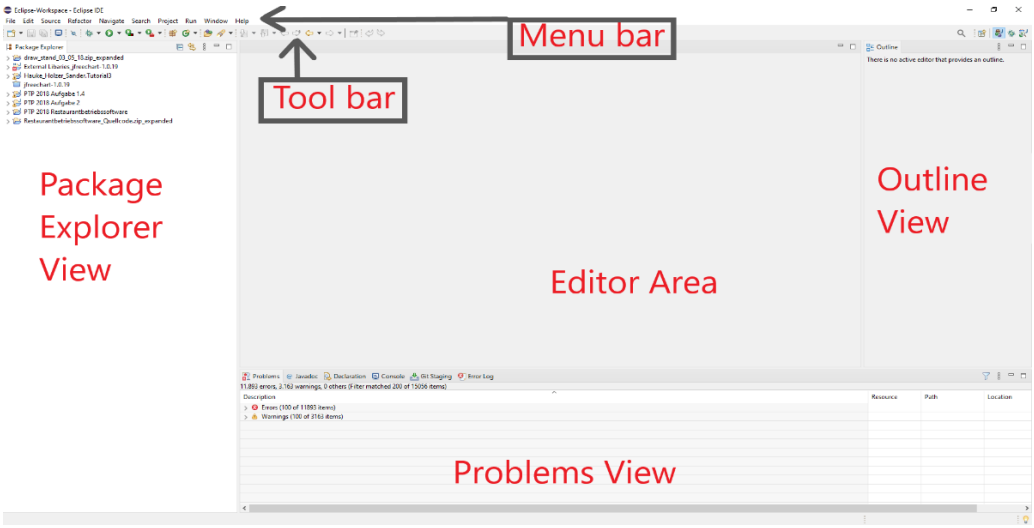


Abbildung 1: Programmfenster von Eclipse; Ansicht nach erstmaligen Ausführen

Das Programmfenster von Eclipse (Abb.1) besteht aus vier Arten von Kernelementen, nämlich der Menüleiste für grundlegende Funktionen des Programms, einer Werkzeugleiste zum Bearbeiten des Projektes, dem Texteditor zur Bearbeitung des Quellcodes sowie verschiedene Ansichten (Views) die die Verwaltung und den Status des Projektes, aber auch Werkzeuge zur Fehlerbehebung anzeigen können. Im Folgenden wird auf diese Elemente näher eingegangen.

Menu Bar

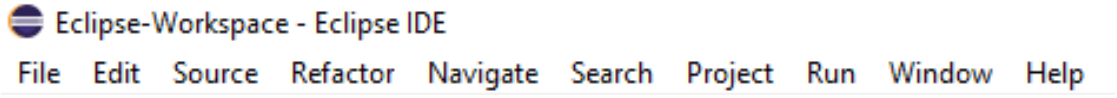


Abbildung 2: Menüleiste von Eclipse

Wie in fast jeder gängigen Software ist auch Eclipse mit einer Menüleiste (Abb. 2) ausgestattet. Unter dieser sind fast alle Funktionen zur Projekterstellung bis hin zum Projektabschluss, sowie die

