



Designing Education
Connecting People

Das erwartet Sie:

- Java-Grundlagen
- Entwicklungswerkzeuge



Funktionalität in Anwendungen realisieren

Lernfeld 11a

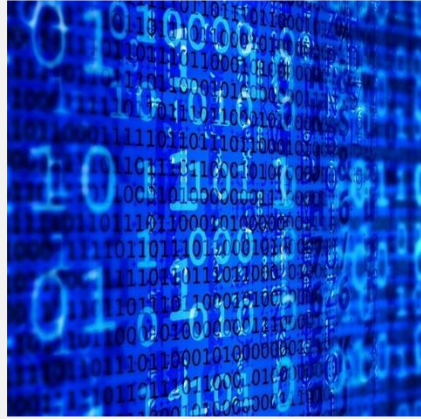
Die Themen und Lernziele



Algorithmen

Lernziel

Verstehen, worum es bei Algorithmen geht und wo man sie einsetzt



Funktionalität in Anwendungen realisieren

Lernziel

Eine typsichere, objekt-orientierte Programmiersprache beherrschen



Benutzerschnittstellen gestalten und entwickeln

Lernziel

Grafische Oberflächen in einer OO-Sprache entwickeln



Software testen

Lernziel

Testfälle formulieren und anwenden

Überblick

Java



**Java-
Grundlagen**

**Objekt-
orientierung**

Praxis



Warum Java?

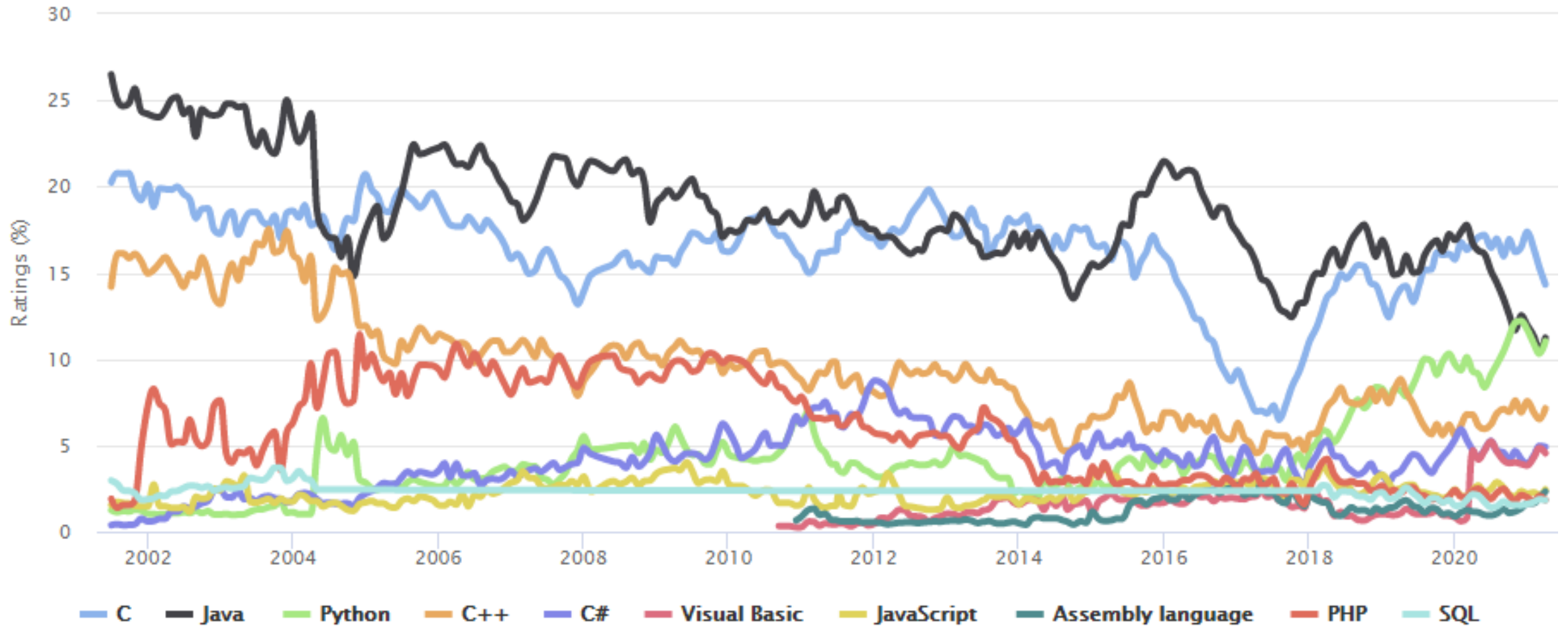
Lernziel

Eine typsichere,
objektorientierte
Programmiersprache
beherrschen

Ranking

TIOBE Programming Community Index

Source: www.tiobe.com



Stand: April 2021

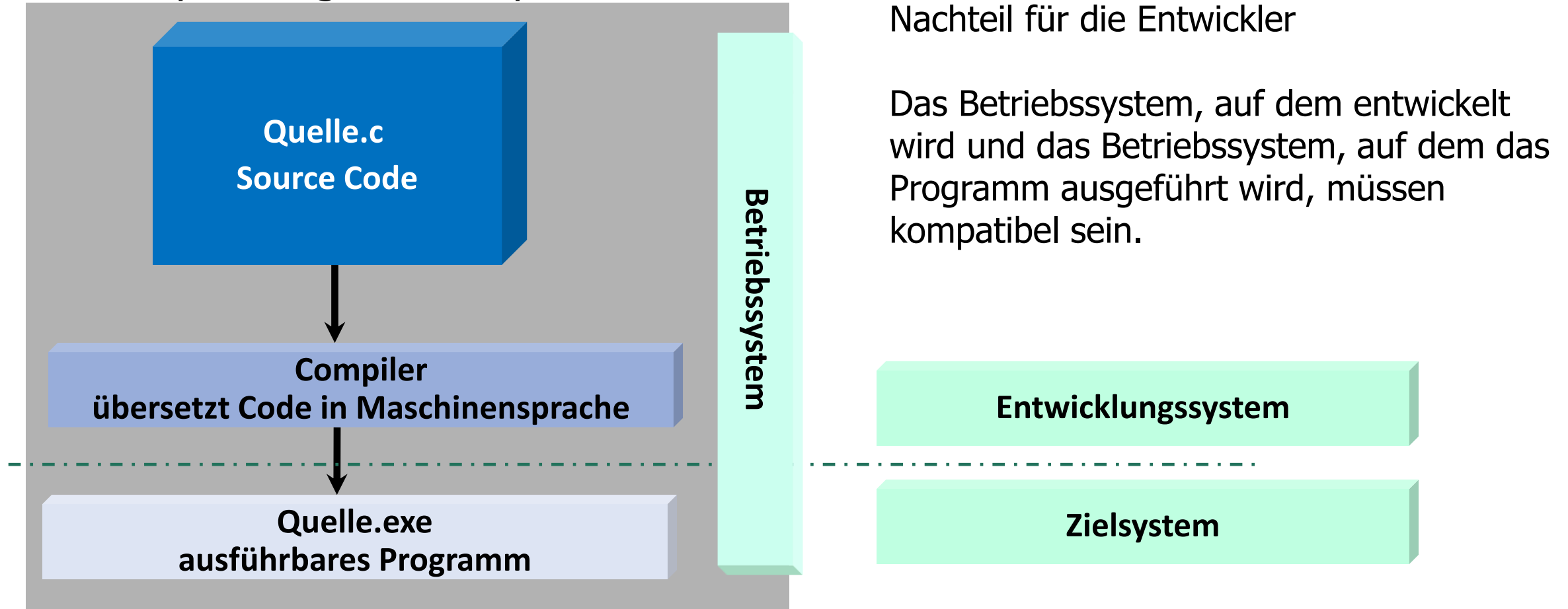
Warum Java

- Java ist eine Programmiersprache
- Java ist einfach, weil strukturiert
- Java ist objektorientiert
- Java unterstützt Multi-threading
- Java ist typsicher
- Java hat automatische Speicher- und Heap-Verwaltung
- Java ist rückwärtskompatibel bis zur Version 1.0
- Java ist plattformunabhängig
- Java-Entwickler sind gefragt

