RWTH AACHEN

Eine Einführung in

(Eclipse 4.12 – 2019-06)

für Mathematisch- Technische Softwareentwickler (MATSE) / Informatik (IHK)

Autor: Michael Wiedau
Aktualisierung: Alexandra Espe, Marcel Nellesen, Mario Grunwald, Tobias Nelles
Version vom 30.08.2019

Inhaltsverzeichnis

Den Workspace wählen

Wenn Sie Eclipse starten, werden Sie zunächst aufgefordert, einen „Workspace“ zu wählen. Der Eclipse-Workspace ist ein Verzeichnis auf Ihrem Computer oder dem Server, in welchem alle zu Ihrer Arbeitsumgebung notwendigen Dateien gespeichert werden.
Jedes in Eclipse erstellte Projekt wird in diesem Verzeichnis durch ein Unterverzeichnis repräsentiert. Der Workspace kann später jederzeit verändert werden (*File → Switch Workspace*).

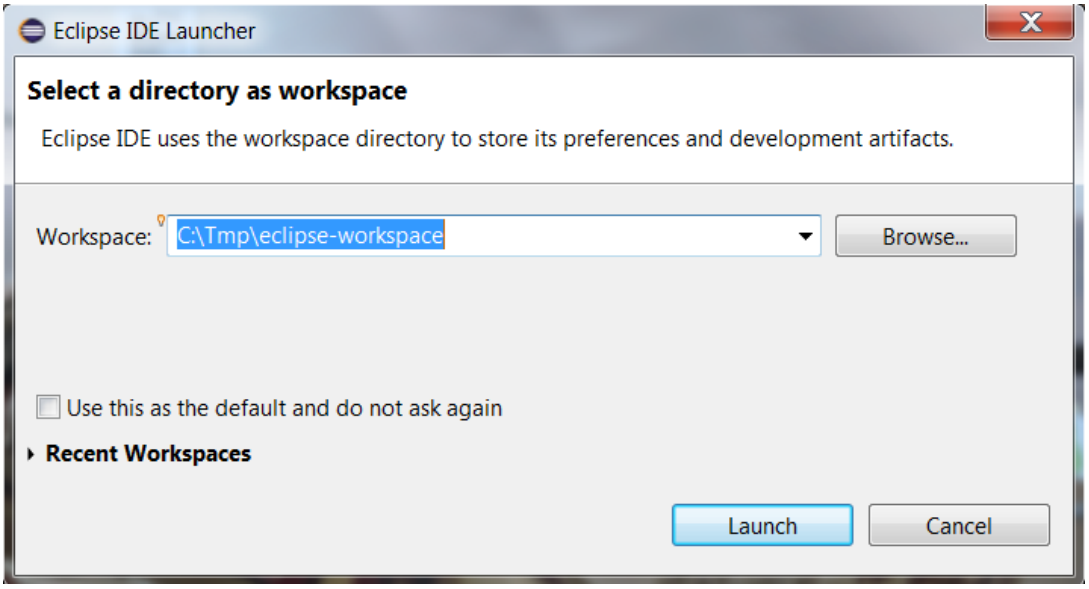

Abbildung 1 - Den Workspace auswählen

Wenn Sie zum ersten Mal Eclipse starten, sehen Sie eine Willkommensseite. Klicken Sie rechts oben auf den nach hinten zeigenden Pfeil (Workbench), somit wechseln Sie auf die Arbeitsoberfläche von Eclipse.

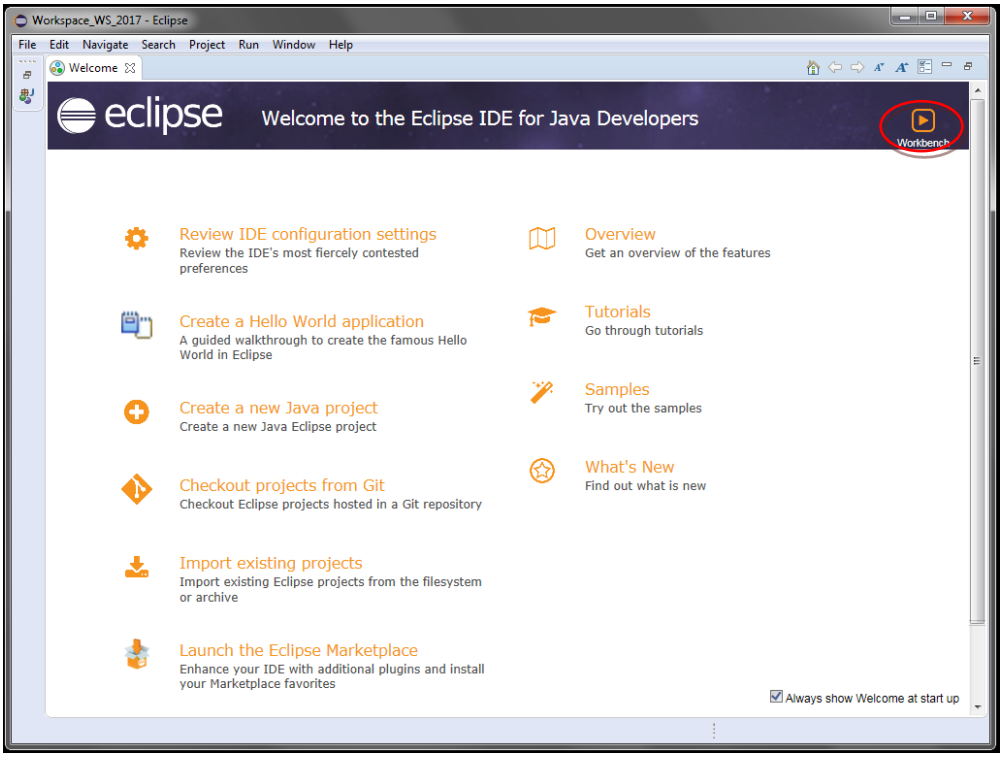

Abbildung 2 - Die Willkommensseite

Ein neues Projekt erstellen

1.1. Die Projektart wählen

Um Java-Programme mit der Eclipse-Entwicklungsumgebung erstellen zu können, müssen Sie zunächst ein neues Java-Projekt erstellen. Wählen Sie dazu aus dem Menü *File → New → Project*. Im nun erscheinenden Fenster wählen Sie im Bereich „Wizards“ den Eintrag „Java Project“. Klicken Sie auf „Next“. Die Kurzform ist über *File → New → Java Project* erreichbar.