Verwaltung von Eclipse selbst verfügbar. Zusätzliche Inhalte können die Menüleiste und deren Inhalt verändern.
Unter **File** sind grundlegende Funktionen für das Verwalten von Dateien wie das Öffnen, Speichern und Umbenennen von Dateien, aber auch der Import und Export von Workspaces und ganzen Projekten möglich.
Bei **Edit** sind bekannte Funktionen wie das Kopieren und Einfügen von Textelementen sowie die Erstellungen von Lesezeichen und Aufgaben zu finden.
Im **Source**-Reiter, der für Java angelegt ist, werden hilfreiche Funktionen zum Editieren des Quelltextes angeboten. Hier können beispielsweise häufig verwendete Ausdrücke in Quelltexten wie ein Try-Catch-Block generiert werden.
Refactor, benannt nach der gängigen Praxis in der Softwareentwicklung zum Überarbeiten von Projekten, ermöglicht weiträumige Änderungen wie die Umbenennung von Klassen und Aktualisierung aller ihrer Referenzen im gesamten Projekt.
Unter **Navigate** kann man verwendete Ressourcen aufspüren und sich diese anzeigen lassen.
Wie der Name impliziert, kann in **Search** entweder nach Dateien oder Elementen im Quelltext gesucht werden.
Unter **Project** befinden sich allgemeine Funktionen für den Projektaufbau. Des Weiteren kann eine erste Version eines Javadocs erstellt werden.
Run ermöglicht die Ausführung des Programms im normalen oder dem Debugmodus aus. Hier befinden sich auch die Debugeinstellungen, wie das Erstellen von Breakpoints.
Im Reiter **Window** können neue Views oder eine Perspective, also eine vorgefertigte Einstellung der zu anzeigenden Views, gefunden werden. Hier befinden sich auch die Grundeinstellungen von Eclipse.
Help ist die eingebaute Hilfefunktion in Eclipse. Hier befinden sich hilfreiche Anmerkungen und Vorschläge zur Bearbeitungen von verschiedenen Themen. Die Suche nach Updates sowie die Installation von neuen Plug-Ins über den Eclipse-Marketplace können ebenfalls hier gefunden werden.

Tool Bar

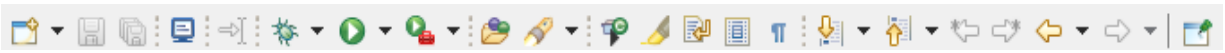


Abbildung 3: Typische Werkzeugleiste, Funktionen wie Erstellung neuer Projekte, Klassen und Packages, aber auch Programmausführung, Suche und Navigation sind hier enthalten.

Die Werkzeugleiste (Abb. 3) beinhaltet Funktionen, die auch in der Menüleiste zu finden sind. Im Prinzip erlaubt diese Leiste lediglich schnelleren Zugriff auf die am häufigsten genutzten Funktionen, die kategorisch nicht zusammenhängend sein müssen. Die Werkzeugleiste kann über den Perspective-Einstellungen im Reiter Window angepasst werden.

View

Views sind graphische Repräsentationen der Metadaten des aktuell zu bearbeitenden Projektes. Sie zeigen kategorisch umfassende, relevante Information über das Projekt wie Fehlermeldungen, Ausgaben der Konsole, Projektdateien oder die Javadoc ein. Views können, wie in Abb. 1 in der Eclipse GUI, an verschiedene Stellen verschoben werden oder in View-Ordnern (Abb. 4) zusammengefasst werden. Welche Views angezeigt werden, kann über den Reiter Window (Abb. 2) angepasst werden. Zusätzlich wechselt Eclipse bei einigen Funktionen, wie dem Debug-Modus, oder der Auswahl einer Voreinstellung des entsprechenden Werkzeugs zu relevanten Views.

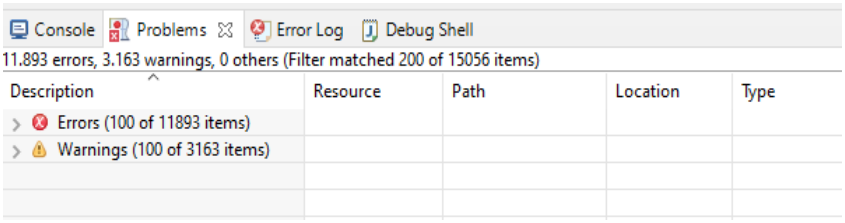


Abbildung 4: View-Ordner mit Konsole, Fehlermeldungen und Fehlerprotokoll

Package Explorer View

Einer der wichtigsten Views ist der Package Explorer View, der eine Übersicht über alle Dateien in Form von Projekten und Ordnern im zuvor bestimmten Workspace liefert. Projekte beinhalten in der Regel den erstellten Quellcode sowie die verwendeten Ressourcen (bspw. Bilder) und externe Bibliotheken (bspw. JFreeChart). Dabei sind die Projekte in einer hierarchischen Struktur abgebildet, bei denen der Quellcode mit den verwendeten Ressourcen gleichgestellt wird. Das Importieren der Ressourcen erlaubt es dem Quellcode auf diese zuzugreifen und sollten sich innerhalb des Projektordners befinden. In Abb. 5 wurde der Quellcode im Ordner „src“ noch in Paketen weiteruntergeordnet, um seine Bestandteile in seiner Funktionalität kenntlich zu machen.

