

PSE - Vorkurs

Tobias, Philipp, Linus, Tillmann

FIUS - Fachgruppe Informatik Universität Stuttgart

13. Juli 2025



Besser machen	11
Neu machen ist ultra hässlich	19
Neu machen ist ultra hässlich	20

Die Universität bietet ein Gast WLAN an jedoch wollen wir in das offizielle WLAN der Universität einsteigen.

- ▶ iOS und Android
- ▶ Windows
- ▶ MacOS
- ▶ Linux

WLAN - iOS und Android

- ▶ Im AppStore bzw. Playstore nach „geteduroam“ suchen und die App installieren
- ▶ Dann nach Universität Stuttgart suchen
- ▶ Nun Student wählen und eure Daten eingeben
- ▶ `stxxxxxx@stud.uni-stuttgart.de` - Benutzername
- ▶ Und eurer Passwort eingeben
- ▶ Nun auf Verbinden klicken und fertig



WLAN - Windows

- ▶ Im Internet auf „geteduroam.app“ gehen und die Windows Version herunterladen
- ▶ Dann nach Universität Stuttgart suchen
- ▶ Nun Student wählen und eure Daten eingeben
- ▶ Nun auf Verbinden klicken und fertig



- ▶ Im Internet auf „uni-stuttgart.de/eduroam“ gehen
- ▶ und als Gruppe „Student“ wählen
- ▶ Lädt das Konfigurationspaket herunter
- ▶ In euren Systemeinstellungen auf „Geräteverwaltung“ gehen
- ▶ Nun über das Plus-Symbol das Konfigurationspaket hinzufügen
- ▶ stxxxxxx - Benutzername
- ▶ Und euer Passwort eingeben
- ▶ Nun auf Verbinden klicken und fertig

- ▶ Im Internet auf „uni-stuttgart.de/eduroam“ gehen
- ▶ Als Benutzergruppe „Student“ wählen
- ▶ Unten links auf „einen anderen Installer auswählen“ klicken
- ▶ „Linux“ auswählen
- ▶ Über den Button das Pythonskript herunterladen
- ▶ Im Terminal mit `$ python3 [Dateipfad zum Pythonskript]` ausführen
- ▶ Den Anweisungen folgen und Benutzername
(stxxxxxx@stud.uni-stuttgart.de) sowie Passwort eingeben

⚠ Achtung: Wir werden uns Lediglich auf IntelliJ IDEA konzentrieren, wenn ihr eine andere IDE nutzen wollt, können wir keinen Support garantieren.

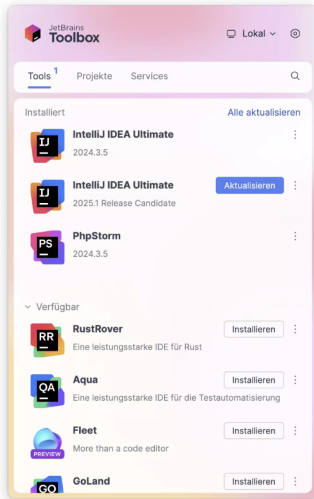
- ▶ Wir laden uns erst die JetBrains Toolbox herunter, um die IDE zu installieren
- ▶ Dann installieren wir IntelliJ IDEA Community Edition



IntelliJ IDEA - Installation

[Andere Versionen](#)

Geht auf „jetbrains.com/toolbox-app/“ und ladet die Toolbox herunter.



JetBrains Toolbox App

**Eure Version
auswählen**

Verwalten Sie Ihre IDEs ohne Stress

Herunterladen

.dmg (macOS Apple Silicon) ▼

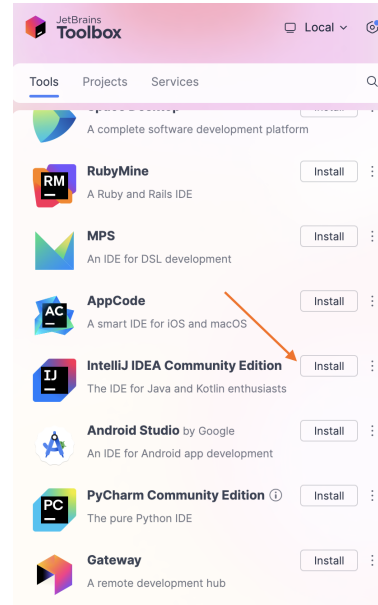
Es ist kostenlos. [Systemanforderungen](#)



IntelliJ IDEA - Installation


Nach der Installation der Toolbox öffnet sich ein Fenster, in dem ihr die IDE auswählen könnt, die ihr installieren wollt. Wählt „IntelliJ IDEA Community Edition“ aus und klickt auf „Install“.

Besser machen



JDK - Installation

Wir benötigen ein JDK (Java Development Kit), um Java Programme zu schreiben und auszuführen.