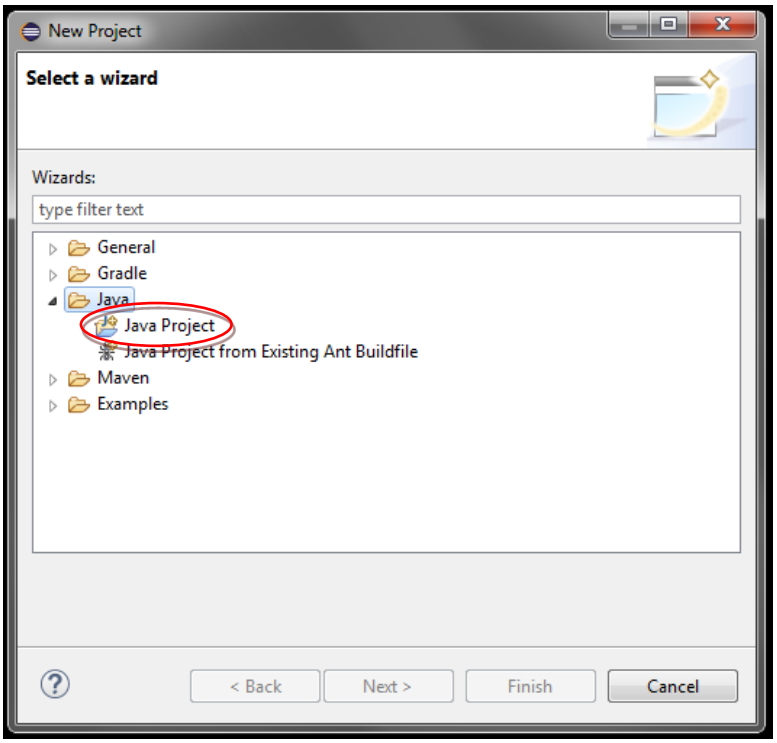

Abbildung 3 - Ein Java-Projekt erstellen

1.2. Den Projektnamen vergeben

Jedes in Eclipse erstellte Projekt muss einen Namen tragen. Im nächsten Fenster wählen Sie daher einen Namen für Ihr Projekt. Dieser Name sollte mit einem Großbuchstaben beginnen und möglichst keine Leerzeichen oder Sonderzeichen (ä,ü,ö,ß, etc.) beinhalten. Wählen Sie den Namen möglichst passend zum Einsatzzweck des Projektes. Wenn Sie z.B. in den Übungen verschiedene Aufgaben auf verschiedenen Übungsblättern durchführen, dann können Sie Ihre Projekte z.B. mit „Vorkurs_Blatt07_A03“ bezeichnen. Da jedes Projekt als separates Unterverzeichnis in Ihrem Workspace-Verzeichnis erstellt wird, erhalten Sie so automatisch eine gute Gliederung in Ihren Daten.

Wichtig!
Bei der ersten Erstellung wählen Sie bitte „Configure JREs…“ (siehe Abbildung 4). In Abbildung 5 sehen Sie das nun erscheinende Fenster. Wählen Sie dort am besten immer die letzte und somit aktuellste Version sowohl als Compiler als auch als JRE aus. Bestätigen Sie die Änderung des Standard-JDK mit „OK“ und das daraufhin erscheinende Fenster mit „Yes“. Sie gelangen zurück zum Fenster aus Abbildung 4.

Geben Sie den Projektnamen ein und klicken Sie auf „Finish“.

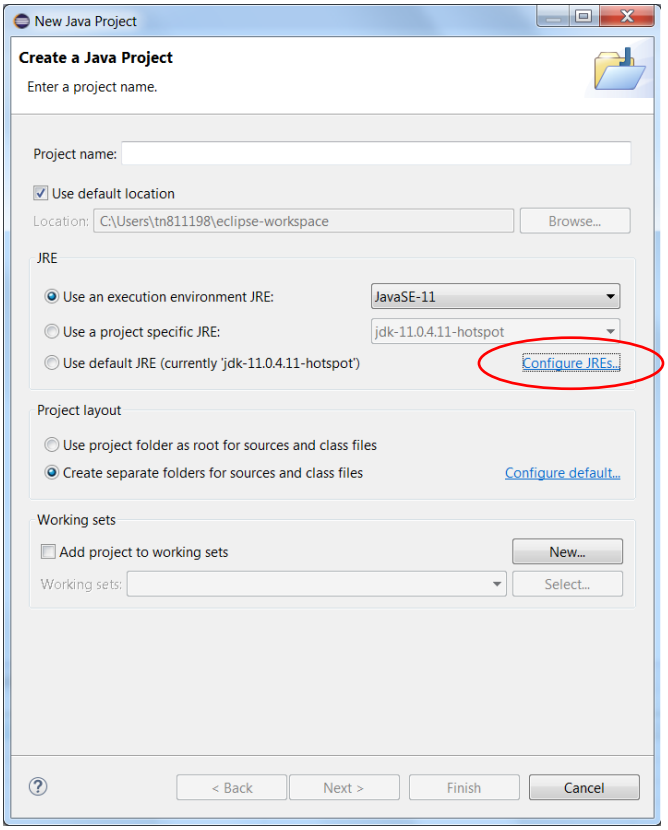

Abbildung 4 - Den Projektnamen wählen

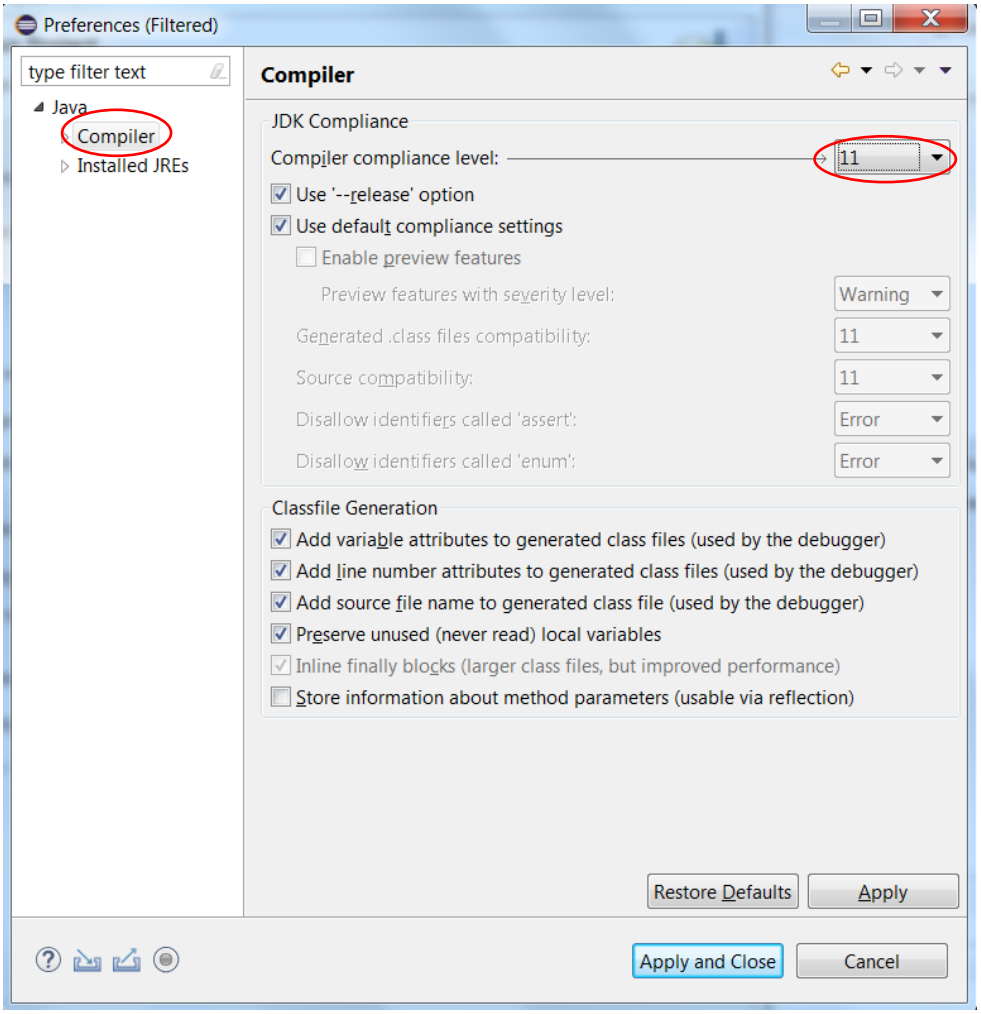

Abbildung 5 - Standard JDK auswählen

1.3. Die Java-Perspektive

Eclipse ist eine sehr umfangreiche Entwicklungsumgebung. Sie können mit ihr nicht nur Java-Programme schreiben, sondern auch viele andere Dinge machen. Für die Entwicklung von Java-Programmen schaltet Eclipse daher auf „Java-Perspective“ um. Eclipse fragt unter Umständen nach, ob Sie auf die Java-Perspektive wechseln möchten. Bestätigen Sie diese Frage.

Ihr Eclipse-Fenster enthält nun mehrere Unterfenster (siehe Abbildung 6).
Diese Unterfenster lassen sich untereinander verschieben. Dadurch können Sie die Perspektive an Ihre individuellen Bedürfnisse anpassen.