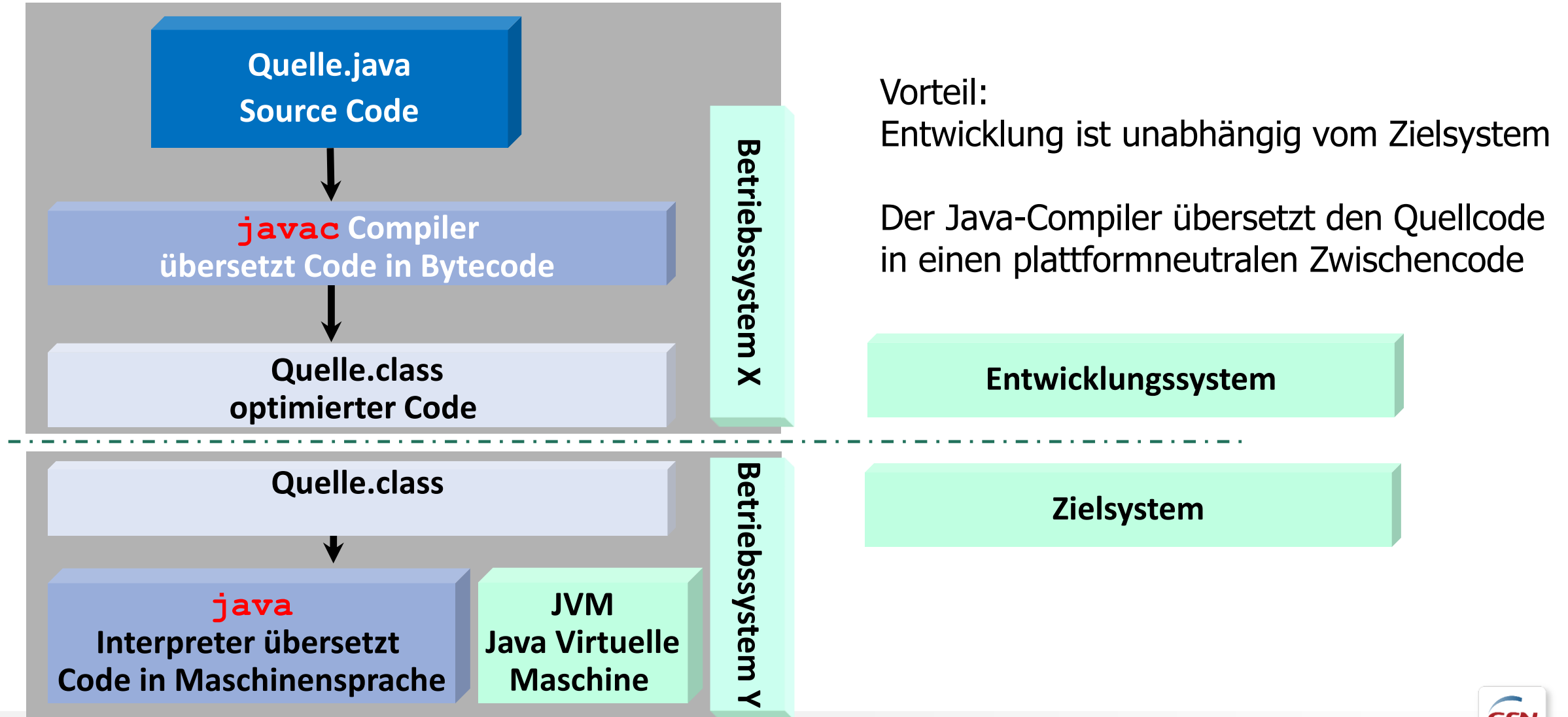


Plattformabhängige Sprache

Am Beispiel Programmiersprache C



Plattformunabhängige Sprache Java



Entwicklungsumgebung

- SDK** *Software Development Kit* ist eine allgemeine Bezeichnung einer Entwicklungsumgebung
- JDK** *Java Development Kit* enthält neben der
- API** *Application Programming Interface* (alle Klassen der Java-Sprache) den Java Compiler (javac), sowie die
- JRE** *Java Runtime Environment* und weitere hilfreiche Werkzeuge
- IDE** *Integrated Development Environment*

Die Distribution der JDK und JRE erfolgt über www.oracle.com

IDE

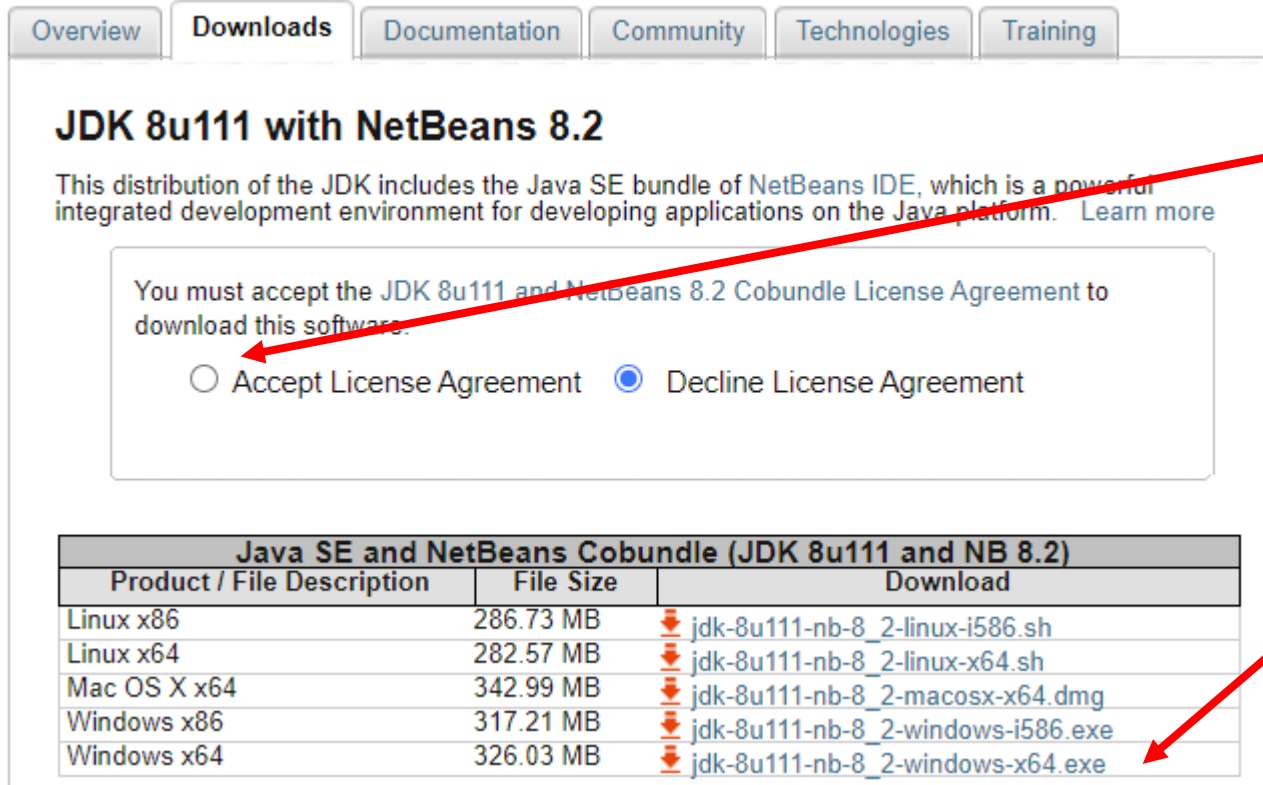
Integrated Development Environment

Als Entwicklungsumgebung werden wir [NetBeans](#) einsetzen

bereits integriert:

- jdk
- jre
- Intelligenter Editor
- GUI Designer
- uvm.
- Download [JDK 8u111 with NetBeans 8.2 - Oracle](#)

IDE NetBeans Download



The screenshot shows the 'Downloads' tab of the NetBeans website. It features a section for 'JDK 8u111 with NetBeans 8.2' which includes a license agreement checkbox. Below this is a table titled 'Java SE and NetBeans Cobundle (JDK 8u111 and NB 8.2)' with columns for 'Product / File Description', 'File Size', and 'Download'. The table lists download links for Linux x86, Linux x64, Mac OS X x64, Windows x86, and Windows x64. Red arrows point from the instructional text on the right to the license agreement and the Windows x64 download link.

