hochschule mannheim

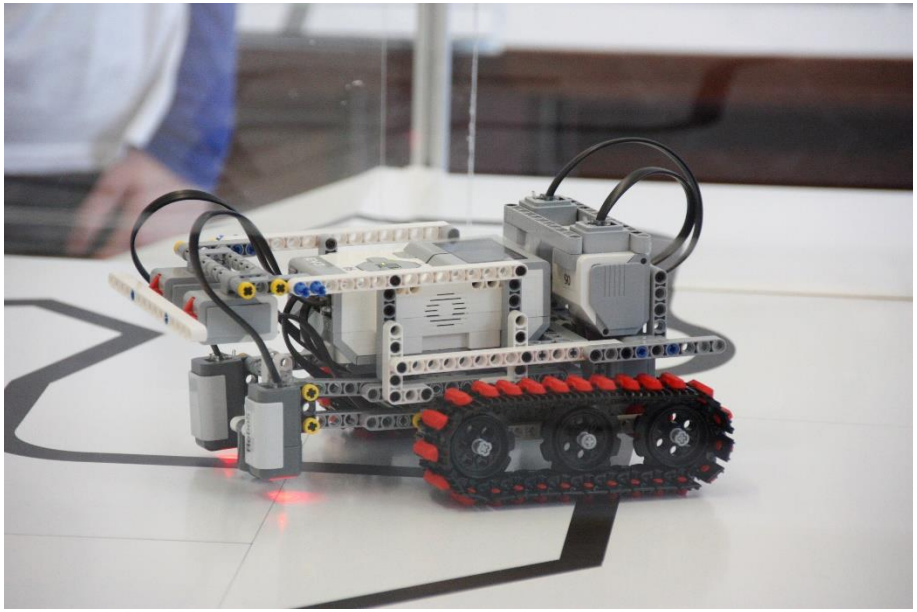
Fakultät für Informatik, Institut für Robotik

EV3 - Laborpraktikum I
Einführung in die Programmierung mit JAVA

Ute Ihme

DAS LEGO® MINDSTORMS® System

Das EV3 System



Prinzip von LEGO® MINDSTORMS®

- Roboter wird gebaut mit
  - programmierbarem LEGO® Stein
  - bis zu 4 Motoren oder Lampen
  - bis zu 4 Sensoren
  - LEGO® TECHNIC Teile
- Erstellung eines Steuerprogramms am Computer
- Übertragen des Programms auf den Roboter
- Testen des Programms