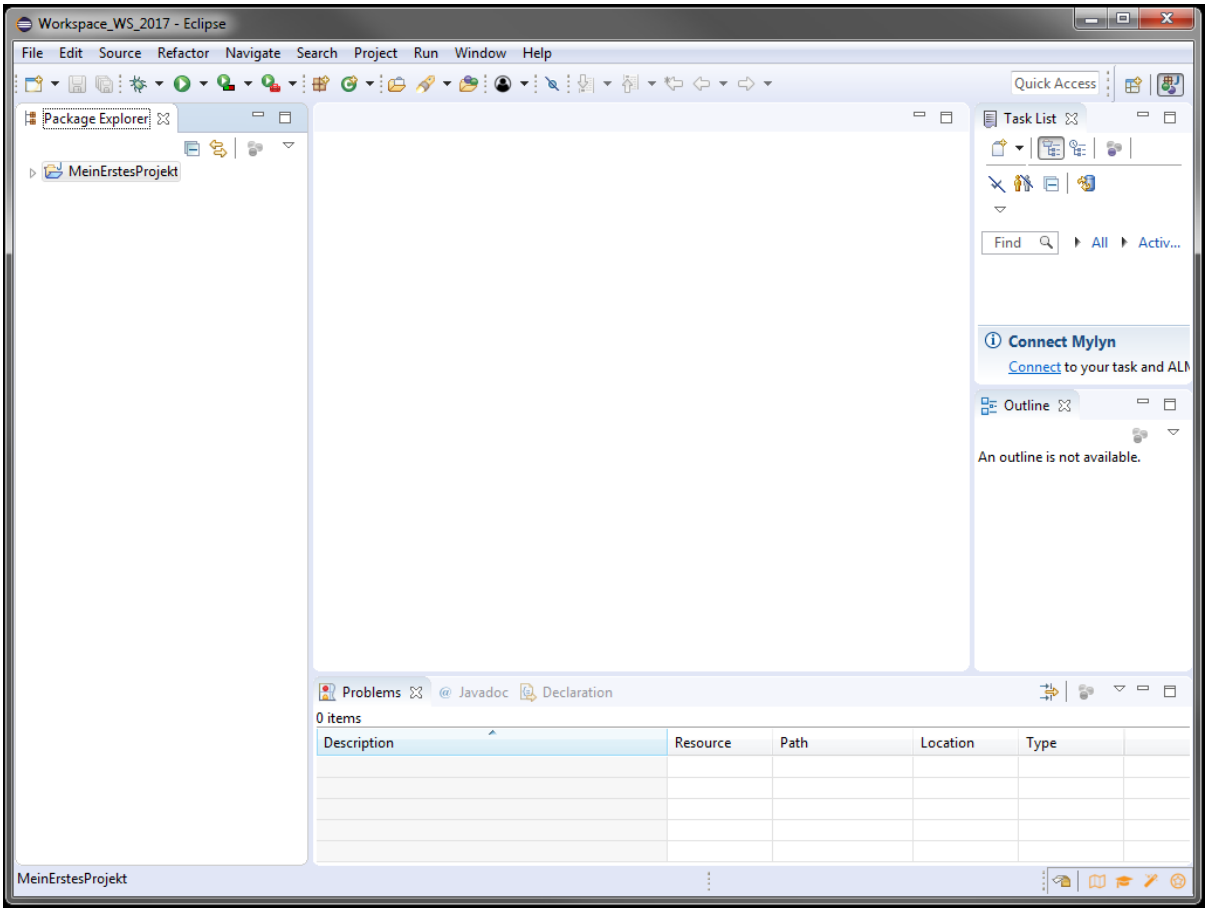

Abbildung 6 - Die Java-Perspektive mit den einzelnen Projekten

2. Eine neue Java-Klasse erstellen

Java-Programme benutzen Klassen, um die Programmierung zu strukturieren. Um ein erstes Programm zu erstellen, müssen Sie eine neue Klasse in Ihrem Projekt erstellen.

Klicken Sie dazu ihr Projekt an und wählen Sie aus dem Menü *File → New → Class.*

Danach erscheint ein Fenster wie in Abbildung 7, in dem Sie den Klassennamen eingeben müssen. Wählen Sie die Methoden-Rümpfe (method stubs) wie in der Abbildung unten angegeben.

Klicken Sie auf *Finish*, um die Klasse erstellen zu lassen.

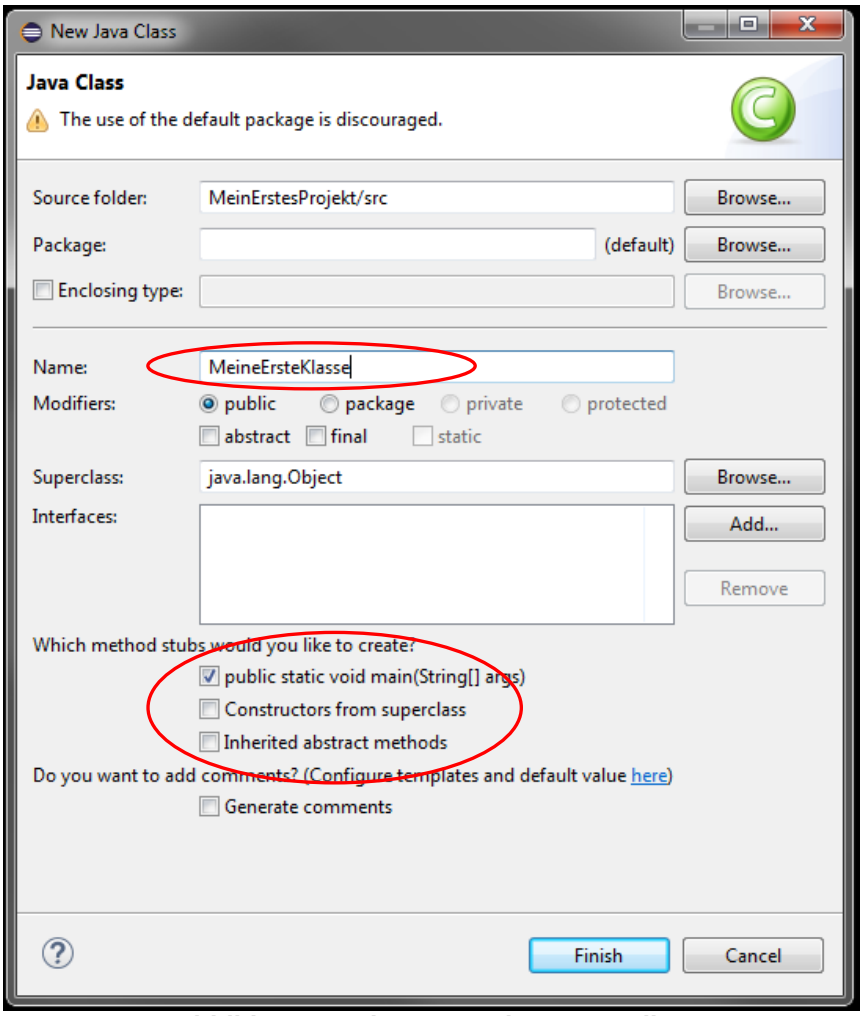

Abbildung 7 - Eine neue Klasse erstellen

2.1. Die erste Klasse

Nun sehen Sie im „Package Explorer“ auf der linken Seite die Klasse. Sie ist im mittleren Editor-Unterfenster geöffnet. Hier können Sie die Klasse nun erweitern.

Sie können nun z.B. die Zeile `System.out.println("Hello World");` einfügen. Hierbei ist wichtig, dass die Zeile zwischen den geschweiften Klammern { } nach `public static void main (String[] args)` steht (siehe Abbildung 8).