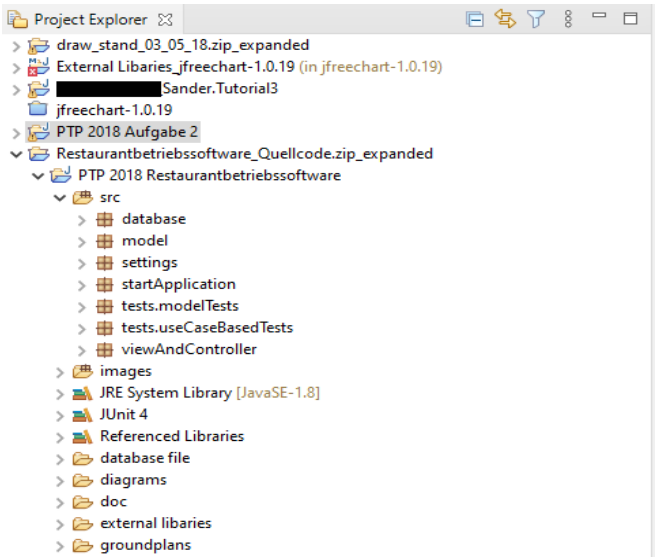


Abbildung 5: Beispielhafter Package View Explorer

3. Umgang mit Eclipse

3.1 Funktionen im Quellcode

Eclipse bietet bei der Bearbeitung von Quelltexten eine Reihe von nützlichen Funktionen an, die ihre Erstellung oder Wartung erleichtern kann, oder sich zeitsparend auswirkt.

Content Assist

Der Content Assist hilft dabei die Anzahl an Zeichen, die man während des Programmierens eingeben muss, zu reduzieren, in dem er basierend auf den bisher eingegeben Zeichen Vorschläge generiert, um das Eingegebene zu vervollständigen. Dieser wird bei gleichzeitigem Drücken mit der CTRL- und der Space-Taste aufgerufen. In Abb. 6 beispielsweise zeigt der Content Assist nach seinem Aufruf Vorschläge zur Vervollständigung einer IF-Anweisung, wobei auf der linken Seite die Optionen aufgelistet werden und auf der Rechten eine Vorschau der markierten Option, die bei ihrer Auswahl konkret eingefügt werden würde.

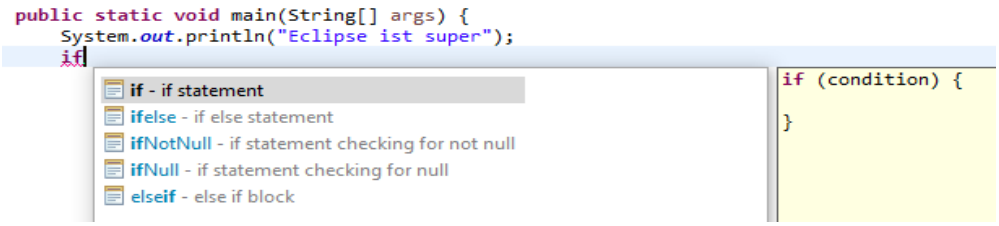


Abbildung 6: Vorschläge zur Vervollständigung einer IF-Anweisung

Der Content Assist vervollständigt nicht nur angefangene Ausdrücke, sondern kann auch die Grundbausteine für einen Methodenblock legen. Wenn der Cursor sich in einer leeren Zeile befindet, werden Bausteine für Methoden angezeigt (Abb. 7). Falls man sich in einer vererbten Klasse befindet, werden auch Methoden der Superklasse angezeigt.