JDK 8u111 with NetBeans 8.2

This distribution of the JDK includes the Java SE bundle of NetBeans IDE, which is a powerful integrated development environment for developing applications on the Java platform. [Learn more](#)

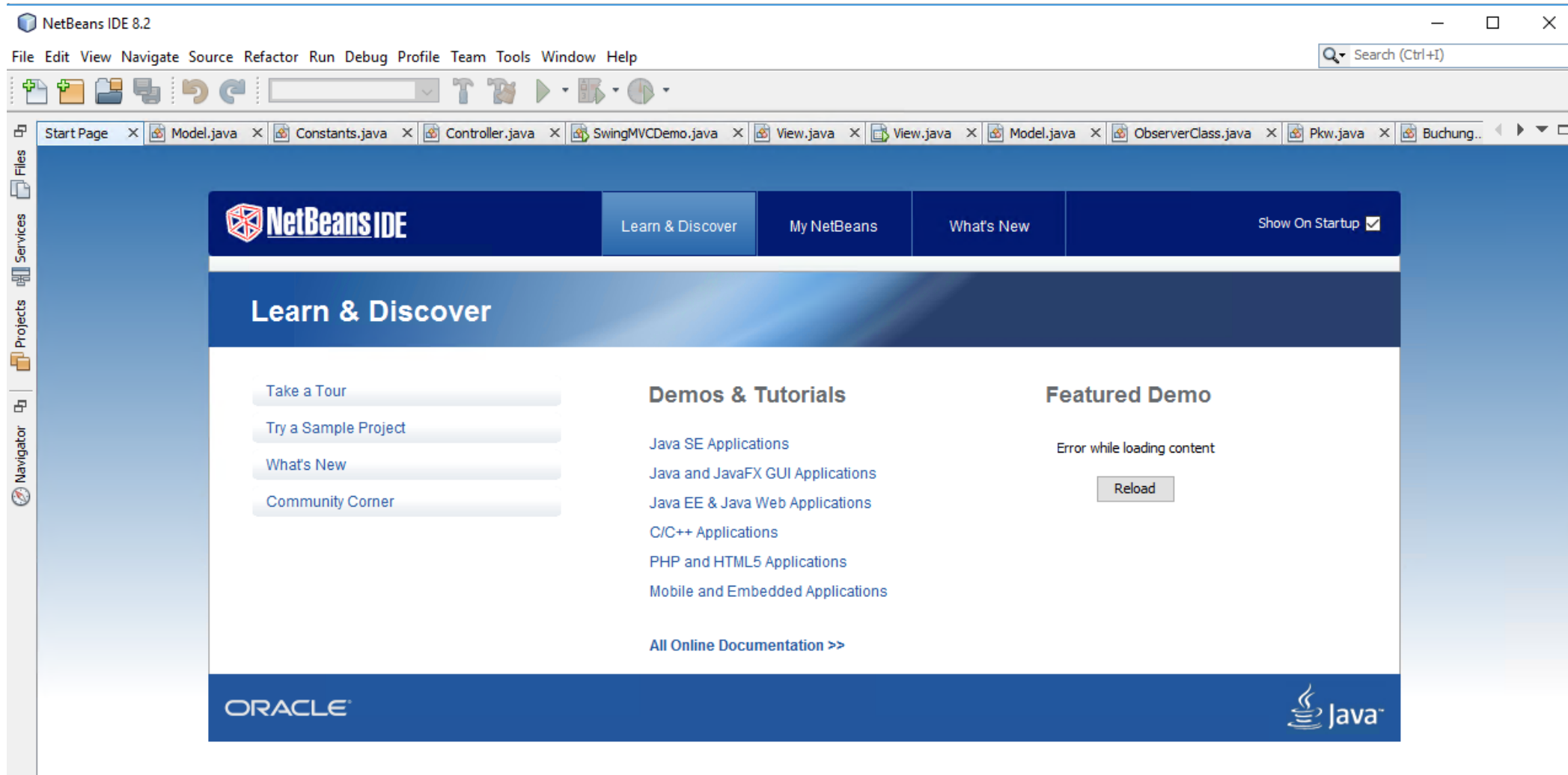
You must accept the JDK 8u111 and NetBeans 8.2 Cobundle License Agreement to download this software.

☐ Accept License Agreement ☒ Decline License Agreement

Java SE and NetBeans Cobundle (JDK 8u111 and NB 8.2)		
Product / File Description	File Size	Download
Linux x86	286.73 MB	jdk-8u111-nb-8_2-linux-i586.sh
Linux x64	282.57 MB	jdk-8u111-nb-8_2-linux-x64.sh
Mac OS X x64	342.99 MB	jdk-8u111-nb-8_2-macosx-x64.dmg
Windows x86	317.21 MB	jdk-8u111-nb-8_2-windows-i586.exe
Windows x64	326.03 MB	jdk-8u111-nb-8_2-windows-x64.exe

1. Für den Download müssen die Lizenzbedingungen akzeptiert werden
2. Wählen Sie das passende Produkt aus
3. Nach erfolgreichem Download starten Sie die Installation auf Ihrem System

Welcome NetBeans




Wie geht's weiter

- NetBeans wurde mit dem Java Development Kit erfolgreich installiert
- Das NetBeans-Fenster schließen wir
- Die nächsten Schritte werden in der Konsole ausgeführt
- Wenn unsere Programme komplexer werden, benutzen wir NetBeans für unsere Projekte

Java Compiler in der Konsole

- Mit der NetBeans-Installation wurde auch die jdk und jre installiert
- Um den Zusammenhang von Compiler und Runtime zu verstehen, nutzen wir das Konsolen-Fenster
- Konsole starten mit **Windows-Taste** und gleichzeitig **R**, Eingabe **cmd**, **OK**-Button wählen oder
Mausklick auf Windows-Symbol und „Eingabeaufforderung“ (oder nur „cmd“) eingeben
- Bevor wir ein erstes Java-Programm erstellen, testen wir, ob der Java-Compiler-Aufruf **javac** funktioniert
- Leider nicht

 Auswählen C:\windows\system32\cmd.exe

```
C:\Users\guenter>javac
Der Befehl "javac" ist entweder falsch geschrieben oder
konnte nicht gefunden werden.

C:\Users\guenter>
```

- Fenster schließen und Zusatzmaterial „[Java Compiler.pdf](#)“ öffnen und den Anweisungen folgen

Verzeichnisstruktur anlegen

- Sie werden in diesem Lernfeld und den folgenden Lernfeldern noch viele Programme/Klassen programmieren
- Legen Sie sich ein Basisverzeichnis für Ihre Programme an, z. B. [java](#)
- In diesem Basisverzeichnis erstellen Sie weitere Unterverzeichnisse, z. B. Tag01
Diese Struktur dokumentiert Ihren Lernfortschritt ganz nebenbei nach Tagen geordnet
So lassen sich auch versäumte Unterrichtstage mit Unterlagen von anderen Teilnehmenden zuordnen
- Beispiel in der Konsole:

```
C:\Users\guenter>mkdir java
```

```
C:\Users\guenter>cd java
```

```
C:\Users\guenter\java>mkdir Tag01
```

```
C:\Users\guenter\java>mkdir Tag02
```

Editorauswahl

- Um Programme zu schreiben, braucht man einen Editor
- NetBeans bietet im Editor sehr viele Hilfen an, die bereits während des Schreibens auf mögliche Fehler hinweisen. Das ist während der Erstellung komplexer Programme sehr hilfreich.
- Der Lernerfolg ist aber größer, wenn man mit einem einfachen Editor beginnt. Dieser sollte mindestens Folgendes enthalten:
 - Syntax-highlighting
 - Autovervollständigung bei paarweise auftretenden Symbolen, z. B. () , { }, “ ”
 - Einrückungen