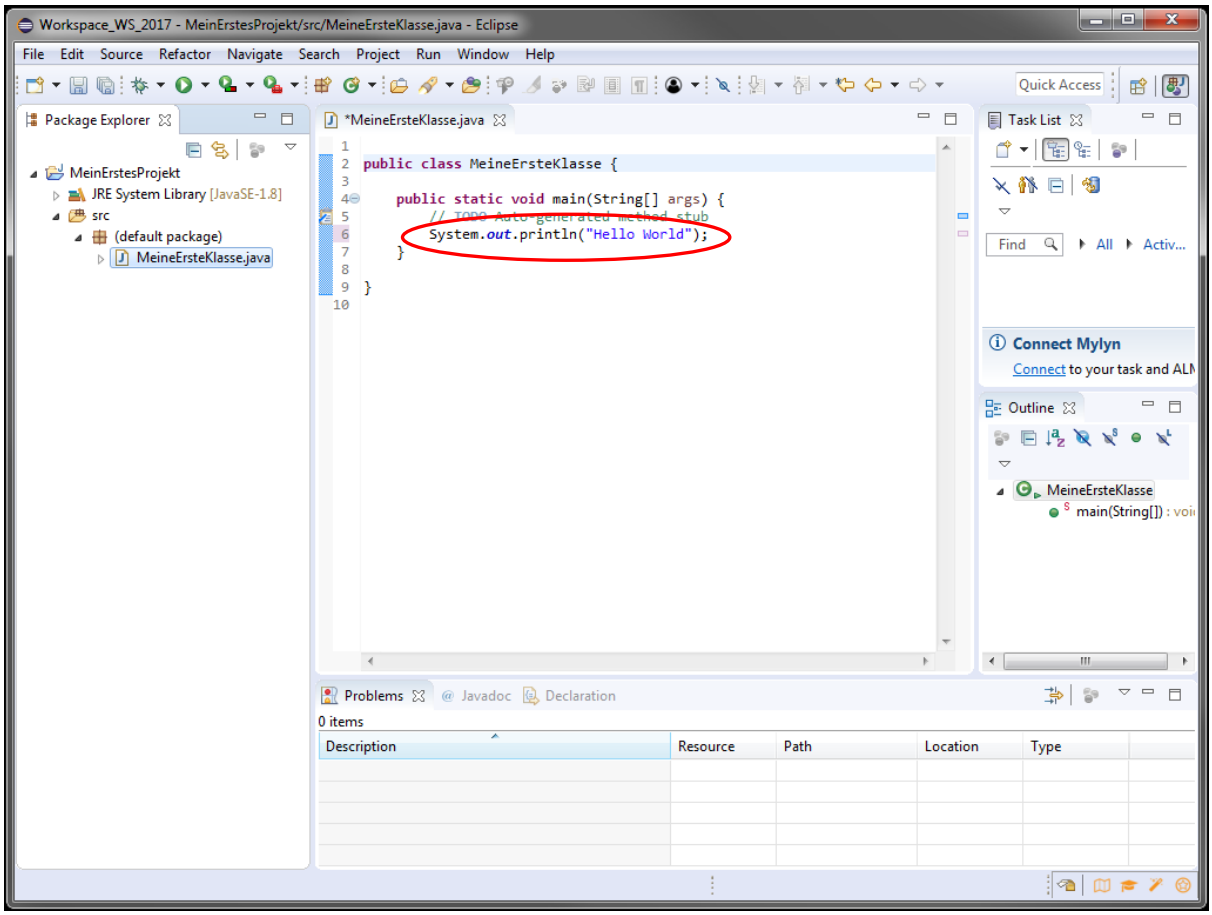

Abbildung 8 - Die erste Klasse im Java-Projekt

3. Kompilieren & Ausführen

3.1. Ein Java-Programm starten

Zum Starten des Java-Programms klicken Sie mit der rechten Maustaste auf diejenige Klasse, welche Ihre Main-Methode enthält. Wählen Sie nun, wie in Abbildung 9 gezeigt, aus dem Menü *Run As* → *Java Application*.

Alternativ lässt sich auch die Tastenkombination Alt+Shift+X, J benutzen.

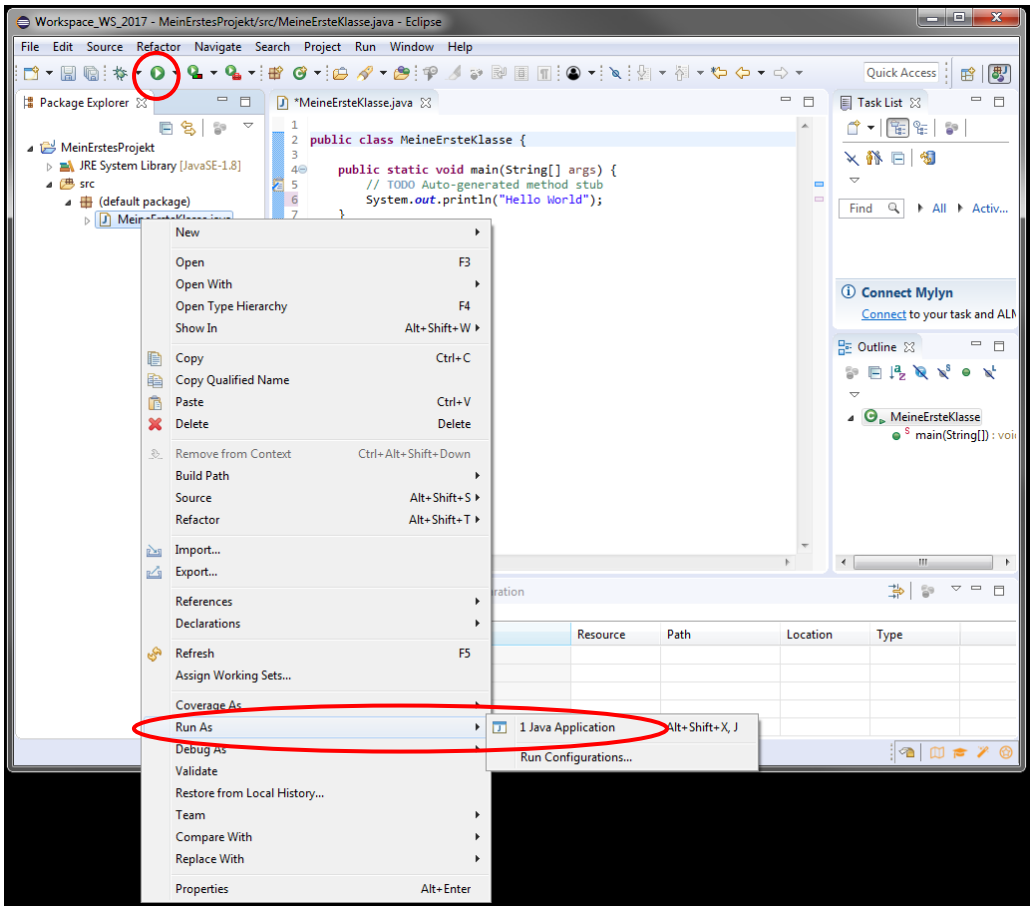

Abbildung 9 - Ein Java Programm ausführen

Nachdem die Klasse so das erste Mal zur Ausführung gebracht worden ist, reicht es ab dem zweiten Mal auf die grüne Pfeiltaste  in der oberen Symbolleiste zu klicken.

3.2. Das Ausgabefenster

Im Normalfall sollte im unteren Bereich neben den Reitern „Problems“, „Javadoc“ und „Declaration“ auch ein Reiter „Console“ zu finden sein. (Die „Console“ sollte spätestens nach der Ausführung des Programms automatisch erscheinen.) In diesem Reiter werden die von Java erzeugten Ausgaben hinein geschrieben. Sollte dieser Reiter bei Ihnen nicht vorhanden sein, so können Sie ihn über das Menü *Window* → *Show View* → *Console* oder mit der Tastenkombination Alt+Shift+Q, C zur Ansicht bringen.