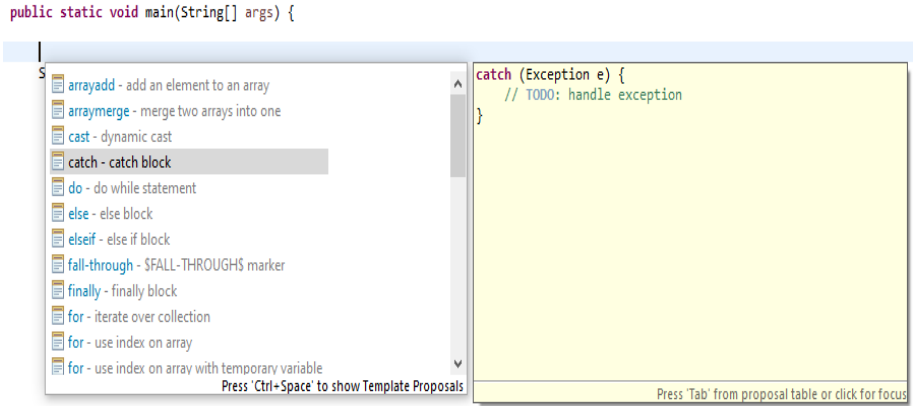


Abbildung 7: Vorschläge zur Generierung von Methodenblöcken

Um eine der Optionen zu wählen, navigiert man mit den Pfeiltasten zur gewünschten Option und bestätigt diese mit der Enter-Taste.

Quick Fix

Während an dem Quelltext gearbeitet wird, überprüft Eclipse den Quelltext auf Fehler und markiert diese mit einer gestrichelten Linie. Dabei steht eine rote Linie für schwere Fehler, wie beispielsweise Syntaxfehler, die eine Programmausführung nicht möglich machen, oder das Fehlen von referenzierten Methoden aus Superklassen (Abb. 8).

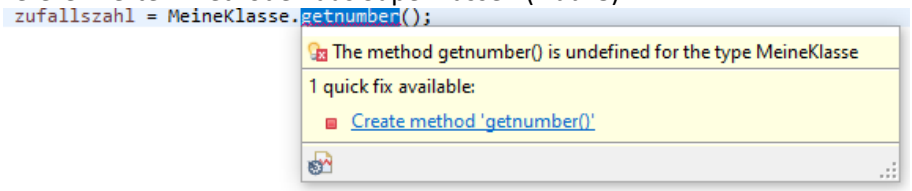


Abbildung 8:Fehlerbehebung durch Generierung einer neuen Methode

Etwas milder sind gelbe Linien, die für Warnungen stehen, die an der Stelle den Autor darauf hinweisen sollen, dass es während der Programmausführung zu Problemen kommen kann. In Abb. 9 wird die Typesafety bemängelt, ein Fehler, der während der Laufzeit auftreten kann, wenn Werte ihren Datentyp ändern.

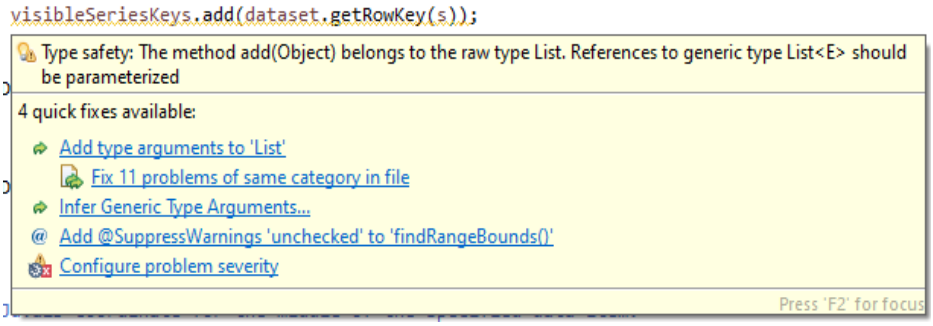


Abbildung 9: Warnungen mit Vorschlägen zur Behebung

Die Vorschläge zur Behebung können auch die Generierung von neuen Methoden, Klassen oder anderen Elementen beinhalten. Des Weiteren überprüft Eclipse auf korrekte Rechtschreibung beispielsweise in Kommentaren, standardmäßig auf Englisch. Der Quick Fix wird durch Markierung des fehlerhaften Textes aufgerufen.