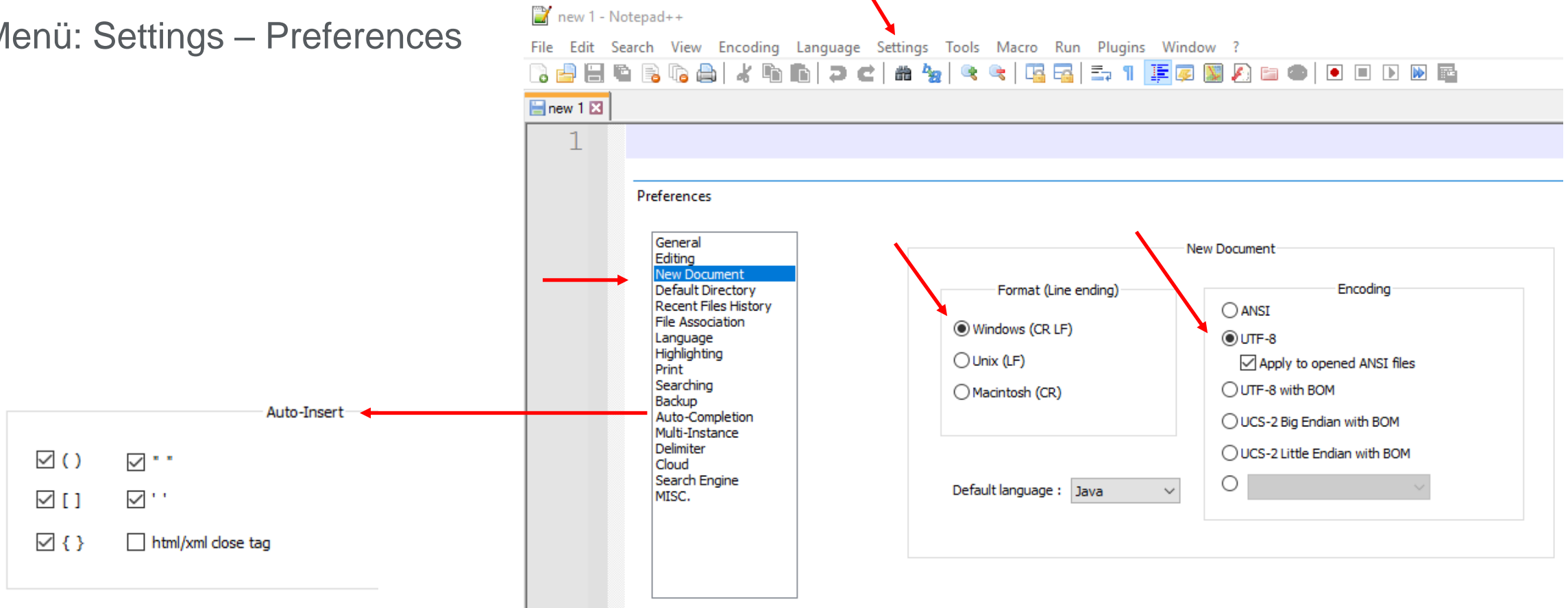
Notepad++ ist dafür eine gute Wahl



Notepad++-Einstellungen

- Folgende Einstellungen sollten Sie vornehmen:

Menü: Settings – Preferences



Java lernen



Wie lernt man eine neue
Programmiersprache?

Was man hört, das vergisst man.

Was man sieht, daran kann man sich erinnern.

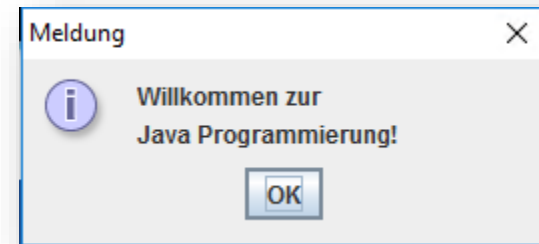
Erst was man tut, kann man verstehen.

Erstes Programm in der Konsole

- Erstellen Sie das folgende Programm im Editor und speichern es als `Willkommen.java` ab; Groß-/Kleinschreibung beachten

```
1 import javax.swing.JOptionPane;
2 /**
3  * Ausdruck von 2 Zeilen in einem Dialogfenster
4  */
5 public class Willkommen {
6     public static void main(String... args) {
7         JOptionPane.showMessageDialog(null, "Willkommen zur\nJava Programmierung!");
8     }
9 }
```

- Kompilieren Sie die Datei mit `javac Willkommen.java` in der Konsole
- Wenn kein Fehler angezeigt wird; ausführen mit `java Willkommen`



Programmbestandteile

1. Ein Java-Programm ist stets eine Klasse und beginnt mit den Schlüsselworten **public class** sowie dem Namen der Klasse (Zeile 5)
2. Ein Java-Programm ist in Einheiten gepackt, sogenannte Blöcke, die von geschweiften Klammern { . . . } umschlossen sind und oft vorweg mit einer Bezeichnung versehen sind (z. B. **public class Willkommen** oder **public static void main(. . .)**)
Zur Übersichtlichkeit sind die Blöcke stets eingerückt.
3. Der Startpunkt jeder Applikation ist die Methode **main**. Hier beginnt der Java-Interpreter (die „virtuelle Maschine“), die einzelnen Anweisungen auszuführen.
Er arbeitet sie „sequenziell“, d. h. der Reihe nach ab (Zeile 6).

Programmbestandteile

4. Die Applikation besteht aus einer einzigen Anweisung (Zeile 7):
Sie zeigt einen Text in einem Fenster an.
5. Um Spezialfunktionen zu verwenden, die von anderen Entwicklern erstellt wurden, kann man mit der Anweisung **import** bereits programmierte Klassen importieren.

Eine Klasse ist in diesem Zusammenhang also so etwas wie ein „Werkzeugkasten“, aus dem wir fertig programmierte „Werkzeuge“ (hier die Methode **showMessageDialog** aus **JOptionPane**) verwenden können.

Eine der wenigen Klassen, die wir nicht importieren müssen, ist die Klasse **System**.

Klassendeklaration und Klassenname

- In der Zeile 5 beginnt die Klassendeklaration der Klasse **Willkommen**.
- **public class** sind reservierte Worte. Sie sind von Java belegt und werden immer klein geschrieben. Der Klassenname Willkommen ist frei wählbar, solange er nicht einem reservierten Wort entspricht.
- Der Klassenname ist ein so genannter Identifier oder Bezeichner.
- Es gilt die Konvention:
 - Klassennamen beginnen mit einem Großbuchstaben
 - Variablennamen beginnen mit einem Kleinbuchstaben
 - Die Namen müssen mit einem Buchstaben, einem Unterstrich oder einem Dollarzeichen beginnen
 - Die Namen dürfen keine Leerzeichen enthalten
- Gültige Bezeichner sind z. B.: **Garten1**, **_wert**, **\$wert**, **GartenNachbar** (hier Camel Case)
- Java ist *case sensitive*, das heißt, die Variable **a5** ist eine völlig andere als **A5**

Die Methode main

- Beim Aufruf `java Willkommen` erwartet die hier gestartete virtuelle Maschine (JVM) in der Klasse Willkommen eine main-Methode.
- Ein komplexes Programm kann aus vielen `.class` Dateien bestehen, die sich aufgrund des internen Codes „kennen“ und von der JVM ansprechbar sind. Gestartet wird aber das Programm immer mit dem Aufruf der Klasse, die eine main-Methode enthält.
- Die Deklaration der main-Methode kann in 2 Varianten erfolgen:
 - `public static void main(String... args)`
 - `public static void main(String[] args)`

Reservierte Wörter und Literale

<code>abstract</code>	<code>assert</code>	<code>boolean</code>	<code>break</code>	<code>byte</code>	<code>case</code>	<code>catch</code>
<code>char</code>	<code>class</code>	<code>continue</code>	<code>default</code>	<code>do</code>	<code>double</code>	<code>else</code>
<code>enum</code>	<code>extends</code>	<code>final</code>	<code>finally</code>	<code>float</code>	<code>for</code>	<code>if</code>
<code>implements</code>	<code>import</code>	<code>instanceof</code>	<code>int</code>	<code>interface</code>	<code>long</code>	<code>native</code>
<code>new</code>	<code>package</code>	<code>private</code>	<code>protected</code>	<code>public</code>	<code>return</code>	<code>short</code>
<code>static</code>	<code>strictfp</code>	<code>super</code>	<code>switch</code>	<code>synchronized</code>	<code>this</code>	<code>throw</code>
<code>throws</code>	<code>transient</code>	<code>try</code>	<code>var</code>	<code>void</code>	<code>volatile</code>	<code>while</code>

Literale

<code>false</code>	<code>true</code>	<code>null</code>	<code>0, 1, -2, 5L</code>	<code>1.2, .0, 2.0f</code>	<code>'A', 'b', ' '</code>	<code>"Abc", ""</code>
--------------------	-------------------	-------------------	---------------------------	----------------------------	----------------------------	------------------------

In Java nicht verwendete reservierte Wörter

<code>byvalue</code>	<code>cast</code>	<code>const</code>	<code>future</code>	<code>generic</code>	<code>goto</code>	<code>inner</code>
<code>operator</code>	<code>outer</code>	<code>rest</code>				

Kommentare

- `// ...` Ein einzeliger Kommentar beginnt mit dem Doppelslash und endet am Zeilenende. Text vor dem Doppelslash wird vom Compiler als Code interpretiert.
- Mehrere Zeilen werden kommentiert mit `/*` beginnend und endend mit `*/`
`/* Dies ist ein Kommentar, der
sich über mehrere Zeilen
erstreckt. */`

Für Dokumentationen gibt es das Kommentarzeichen `/** */`, aus dem der Aufruf `javadoc` eine Programmdokumentation generiert, die dem Format der API-Dokumentation entspricht.