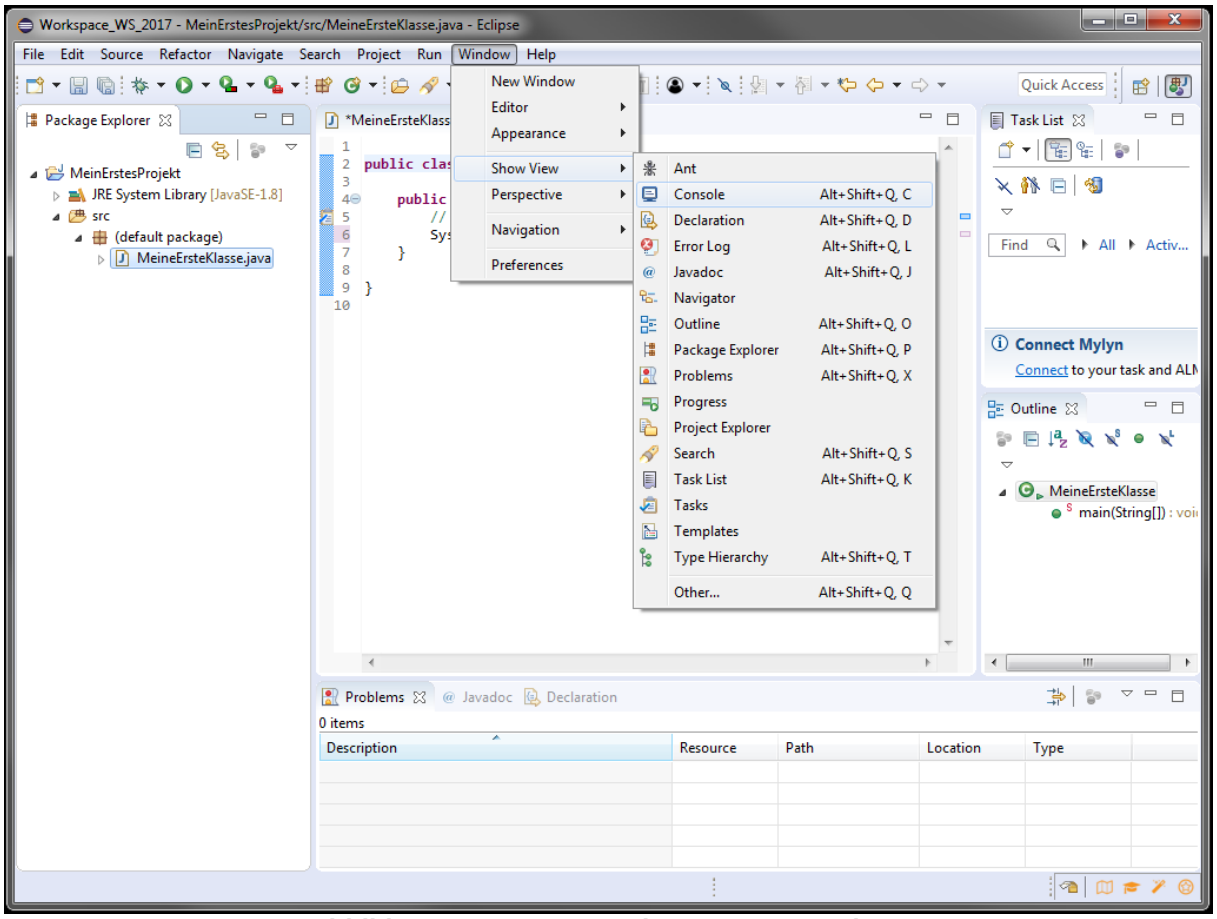

Abbildung 10 - Das Consolen-Fenster anzeigen

4. Der Debugger

Um bei der Programmerstellung den Inhalt von Variablen kontrollieren zu können, kann man sich die Variablen, z.B. mit `System.out.print()` ausgeben lassen. Diese Vorgehensweise scheint zwar im ersten Moment die einfachste, aber es geht einfacher und komfortabler: Mit dem Debugger.

4.1. Einen Breakpoint setzen

Beim Debuggen kann das Programm bis zu einem bestimmten Punkt laufengelassen und dann „pausiert" werden. Dieser Punkt nennt sich „Breakpoint". Ein Breakpoint lässt sich in einer beliebigen Programmzeile setzen (ausgenommen Kommentarzeilen, Leerzeilen, etc.). Um einen Breakpoint zu setzen, doppelklicken Sie mit der Maus auf die grau schraffierte Fläche vor der von Ihnen gewünschten Zeile. Nun erscheint dort ein Punkt. Sie können auch mit dem Cursor in die gewünschte Zeile gehen und die Tastenkombination Strg+Shift+B drücken.

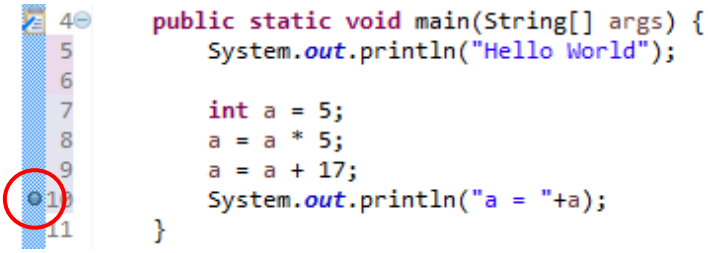

Abbildung 11 - Einen Breakpoint setzen

4.2. Den Debugger starten

Klicken Sie auf das kleine Käfer- (engl. Bug) Symbol  , um den Debugger zu starten. Alternativen sind hier der Rechtsklick in die Klasse mit der Main-Methode und die Auswahl von *Debug As → Java Application* oder der Shortcut Shift+Alt+D, J.
Für den Debugger hat Eclipse eine eigene Perspektive. In der Perspektive sind die einzelnen Fenster neu angeordnet und es kommen weitere Unterfenster hinzu. Eclipse fragt Sie, ob Sie in die Debug-Perspektive wechseln wollen. Setzen Sie den Haken bei „Rember my decision" und bestätigen Sie den Dialog mit „Yes".

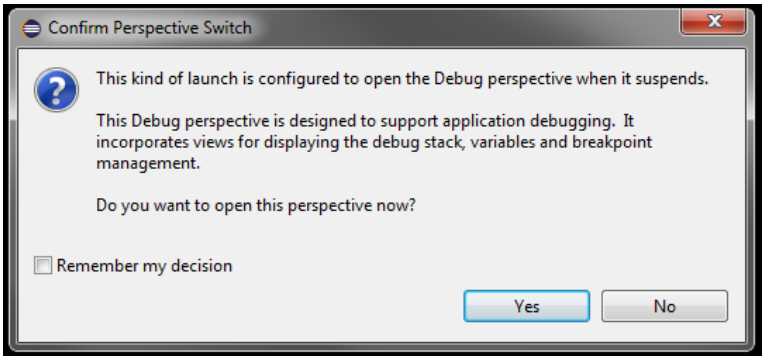

Abbildung 12 - In die Debug-Perspektive wechseln

4.3. Die Debug-Perspektive

In der Debug-Perspektive sehen Sie einige neue und einige alte Unterfenster in neuer Anordnung. In Abbildung 13 sehen Sie, wie das Programm bis zum vorher gesetzten Breakpoint gelaufen ist (1). Es wird nun ein kleiner Pfeil über dem Breakpoint angezeigt. Dort ist das Programm nun stehengeblieben. Es hat bereits „Hello World!“ ausgegeben (2), und die Variable a besitzt nun den Wert 42 (3). Im Unterfenster „Variables“ werden alle Variablen des aktuellen Kontextes angezeigt. Sie können das Programm nun durch Drücken der gelb-grünen Pfeiltaste (4 – Resume F8) bis zum nächsten Breakpoint laufen lassen. Es ist ebenfalls möglich, das Programm zeilenweise weiter laufen zu lassen. Dabei gibt es zwei Tasten (5), die von Bedeutung sind. Mit der ersten Taste (Step Into F5) springen Sie in ein eventuelles Unterprogramm, mit der zweiten Taste (Step Over F6) übergehen Sie einen eventuellen Unterprogrammaufruf. Nachdem Sie mit dem Debuggen fertig sind, können Sie über den Java Button (6) wieder in die Java Perspektive wechseln.