DAS LEGO® MINDSTORMS® System

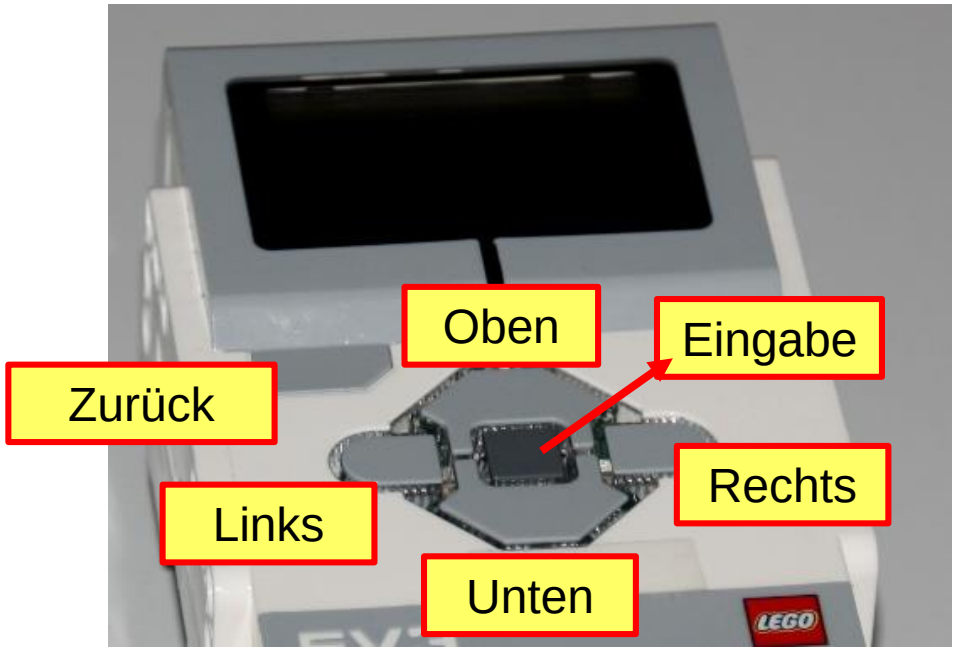
Der EV3 Stein

Motoren werden an die **Anschlüsse A, B, C und D** angeschlossen.

Sensoren werden an die **Anschlüsse 1, 2, 3 und 4** angeschlossen.

DAS LEGO® MINDSTORMS® System

Der EV3 Stein – Bezeichnung der Buttons



DAS LEGO® MINDSTORMS® System

Motoren

Quelle: Lego

Motoren werden an die **Anschlüsse A, B, C und D** angeschlossen.

Servomotor

- Verfügt über integrierten **Rotationssensor**
  - misst Geschwindigkeit und Abstand
  - Leitet Ergebnisse an NXT Stein weiter
- Motor kann auf einen Grad genau gesteuert werden
- Kombinationen mehrerer Motoren möglich
  - arbeiten ggf. mit gleicher Geschwindigkeit

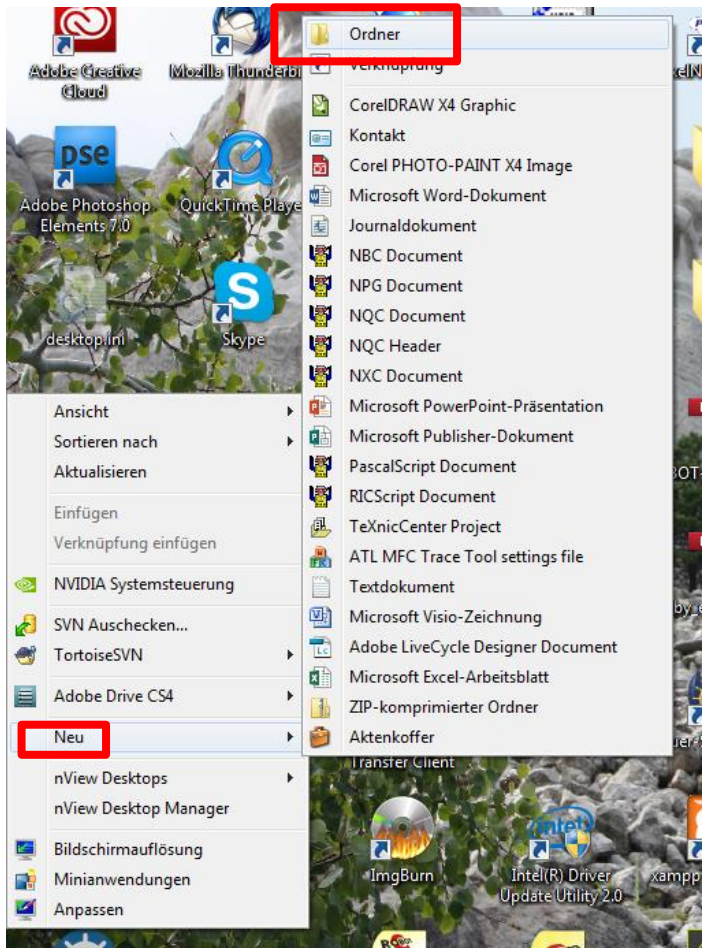
Start der Entwicklungsumgebung

Starten von Eclipse

Soll nicht im Standardverzeichnis gearbeitet werden:
Erstellen eines neuen Directories für Lego-Java-Programme