Import-Anweisung

- Fertige Programme werden in Verzeichnissen, den sogenannten Paketen gespeichert, z. B. die Klassen des Swing-Pakets `javax.swing` im Verzeichnis `/javax/swing`. Auch alle Programme der Java-API werden in Paketen bereitgestellt.
- Mehrere Pakete des gleichen Themenbereiches werden oft noch einmal gebündelt in `.jar`- oder `.zip`-Dateien.
- Um eine Klasse oder alle Klassen eines Pakets im eigenen Programm nutzen zu können, formuliert man an den Anfang des Programms die Anweisung

```
import javax.swing.*;
```


String und Ausgabe

- Der in die Anführungszeichen (") gesetzte Text ist ein String, also eine Kette von beliebigen Zeichen. In Java wird ein String stets durch die doppelten Anführungszeichen eingeschlossen, also

"... ein Text "

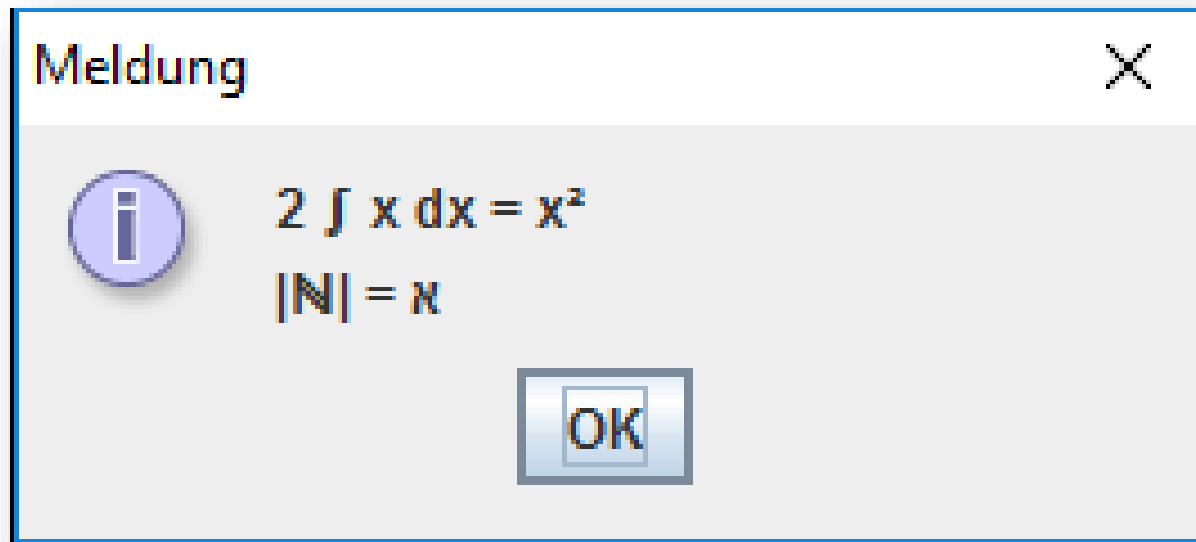
- Innerhalb der Anführungszeichen kann ein beliebiges Unicode-Zeichen stehen (praktisch alles, was die Tastatur hergibt . . .), also insbesondere Leerzeichen, aber auch „Escape-Sequenzen“ wie das `\n` für Zeilenumbruch.

Escape-Zeichen	Bedeutung	Beschreibung
<code>\uxxxx</code>	Unicode	das Unicode-Zeichen mit Hexadezimal-Code <i>xxxx</i> („ <i>Code-Point</i> “); z.B. <code>\u222B</code> ergibt ∫
<code>\n</code>	line feed LF	neue Zeile. Der Bildschirmcursor springt an den Anfang der nächsten Zeile
<code>\t</code>	horizontal tab HT	führt einen Tabulatorsprung aus
<code>\\</code>	<code>\</code>	Backslash <code>\</code>
<code>\"</code>	<code>"</code>	Anführungszeichen <code>"</code>
<code>\'</code>	<code>'</code>	Hochkomma (Apostroph) <code>'</code>

Escape-Sequenz

- Folgenden Code im Dialogfenster anzeigen lassen

```
JOptionPane.showMessageDialog(null, "2 ∫ x dx = x²\n|N| = x");
```



String Ausgabe auf der Konsole

- Ausgaben auf der Konsole werden mit folgenden Befehlen erzeugt:

Ohne Zeilenumbruch nach der Ausgabe

```
System.out.print("Hallo, GFN");
```

```
System.out.print("-Umschüler");
```

Mit Zeilenumbruch nach der Ausgabe (`println` gesprochen print line)

```
System.out.println(" und Umschülerinnen");
```

Ergibt die Anzeige

```
Hallo, GFN-Umschüler und Umschülerinnen
```

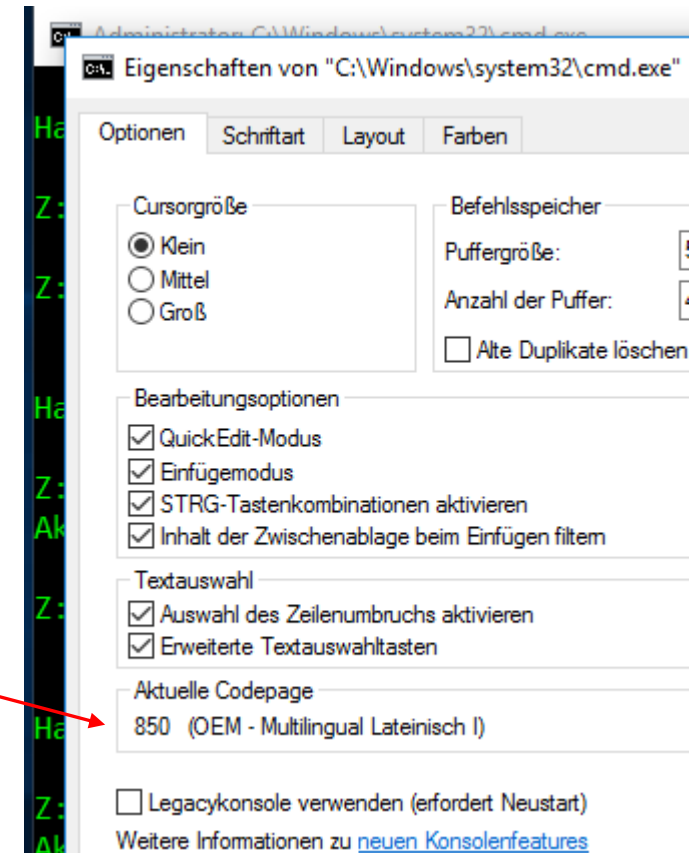
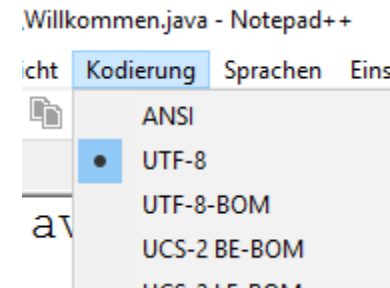
Seltsame Anzeige

- Eventuell werden die Umlaute nicht richtig angezeigt, z. B.



- Der Grund ist die Zeichenkodierung
- Der Editor, NotePad++ hat die Einstellung UTF-8
- Die Konsole interpretiert Zeichen des String als ANSI

- Lösung:
in der Konsole den Befehl `chcp 65001` eingeben
oder
in Notepad++ Menü Kodierung – Konvertiere zu ANSI
Datei speichern und nochmal mit Chip `javac` kompilieren
oder
Die Kodierung in Notepad++ auf UTF-8 belassen und die Ausgabe nicht über `System.out` sondern mit `JOptionPane.showMessageDialog` anzeigen lassen



Kompetenzcheck



- a. Mit welchem Befehl wird der Java-Compiler aufgerufen?
- b. Wie lautet der gesamte Befehl?
- c. Was ist bei Dateinamen in der Java-Programmierung zu beachten?
- d. Welche Dateinamensendung haben die kompilierten Java-Programme?
- e. Mit welchem Befehl wird eine Java-Datei zur Ausführung aufgerufen?
- f. Welche Methode muss in der Klasse kodiert sein, damit sie ausführbar ist?
- g. Mit welchem Code werden Ausgaben auf der Konsole ausgegeben?
- h. Was bewirkt die import-Anweisung?
- i. An welcher Position muss die import-Anweisung kodiert werden?