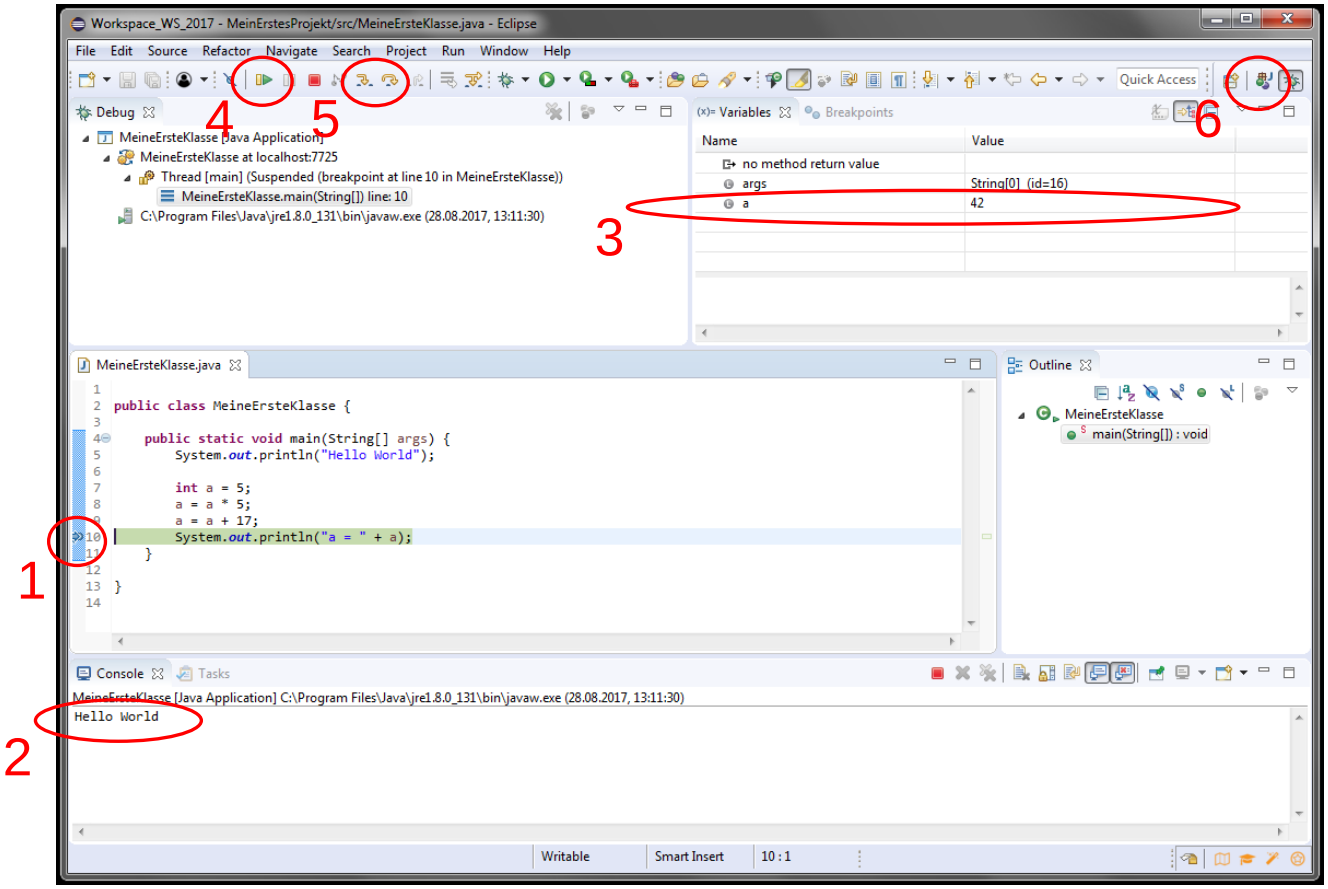

Abbildung 13 - Die Debug-Perspektive

5. Fehlermeldungen

Syntaxfehler werden in Eclipse bereits während der Eingabe markiert.
Im Editorfenster werden die Fehler rot geschlängelt unterstrichen. Sie werden ebenfalls im Reiter „Problems“ angezeigt. Die Syntaxüberprüfung während der Eingabe kann abgeschaltet werden, indem Sie das Menü *Window* → *Preferences* aufrufen und im dort erscheinenden Fenster unter *Java* → *Editor* den Haken vor *Report problems as you type* entfernen.

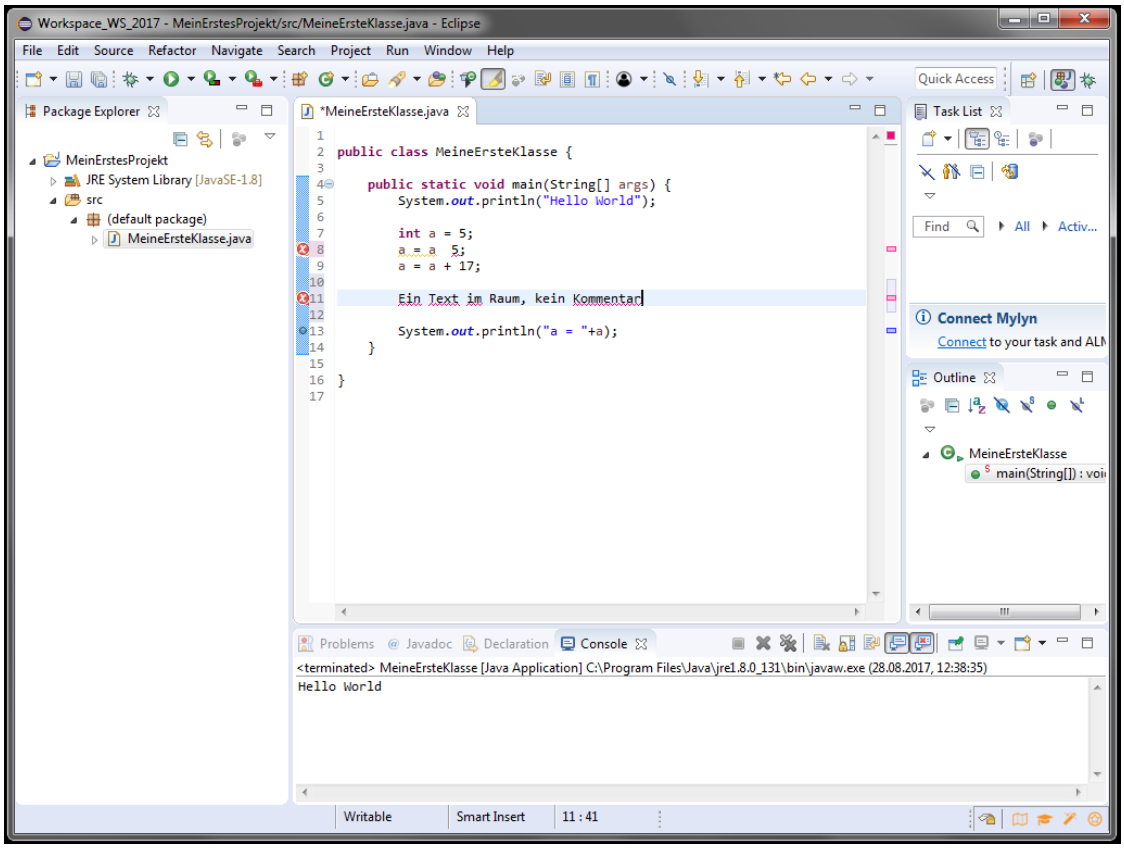

Abbildung 14 - Syntaxfehler während der Eingabe

6. Importieren von Java-Dateien

Zum Arbeiten mit Eclipse auf verschiedenen Rechnern empfiehlt es sich, den Eclipse-Workspace direkt auf einem USB-Stick abzuspeichern. Sie können einen lokal erstellten Workspace einfach über Ihr Dateisystem (z.B. Microsoft Explorer oder Midnight Commander unter Linux) auf ihren USB-Stick und von dort auf einen zweiten Rechner kopieren. Sie können den Workspace auch direkt von Ihrem USB-Stick öffnen.