Hover Help

Eine weitere praktische Funktion ist die sogenannte Hover Help. Wird der Mauszeiger über ein Element im Quelltext bewegt, werden zusätzliche Informationen zu diesem angezeigt (Abb. 10). Standardmäßig wird eine Beschreibung aus der Javadoc verwendet, jedoch kann nach Bedarf unter den Einstellungen von Eclipse weniger oder zusätzliche Informationen, wie Werte von Variablen oder Problembeschreibungen, angezeigt werden.

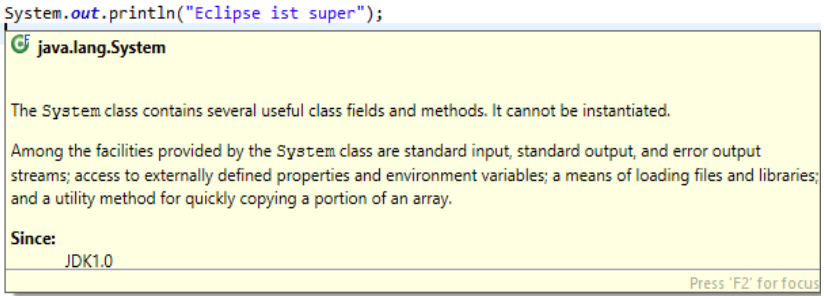


Abbildung 10:: Informationen zur Klasse System

3.2 Debugging

Zum Softwareentwicklungs-prozess gehört auch die Fehlerbeseitigung, also das Debugging, und da liefert Eclipse auch jede Menge Werkzeuge, die das Beheben erleichtern. Um mit dem Debugging zu starten, kann man im Reiter „Run“ „Debug-As“ anwählen und auf „Java-Application“ klicken. Dann wird auf die Debug-Perspective (Abb. 11) gewechselt, die eine Reihe von kategorisch passenden Views öffnet. Der Debug-View oben links in Abb. 11 gibt dabei an, welcher Teil im

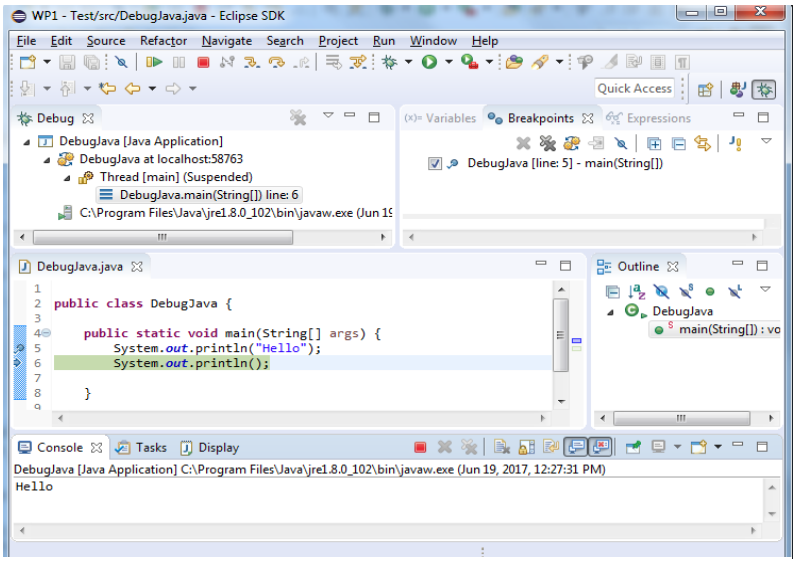
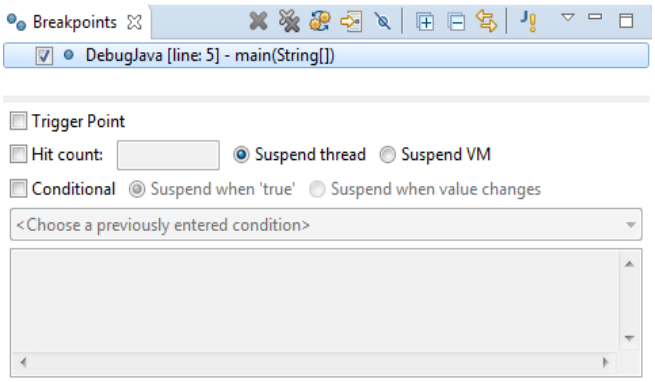