Mit NetBeans Programme entwickeln

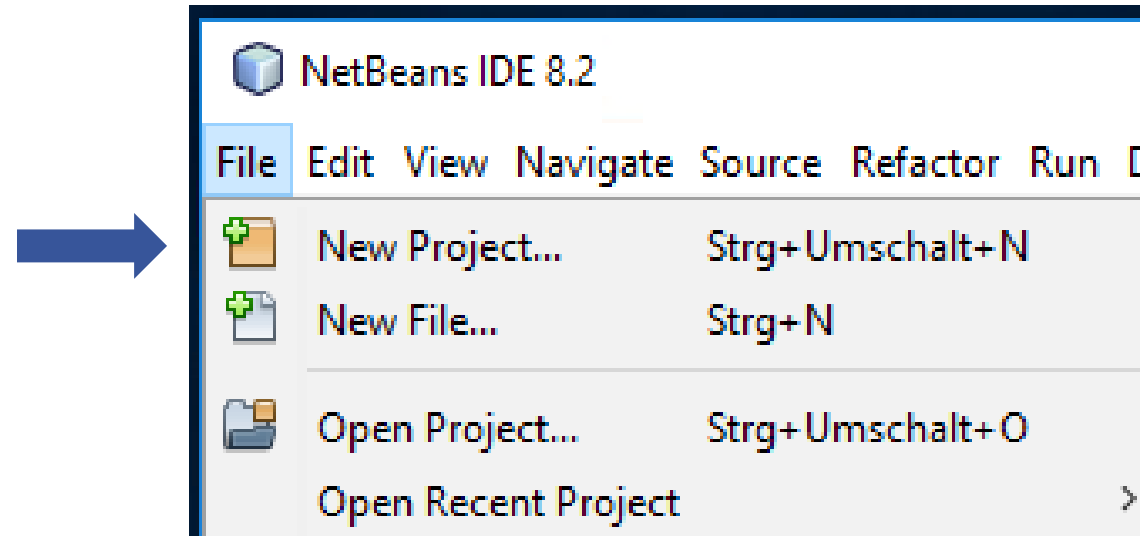
- Nun, endlich, benutzen wir die IDE NetBeans
- Starten Sie NetBeans
- Ändern Sie vorerst keine Einstellungen, wenn Sie mit der Oberfläche und dem Projektkonzept von NetBeans noch nicht vertraut sind
- Sie lernen in den folgenden Folien die Einstellungen für das Projekt kennen
- Sie erfahren, wo NetBeans die Dateien in Ihrem System ablegt
- Sie lernen, wie man ein Projekt von dem/r Trainer/-in oder eines Mitschülers übernehmen kann und in seiner eigenen Verzeichnisstruktur ablegt
- Sie schaffen Ordnung auf Ihrem System für die vielen Projekte und Dateien, die Sie in Ihrer Ausbildung noch programmieren und testen werden

NetBeans

- In NetBeans wird jedes Programm, auch ein „HelloWorld.java“ in einem Projekt verwaltet
So gibt es die Möglichkeit, in einem Themengebiet mehrere Programme oder Programmteile besser zu verwalten und auch wiederzufinden.
- Das Basis-Dateiverzeichnis für alle Projekte sollten Sie sich gut überlegen, damit Sie die Dateien auf Ihrem System schnell finden. Beispiel:
`username\LF11a\NB`
- Wenn Sie ein neues Projekt mit „New Project“ in NetBeans anlegen, soll der Projektname als Verzeichnis in Ihrem Basis-Dateiverzeichnis erscheinen
- Innerhalb eines Projektes gibt es immer ein Paket, das in der NetBeans Anzeige **Source Packages** heißt, und als Ordner **src** im Projektordner zu sehen ist
- Hier können Sie beim Anlegen des Projektes eigene Packages bei freier Namenswahl erstellen, - jedoch ist Kleinschreibung üblich –
die als Unterverzeichnis im Ordner **src** erscheinen
- In diesem von Ihnen erstellten Package werden die Klassen; Name.java gespeichert

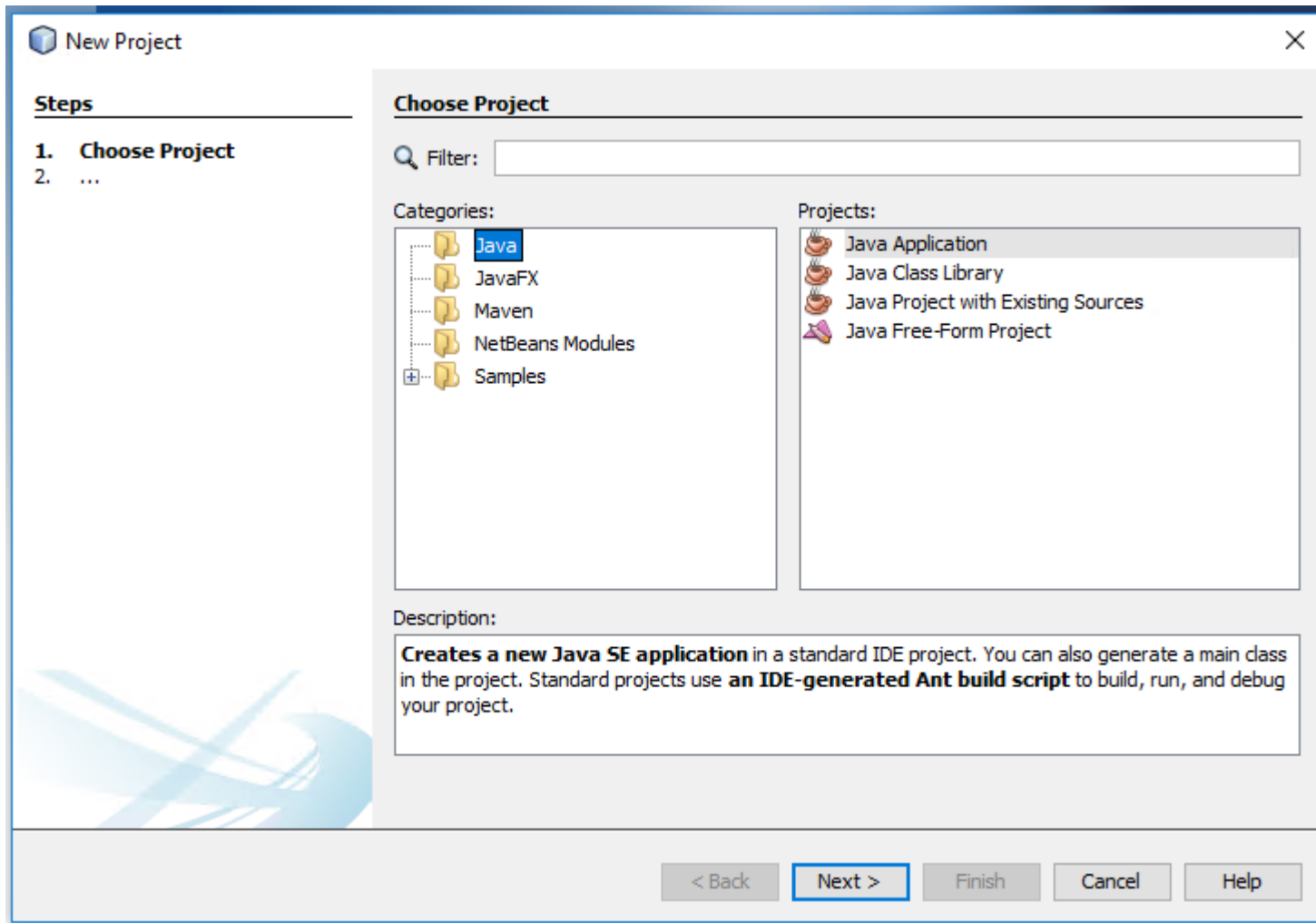
Neues Projekt in NetBeans (1)

- Ein Verzeichnis [LF11a](#) habe ich bereits in meinem [Benutzer-Verzeichnis\Dokumente](#) angelegt
- In NetBeans wähle ich aus dem Menü [File - New Project...](#)



- Es erscheint ein Fenster

Neues Projekt in NetBeans (2)



Categorie: **Java**

Projects: **Java Application**



Next klicken

Neues Projekt in NetBeans (3)

New Java Application

Steps

1. Choose Project
2. **Name and Location**

Name and Location

Project Name:

Project Location:

Project Folder:

☐ Use Dedicated Folder for Storing Libraries

Libraries Folder:

Different users and projects can share the same compilation libraries (see Help for details).