Sollten Sie Dateien aus einem anderen Eclipse-Workspace bzw. Projekt importieren wollen, so können Sie im „Package Explorer“ mit der rechten Maustaste auf Ihr aktuelles Projekt klicken und dann den Menüpunkt „Import...“ wählen.

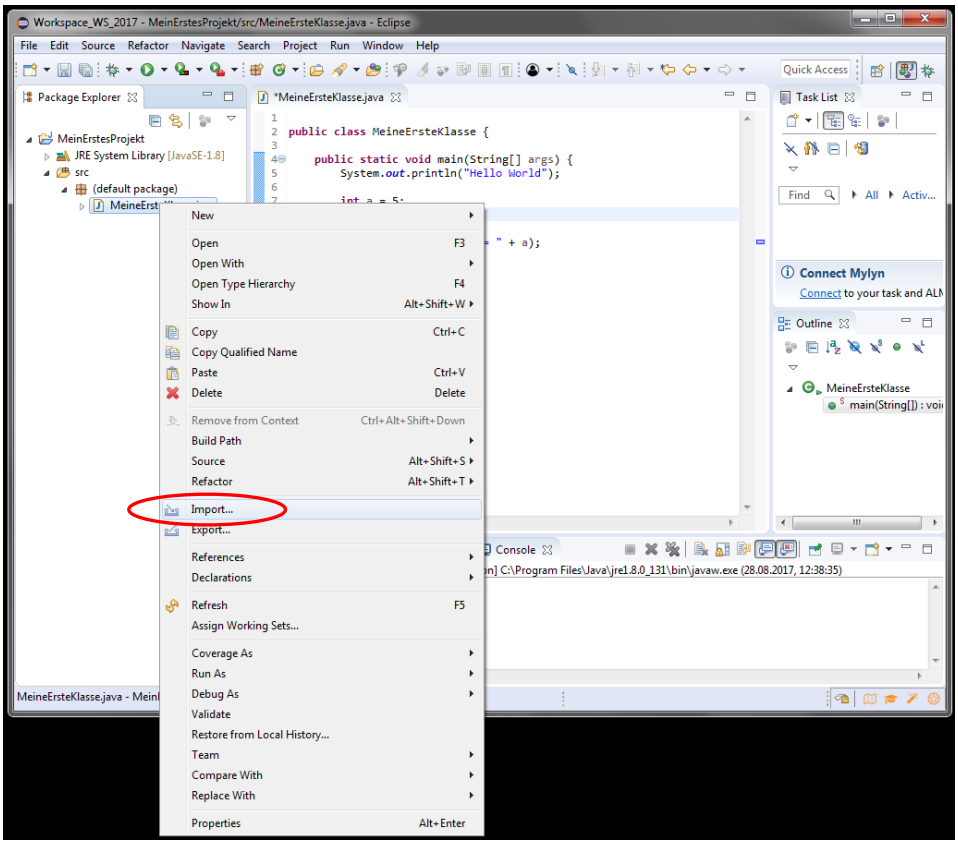

Abbildung 15 - Importieren

Im nun erscheinenden Fenster haben Sie verschiedene Möglichkeiten, Dateien in Ihre Projekte zu importieren. Sie können z.B. „File System“ für den Import aus einem anderen Java-Projekt benutzen.

7. Einstellungen

Über den Menüpunkt *Window* → *Preferences* können Einstellungen vorgenommen werden. Dabei ist das Eingabefeld links oben (in dem „type filter text“ steht) ein guter Weg, Einstellmöglichkeiten schnell zu finden. Um z.B. die Schriftgröße zu ändern, können Sie einfach in das Eingabefeld „font“ eintippen, und die Baumstruktur darunter öffnet sich schon während der Eingabe für die gefundenen Möglichkeiten (siehe Abbildung 16). Unter *Colors and Fonts* wählen Sie *Java* aus. Über den *Java Editor Text Font* lassen sich nun Schriftart und Schriftgröße verändern.

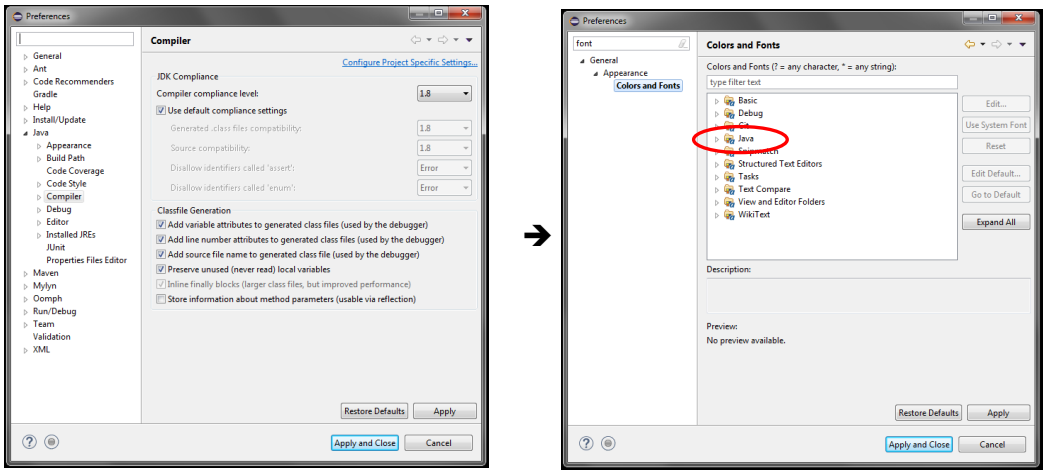

Abbildung 16 - Suche für Einstellmöglichkeiten

7.1. Zeilennummern

Sie können sich die Zeilennummerierung vor den Zeilen anzeigen lassen. Rechtsklicken Sie mit der Maus auf die grau schraffierte Fläche im Editor und wählen Sie den Menüpunkt „Show Line Numbers“. Diese Einstellung können Sie auch unter *Window* → *Preferences* → *General* → *Editors* → *Text Editors* global für alle Dateien vornehmen.