Im Ordner:
Lokaler Datenträger/Benutzer/IhrKonto

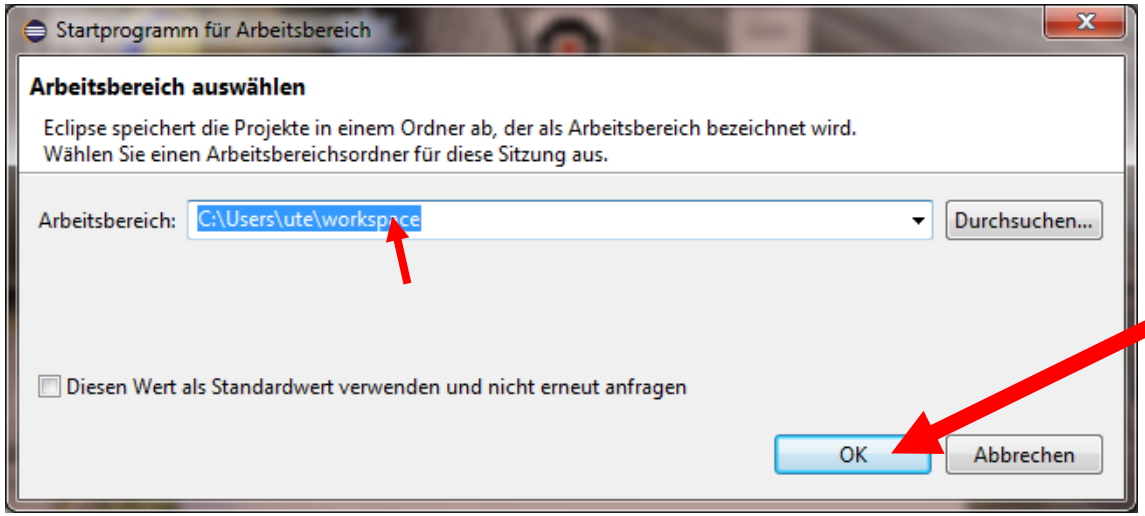
Ordnername: workspace_Lego



Start der Entwicklungsumgebung

Starten von Eclipse

1. Starten der VM aus bwLehrpool
2. Starten von Eclipse
3. Auswahl des Arbeitsbereiches
   Standardeinstellungen übernehmen oder
   ggf. Directory wechseln