Abbildung 11: Debug-Perspective

Quelltext gerade ausgeführt wird. Im Breakpoint-View rechts daneben sieht man seine manuell eingefügten Breakpoints. Unter dem in Abb. 11 ausgegrauten Variables-View links neben dem Breakpoint-View können auch aktuelle Werte der Variablen zu bestimmten Zeitpunkten eingesehen werden.

Breakpoints

Breakpoints werden manuell im Quelltext auf der Zeilennummer mit der Auswahl „Toggle Breakpoint“ eingefügt. An dieser Stelle wird das Programm im Debug-Modus angehalten und im Breakpoint-View (Abb. 12) werden einige Informationen zum Zustand des Programms ersichtlich. Zunächst sind alle verteilten Breakpoints auf einen Blick zu sehen. Nach Auswahl eines der Breakpoints werden konfigurierbare Optionen für weitere Einsichten angezeigt (Abb. 12). Durch Aktivieren des „Trigger-Points“ werden alle anderen Breakpoints deaktiviert bis der ursprünglich ausgewählte Punkt erreicht wurde. Beim Eingabefeld „Hit count“ kann eine Obergrenze angegeben werden, wie oft der ausgewählte Breakpoint erreicht werden soll, bevor die Ausführung terminiert wird. Unter „Conditional“ kann eine Bedingung deklariert werden, die die Ausführung beenden soll, sobald diese eintritt.



Abbilung 12: Breakpoint-View

3.3 Weitere nützliche Funktionen

Task Management

Durch den Ausdruck „//TODO“ im Quelltext oder durch Rechtsklick bei der Zeilennummer und Wahl der gleichnamigen Funktion kann eine Übersicht generiert werden, an welchen Stellen im Quelltext noch Bearbeitungsbedarf besteht. Im Task-View (Abb. 14) können alle erstellten Tasks eingesehen werden und direkt zu ihnen gesprungen werden.

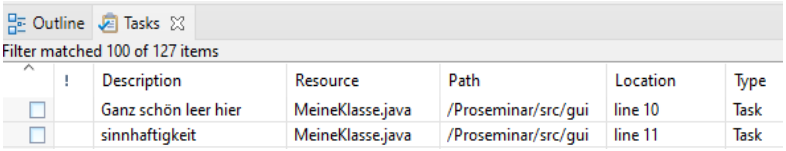


Abbildung 13: Task-View

Unter seinen Eigenschaften (Abb. 14) kann auch die jeweilige Priorität der auszustehenden Task ausgewählt werden. Zudem Zudem kann durch Aktivierung des „Completed“-Kästchens anderen Autoren die Erfüllung der Task mitgeteilt werden, ohne dass diese gelöscht werden muss.