- ▶  Achtung: MacOS: auf File/ Datei gehen oben links in der Menüleiste
- ▶ Nun auf Project Structure klicken
- ▶ Dann auf SDKs klicken
- ▶ Dann auf das Plus-Symbol klicken und auf Download JDK klicken

⚠ Achtung: Die Universität nutzt die OpenJDK Version 21, die wir auch verwenden werden.

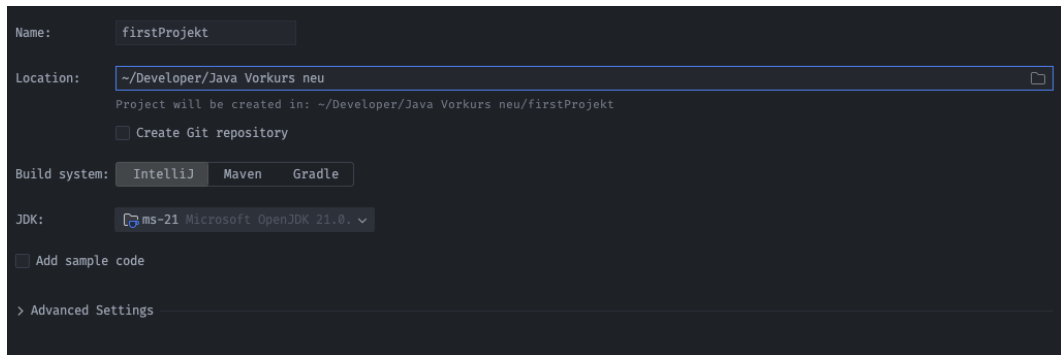
- ▶ Dann Microsoft OpenJDK als Anbieter/ Vendor auswählen
- ▶ Und nen geeigneten Speicher Ort auswählen, der Standart Pfad sollte aber ausreichen.

Projekt erstellen

- ▶ In IntelliJ auf neues Projekt erstellen/ New Project klicken
- ▶ Java auswählen und das Projekt firstProjekt nennen
- ▶ Nun einen geeigneten Speicher Ort auswählen ⚠ Achtung: KEIN Git Repository erstellen
- ▶ Als Build System IntelliJ verwenden und als JDK unsere derzeitige heruntergeladene 21 auswählen
- ▶ Wir wollen keinen Sample Code

Projekt erstellen

Es sollte nun so bei euch aussehen:



The screenshot shows the 'New Project' dialog in IntelliJ IDEA with the following configuration:

- Name:** firstProjekt
- Location:** ~/Developer/Java Vorkurs neu (with a folder icon on the right)
- Project will be created in:** ~/Developer/Java Vorkurs neu/firstProjekt
- ☐ Create Git repository
- Build system:** IntelliJ (selected), Maven, Gradle
- JDK:** ms-21 Microsoft OpenJDK 21.0. (with a dropdown arrow)
- ☐ Add sample code
- [> Advanced Settings](#)

Projekt erstellen

- ▶ Nun auf `Create` klicken und das Projekt wird erstellt
- ▶ Nun links in der Projektansicht auf `src` rechtsklicken und eine neue Java Klasse erstellen
- ▶ Diese Klasse `Main` nennen und auf `Enter` drücken
- ▶ Nun sollte sich eine neue Datei öffnen, in der wir unseren Code schreiben

Erste Java Klasse

Nun schreiben wir folgenden Code in die Main Klasse rein

```
1 public class Main {  
2     public static void main(String[] args) {  
3  
4     }  
5 }
```

⚠ Achtung: Was das alles bedeutet werden wir euch dann in den nächsten Tagen erklären!

GitHub ist eine Plattform, auf der man Code online speichern, verwalten und mit anderen teilen kann. Es basiert auf Git, einem Versionskontrollsystem.

Typische Begriffe auf GitHub

Neu machen ist ultra hässlich

- ▶ **Repository (Repo):** Dein Projektordner auf GitHub.
- ▶ **Commit:** Eine gespeicherte Änderung am Code.
- ▶ **Push:** Hochladen deiner Änderungen auf GitHub.
- ▶ **Pull:** Änderungen von GitHub herunterladen.
- ▶ **Clone:** Ein Projekt von GitHub auf deinen Computer kopieren.

Warum GitHub für Java-Projekte?

Neu machen ist ultra hässlich

- ▶ **Sicherung:** Du kannst deinen Java-Code (Klassen, Methoden, Projekte) sicher speichern.
- ▶ **Versionskontrolle:** Du siehst, was sich zwischen Versionen geändert hat – praktisch beim Lernen oder bei Fehlern.
- ▶ **Zusammenarbeit:** Du kannst mit anderen zusammenarbeiten (Teamarbeit, Feedback).
- ▶ **Überall verfügbar:** Zugriff von überall – egal ob Uni, zuhause oder unterwegs.