☒ Create Main Class

< Back Next > Finish Cancel Help

1. **Project Name:** ändern

2. **Project Location:** ändern
im Textfeld
oder komfortabel
mit Button [**Browse...**]

4. **Package Name:**
den Text vor dem Punkt
ändern

3. Auswahl Create Main Class

**Zuletzt
alles kontrollieren
und [Finish] klicken**

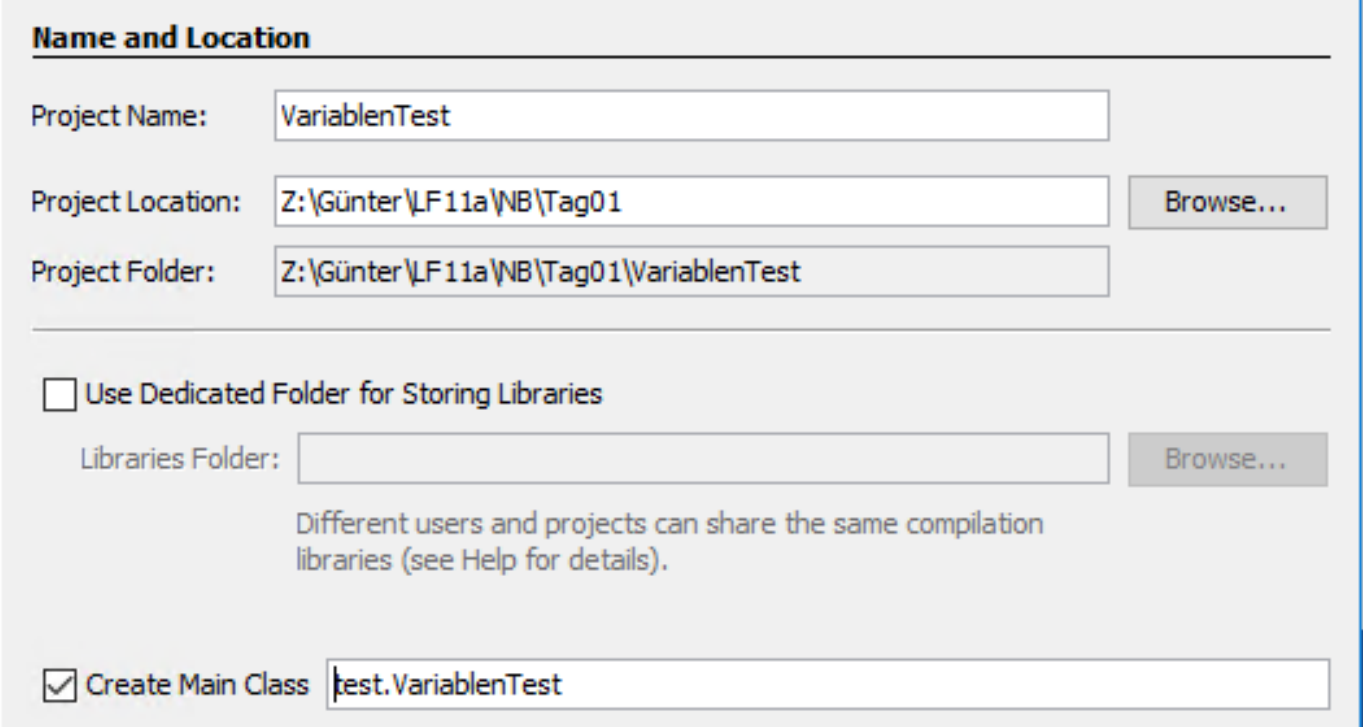
Neues Projekt in NetBeans (4)

Meine Eingaben:

Project Name: VariablenTest

Project Location: Z:\Günter\LF11a\NB\Tag01

Create Main Class: test.VariablenTest



The image shows the 'New Project' dialog box in NetBeans, specifically the 'Name and Location' tab. The dialog is titled 'Name and Location' and contains several input fields and checkboxes. The 'Project Name' field is filled with 'VariablenTest'. The 'Project Location' field is filled with 'Z:\Günter\LF11a\NB\Tag01', and there is a 'Browse...' button next to it. The 'Project Folder' field is filled with 'Z:\Günter\LF11a\NB\Tag01\VariablenTest'. Below these fields, there is a checkbox labeled 'Use Dedicated Folder for Storing Libraries' which is currently unchecked. Next to it is a 'Libraries Folder' field and another 'Browse...' button. Below the 'Libraries Folder' field, there is a note: 'Different users and projects can share the same compilation libraries (see Help for details)'. At the bottom, there is a checkbox labeled 'Create Main Class' which is checked, and next to it is a text field containing 'test.VariablenTest'.

Name and Location

Project Name: VariablenTest

Project Location: Z:\Günter\LF11a\NB\Tag01 Browse...

Project Folder: Z:\Günter\LF11a\NB\Tag01\VariablenTest

☐ Use Dedicated Folder for Storing Libraries

Libraries Folder: Browse...

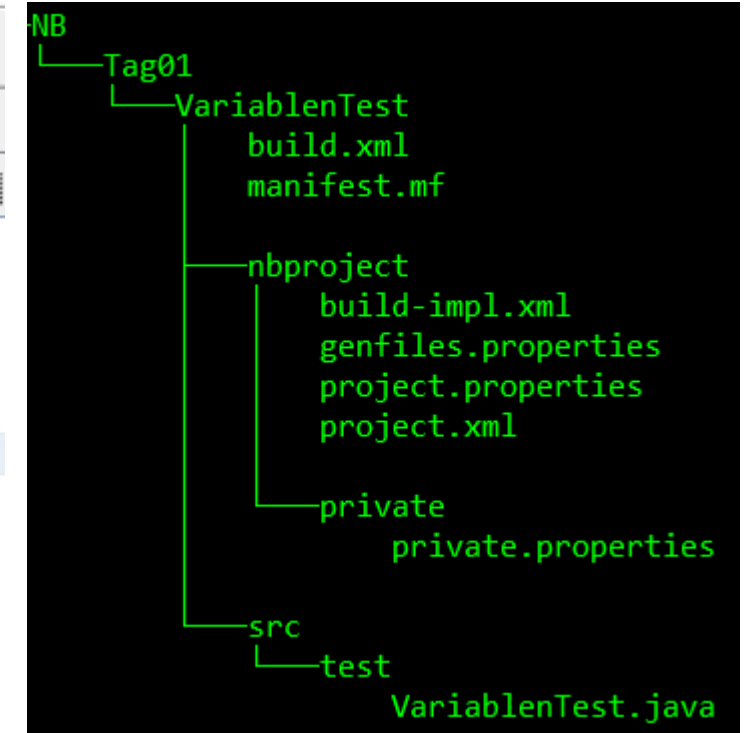
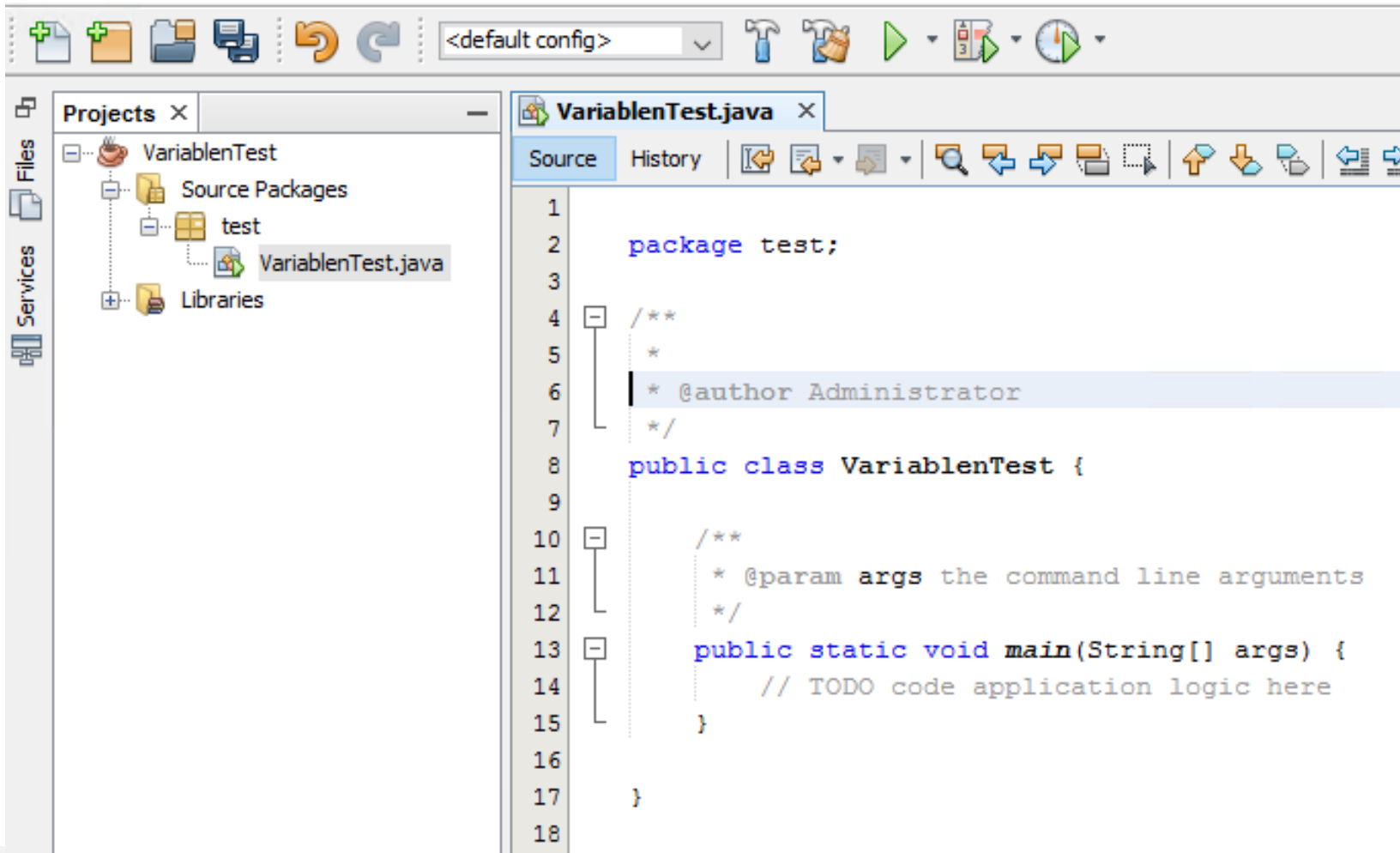
Different users and projects can share the same compilation libraries (see Help for details).