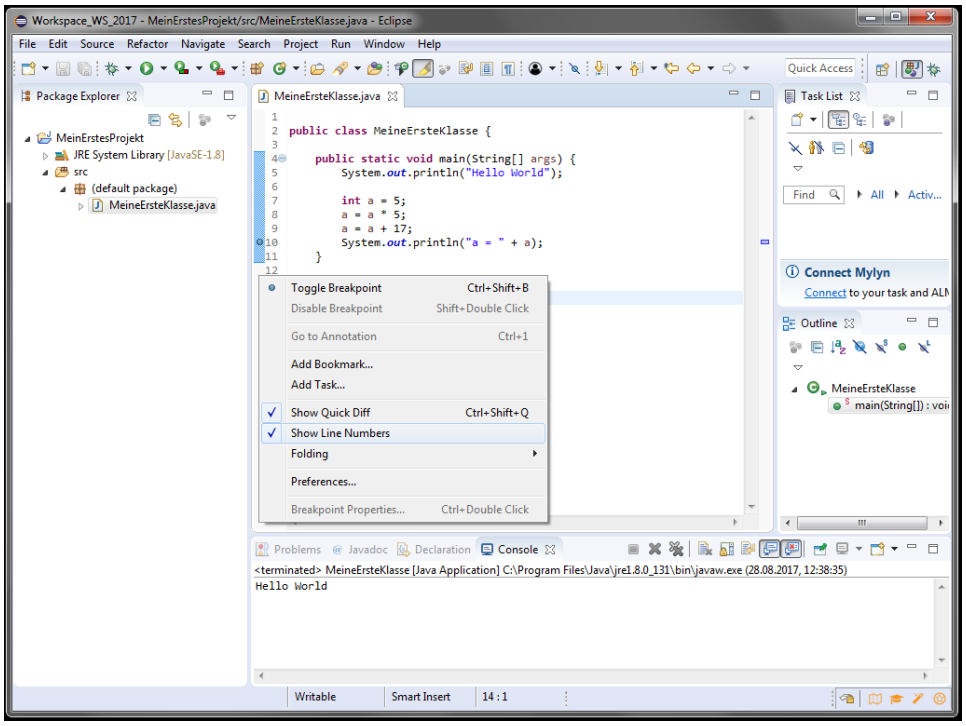

Abbildung 17 - Zeilennummerierung einschalten

7.2. Einbinden der API

Beim Programmieren ist es hilfreich, wenn man direkt die API nutzen kann. Diese sollte lokal vorhanden sein, weil man nicht zwangsweise immer eine Internetverbindung zur Verfügung hat.

Dafür müssen Sie Eclipse den lokalen Pfad Ihrer API bekannt geben. Dies funktioniert über *Window → Preferences → Java → Installed JREs*. Dort sehen Sie eine Übersicht der aktuell installierten JREs. Wählen Sie die JRE, von der Sie die API lokal auf ihrem Rechner liegen haben, und klicken auf *Edit*.

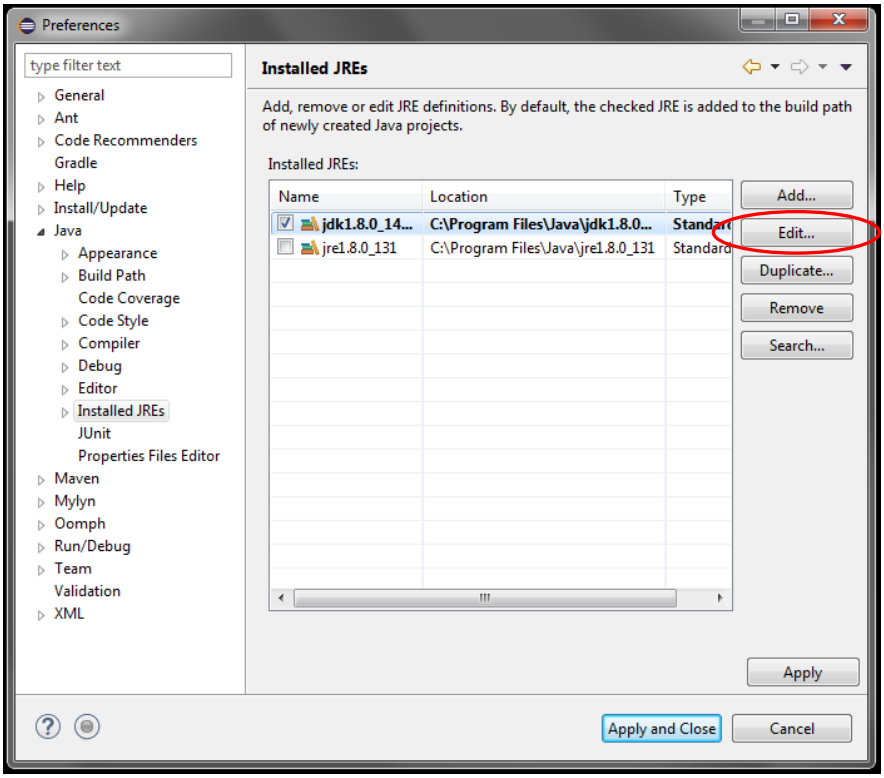

Abbildung 18 Einbinden der API

Wählen Sie die Library *rt.jar* aus und klicken *Javadoc Location* an. Unter *Javadoc URL* geben Sie nun den Pfad zu dem Unterverzeichnis *api* der lokalen und extrahierten(!) API an. Liegt Ihre API beispielsweise unter `C:\User\Matse\jdk-8u51-docs-all`, lautet die von Ihnen anzugebende URL `C:\User\Matse\jdk-8u51-docs-all\docs\api`.

Wenn Sie nun Ihren Mauszeiger im Quellcode über eine Klasse oder eine Methode stehen haben, bekommen Sie automatisch in einem Tooltip die zugehörige API angezeigt. Wenn Sie diese in einem separaten Fenster in Eclipse betrachten möchten, können Sie dies mit Shift+F2 machen.

8. Templates & Shortcuts

8.1. Wichtige Templates

In Eclipse können Sie sich ein bisschen Tipp-Arbeit sparen, indem Sie Templates für wichtige Befehle benutzen. Sie können diese Templates aufrufen, indem Sie den Kurznamen des Templates in den Editor schreiben (z.B. „sysout“) und dann die Tastenkombination „Strg+Leertaste“ drücken. Nun erscheint der vollständige Code (hier „System.out.println “).

Die wichtigsten Templates können Sie über das Menü „Window“ → „Preferences“ und dort über die Gruppe „Java“ → „Editor“ → „Templates“ sehen.

Hier eine kleine Auswahl der wichtigsten Templates:

Tabelle 1 - Wichtige Templates

Templatename	Beschreibung	Erzeugter Code
`sysout`	print to standard out	`System.out.println();`

`if`	if statement	`if (...) {`
		
		`}`

`ifelse`	if else statement	`if (...) {`
		
		`} else {`
		
		`}`

`main`	main method	`public static void main(String[] args) {`
		
		`}`

`for`	iterate over array	`for (int i = 0; i < array.length; i++) {`
		
		`}`

`while`	while loop with condition	`while (...) {`
		
		`}`

`try`	try catch block	`try {`
		
		`} catch (Exception e) {`
		
		`}`

8.2. Schnelle Shortcuts

Viele Funktionen in Eclipse lassen sich mit Hilfe von Tastenkombinationen schneller aufrufen. Hier ist eine kleine Liste von wichtigen Tastenkombinationen:

Tabelle 2 - Shortcuts

Tastenkombination	Auswirkung
Alt + Shift + X, J Strg + F11	Java Anwendung ausführen
Alt + Shift + D, J	Java Anwendung debuggen
Alt + Shift + Q, C	Consolenfenster anzeigen
F8	Debugger weiterlaufen lassen (Resume)
F5	Debugger: Unterprogrammaufruf (Step Into)
F6	Debugger: Unterprogrammaufruf überspringen (Step over)
Strg + Shift + B	Breakpoint ein-/ausschalten
Strg + F	Suchen/Ersetzen
Strg + K	Nächstes Vorkommen finden
Strg + S	Aktuelle Datei speichern
Strg + A	Gesamten Text markieren
Strg + Shift + F	Markierten Text formatieren
Strg + C	Markierten Text in die Zwischenablage kopieren (copy)
Strg + V	Text aus der Zwischenablage einfügen (paste)
Strg + X	Markierten Text entfernen und in der Zwischenablage ablegen
Strg + Shift + C Strg + 7	Zeile des Cursors bzw. markierte Zeilen auskommentieren
Strg + D	Zeile des Cursors löschen
Strg + Shift + O	Alle fehlenden Imports automatisch erkennen und hinzufügen
Strg + 1	Vorschläge zur Syntaxfehlerbehebung zum markierten Fehler anzeigen

9. Abbildungs- und Tabellenverzeichnis

9.1. Abbildungen

9.2. Tabellen