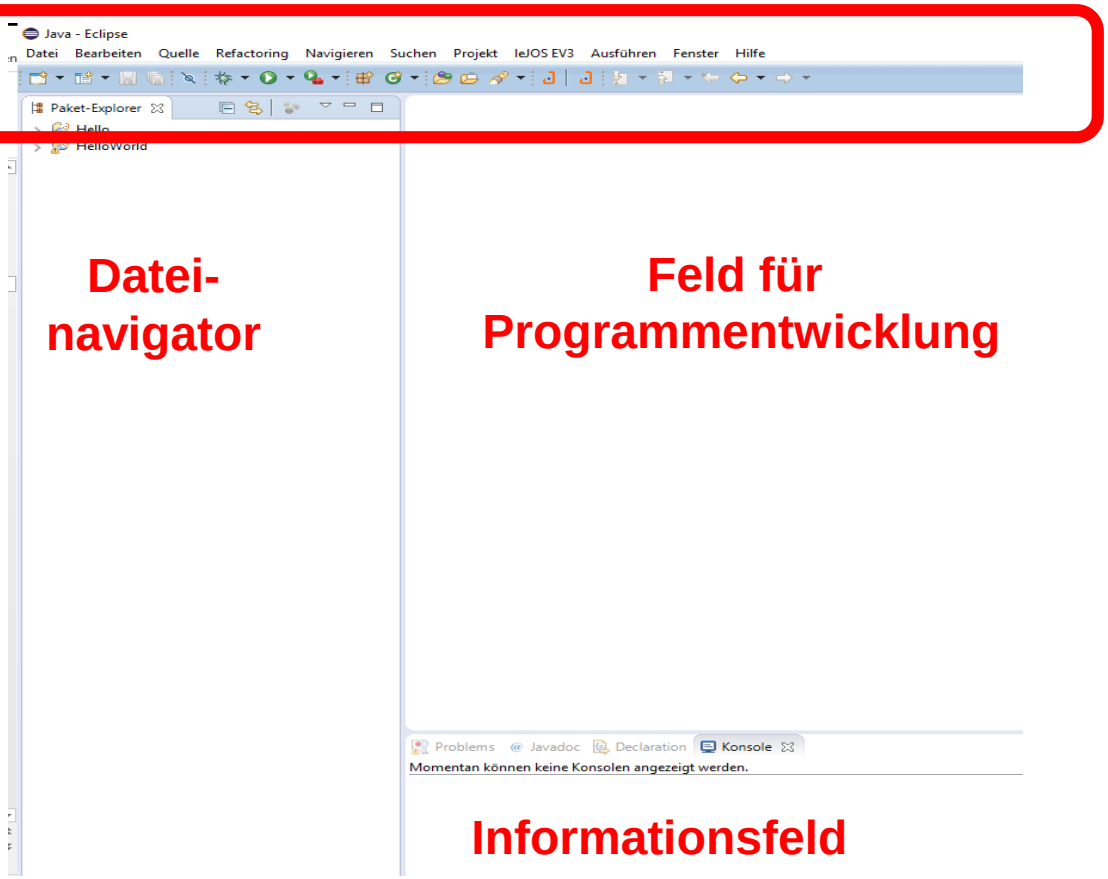


auf OK-Button drücken

Start der Entwicklungsumgebung

Starten von Eclipse

Arbeitsfläche von Eclipse



Navigationsleiste

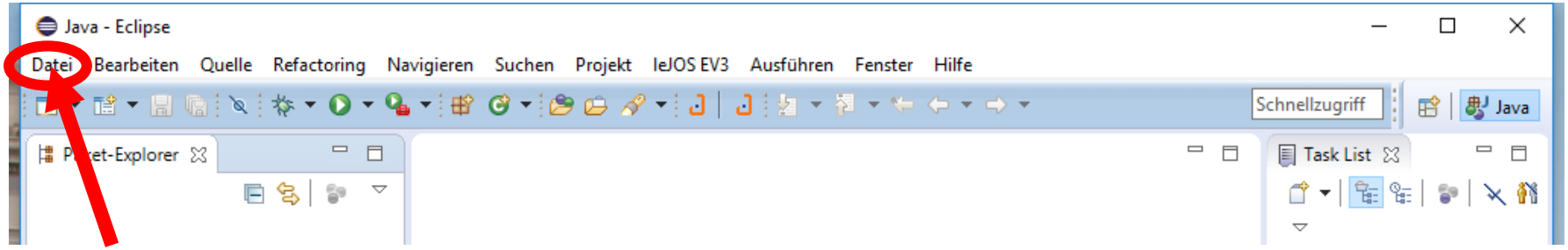
Datei-navigator

Feld für Programmentwicklung

Informationsfeld