☒ Create Main Class test.VariablenTest

Neues Projekt in NetBeans (5)

VariablenTest - NetBeans IDE 8.2

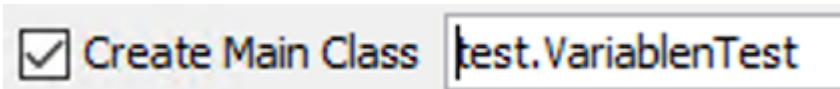
File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help



Ordner und Dateien im System

NetBeans Features

- NetBeans generiert beim Anlegen eines Projektes die Grundstruktur der sogenannten **Main Klasse**. Obwohl es sich nicht um eine **class Main** handelt, wird diese Klasse, die Sie beim Anlegen mit



definierten, als Einstiegspunkt für die virtuelle Maschine genutzt, weil sie die statische main-**Methode** enthält

- Die erste Zeile ist das **package** (hier test), das Sie mit Punkt und Klassennamen festgelegt haben
- Es gibt immer nur **eine package** Anweisung in einer Java-Datei
- Import Anweisungen - es können mehrere sein - werden nach der package-Anweisung kodiert
- Es folgt ein Dokumentationskommentar **/** */** vor der Klassendeklaration
- Auch vor der Methodendeklaration wurde ein Dokumentationskommentar **/** */** generiert
- Beachten Sie auch die korrekten Einrückungen, die Tabulatorsprünge
- Sie können Ihren Code auch mit dem Menüpunkt **Source - Format** automatisch formatieren lassen



Übungen

Schreiben Sie in den Anweisungsblock der main-Methode etwas Java-Code, den Sie schon kennen, z. B.:

```
System.out.println( ...
```

```
JOptionPane.showMessageDialog( ...
```

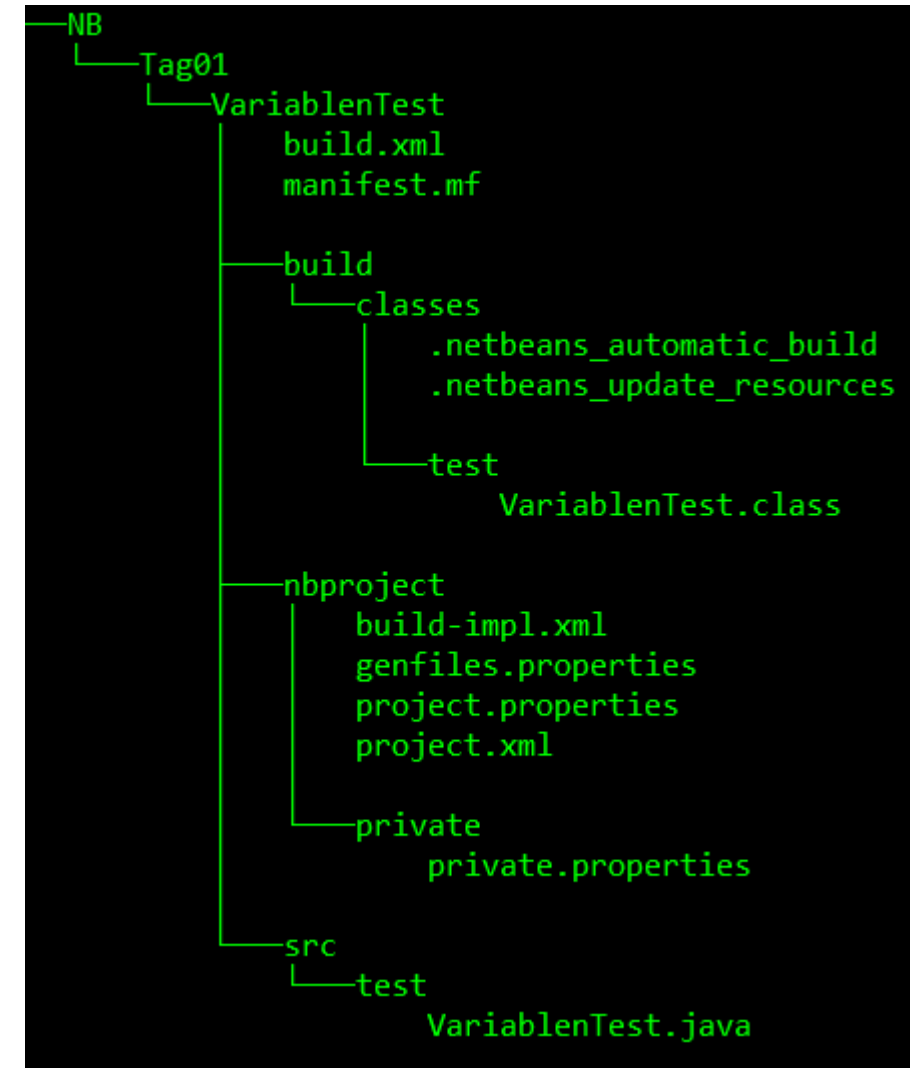
Den Kommentar `// TODO code application logic here` können Sie überschreiben

Beobachten Sie mögliche Hilfeangebote von NetBeans

Starten sie das Programm mit dem grünen Pfeil oder mit [F6]

Projekt Ordner- und Dateistruktur

- Wir betrachten noch einmal die Ordner und Dateien im System
- Es ist ein neuer Ordner **build** entstanden. Dieser entsteht durch das Starten des Programms, wobei NetBeans das Projekt kompiliert und die entstandene **.class** Datei mit dem **package** als Ordner speichert.



Was folgt?



**Weniger
PowerPoint**



Viele Übungen

- Datentypen
- Operatoren
- Logik
- Kontrollstrukturen
- Klassen/Objekte
- Methode
- Vererbung
- Polymorphie
- uvm.



neue Erkenntnisse