Variablen

- ▶ Wir benötigen Variablen um Werte im Programm zwischenzuspeichern
- ▶ Variablen haben einen Namen (Bezeichner), einen Datentyp und speichern einen Wert
- ▶ Über den Namen kann später auf den gespeicherten Wert zugegriffen werden

```
1  int alter = 20;
```

- ▶ Hier ist `alter` der Name der Variable, `int` der Datentyp (ganzzahlige Zahl) und `20` der gespeicherte Wert

⚠ Achtung: Bezeichner müssen mit einem Buchstaben, Unterstrich (`_`) oder Dollarzeichen (`$`) beginnen. Danach dürfen beliebig viele Buchstaben, Ziffern, Unterstriche oder Dollarzeichen folgen – aber keine Leerzeichen oder Sonderzeichen.

Variablen

- ▶ Deklaration → Variable wird angelegt/erstellt **OHNE** einen Wert zuzuweisen

```
1  int zahl;
```

Variablen

- ▶ Deklaration → Variable wird angelegt/erstellt **OHNE** einen Wert zuzuweisen

```
1  int zahl;
```

- ▶ Initialisierung → Variable wird sein **ERSTER** Wert zugewiesen

```
1  zahl = 42;
```

Variablen

- ▶ Deklaration → Variable wird angelegt/erstellt **OHNE** einen Wert zuzuweisen

```
1  int zahl;
```

- ▶ Initialisierung → Variable wird sein **ERSTER** Wert zugewiesen

```
1  zahl = 42;
```

```
1  //In der Regel Deklaration und Initialisierung gleichzeitig:
2  int zahl = 42;
3
4  //aber auch so möglich
5  int zahl;
6  zahl = 42;
```


- ▶ Variablen können auch später im Programm verändert werden

```
1  int alter = 20;  
2  alter = 21; // alter ist jetzt 21
```

- ▶ Variablen können auch mehrfach deklariert werden, aber nur einmal initialisiert

```
1  int a = 5;  
2  int b = 10;  
3  int c = a + b; // c ist jetzt 15
```

⚠ Achtung: Variablen müssen vor ihrer Verwendung deklariert und initialisiert werden

Primitive Datentypen in Java

Typ	Beschreibung
int	Ganzzahlen (z. B. 1, 2, 3)
float	Kommazahlen (weniger genau)
double	Kommazahlen (hohe Genauigkeit)
char	Einzelnes Zeichen (z. B. 'a')
short	Kleine Ganzzahlen
long	Große Ganzzahlen
boolean	Wahrheitswert (true / false)

Referenztypen in Java

Typ	Beschreibung
String	Text (z. B. "Hallo")
Array	Liste von Werten eines Typs
Object	Allgemeiner Objekttyp

Operatoren in Java

Operator	Bedeutung	Gültige Typen
+	Addition / Verkettung	Zahlen, String
-	Subtraktion	Zahlen
*	Multiplikation	Zahlen
/	Division	Zahlen
%	Modulo (Rest)	int, long, etc.

Beispiel: String + Zahl

```
1 String s = "Ergebnis: ";  
2 int zahl = 5;  
3 System.out.println(s + zahl);
```

Beispiel: String + Zahl

```
1 String s = "Ergebnis: ";  
2 int zahl = 5;  
3 System.out.println(s + zahl);
```

Ergebnis: 5

Fehlertypen in Java

- ▶ **Syntaxfehler:** Code verletzt Java-Regeln (z. B. fehlendes ‘;’)
- ▶ **Laufzeitfehler:** Fehler während der Ausführung (z. B. Division durch 0)
- ▶ **Logikfehler:** Programm läuft, aber liefert falsche Ergebnisse

Syntaxfehler – Beispiele

```
1 // Fehlendes Semikolon
2 System.out.println("Hallo")
3
4 // Falsches Schlüsselwort
5 publik class Test {}
6
7 // Variable nicht deklariert
8 x = 5;
```


Laufzeitfehler – Beispiele

```
1 String s = null;  
2 s.length(); // NullPointerException  
3  
4 int x = 5 / 0; // ArithmeticException  
5  
6 int[] a = {1, 2};  
7 a[3]; // ArrayIndexOutOfBoundsException
```

Logikfehler – Beispiel

```
1 public int add(int x, int y) {  
2     return x - y; // Falsche Operation!  
3 }
```

Kommentare in Java

- ▶ `//` Einzeilige Kommentare
- ▶ `/*` Mehrzeilige Kommentare `*/`

```
1  float gewicht = 23.4; // Gewicht in kg
2
3  /*
4   Berechnet die Summe
5   zweier Zahlen
6   */
7  int summe = x + y;
```

Textausgabe in Java

```
1 System.out.print("Hallo ");  
2 System.out.println("Welt!");
```

Hallo Welt!

Ausgabe mit Variablen

```
1  int number = 42;  
2  System.out.println("Lieblingszahl: " + number);
```

Lieblingszahl: 42

Eingabe mit Scanner

```
1  import java.util.Scanner;
2
3  public class Main {
4      public static void main(String[] args) {
5          Scanner scanner = new Scanner(System.in);
6          int x = scanner.nextInt();
7          int y = scanner.nextInt();
8          System.out.println(x + y);
9      }
10 }
```

Hinweis: `scanner.nextInt()` liest eine Ganzzahl von der Konsole.