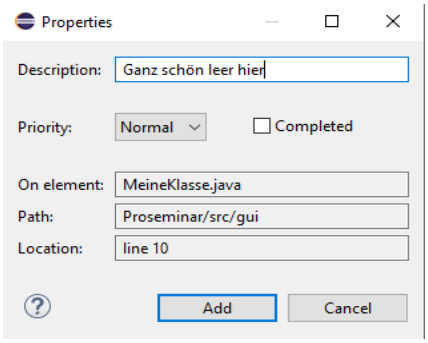


Abbildung 14: Eigenschaften einer Task

Search Menu

Die Suchfunktion erlaubt es Dateien im gesamten Workspace, bestimmten Projekten, ausgewählten Ordnern zu suchen, indem ein Teilstring angegeben wird. Zusätzlich besteht die Möglichkeit Tasks, installierte Plug-Ins oder auch Git-Commits zu durchsuchen. Das Suchfenster (Abb. 15) erlaubt zusätzlich die Einschränkung auf bestimmte Eigenschaften, beispielsweise nach Konstruktoren im Quelltext.

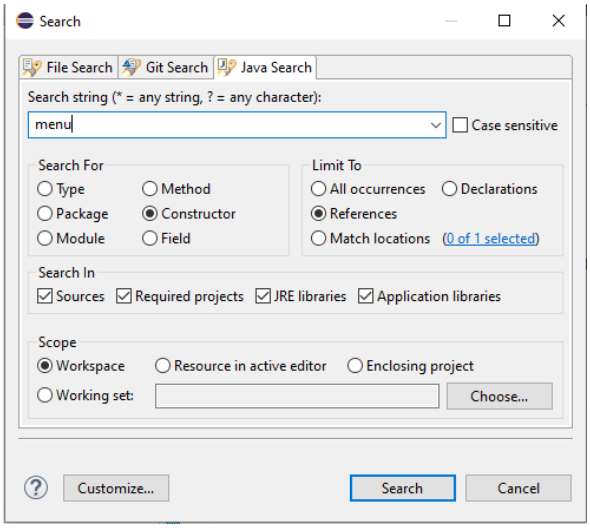


Abbildung 15: Suche nach Menu im Quelltext

Eclipse Marketplace

Die am Anfang erwähnten, herunterladbaren Plug-Ins können im Eclipse-Marketplace gefunden werden. Dieser bietet eine große Auswahl an weiteren Entwicklungsumgebungen, also die Unterstützung von weiteren Programmiersprachen, und Tools, die das Erstellen von Projekten erleichtern können (z.B. Codota) oder die Nutzerfreundlichkeit von Eclipse (z.B. OsgiEquinoxJ) verbessern. Der Marktplatz ist unter dem Menüreiter „Help“ zu finden.

Abbildung 16: Ausschnitt aus dem Eclipse-Marketplace

4. Fazit

Eclipse ist eine sehr umfangreiche Entwicklungsumgebung, die für Neulinge überwältigend erscheinen mag. Die eingebauten Hilfestellungen erwarten größtenteils einen fortgeschrittenen Wissensstand über die Softwareentwicklung. Deshalb sollte eine Einführung mit Tutorials oder erfahrenen Person erfolgen.
Hat man einen guten Einstieg gefunden, wird man feststellen, dass Eclipse sehr viel zu bieten hat und sich durch seine Erweiterbarkeit und der Eigenkonfiguration der GUI den Bedürfnissen des Nutzers leicht anpassen kann. Auch in der Standardausführung bietet Eclipse eine Menge an Funktionen, die die Entwicklung und Wartung von Programmen erleichtert.
Weitere Vorteile umfassen die Kollaborationsfähigkeit durch Github, wodurch Projekte in größeren Teams bewältigt werden können sowie kostenfreie Updates, die von anderen Nutzern entworfen werden.
Insgesamt ist es leicht zu erkennen, warum Eclipse im Bereich der Softwareentwicklung hoch wertgeschätzt wird und Ich persönlich hatte bisher keine größeren Probleme mit dem Umgang, für die es keine Lösung gab.

5. Quellenverzeichnis

Tutorials:

https://www.tutorialspoint.com/eclipse/eclipse_debugging_program.htm

https://javatutorial.net/java-eclipse-tutorial

https://www.eclipse.org/community/eclipse_newsletter/2017/june/article1.php

https://www.tabnine.com/blog/14-free-plugins-for-eclipse-ide/