Start der Entwicklungsumgebung

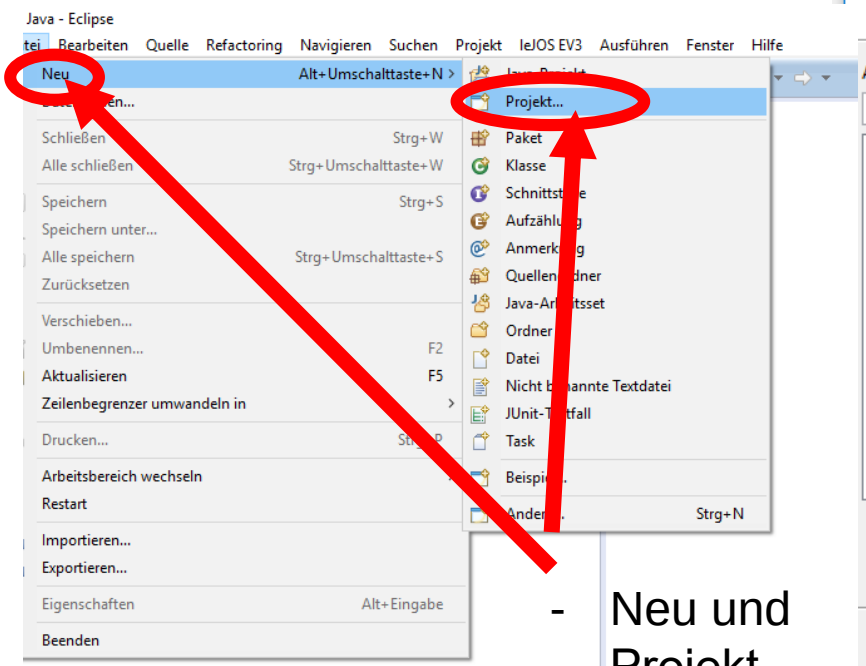
Erstellen von Projekten und Klassen



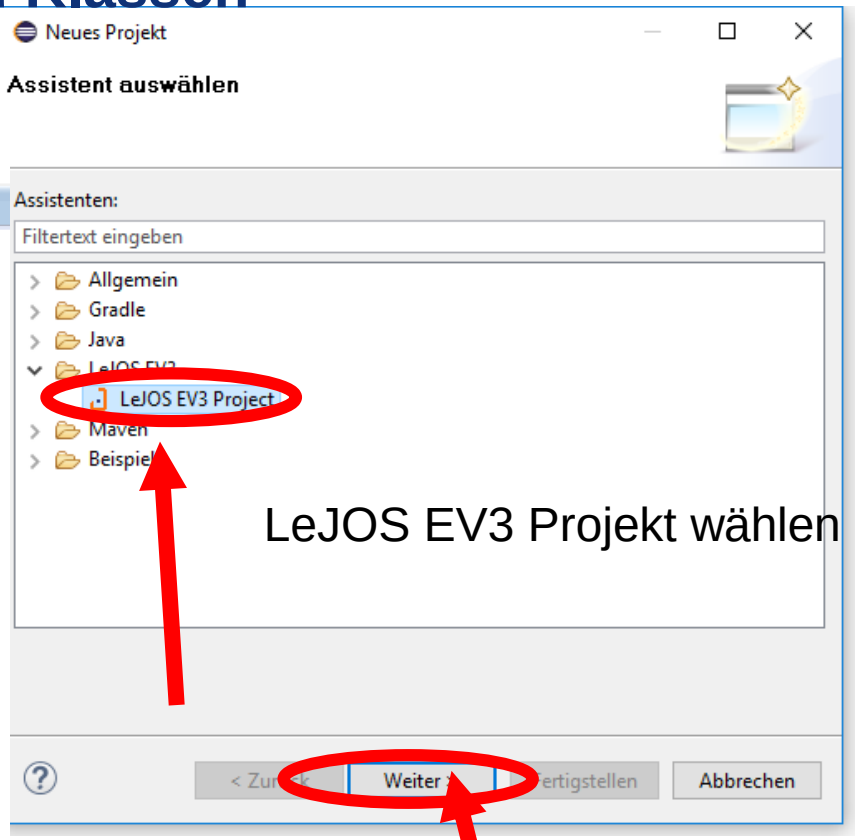
Datei wählen

Start der Entwicklungsumgebung

Erstellen von Projekten und Klassen



- Neu und
- Projekt… auswählen

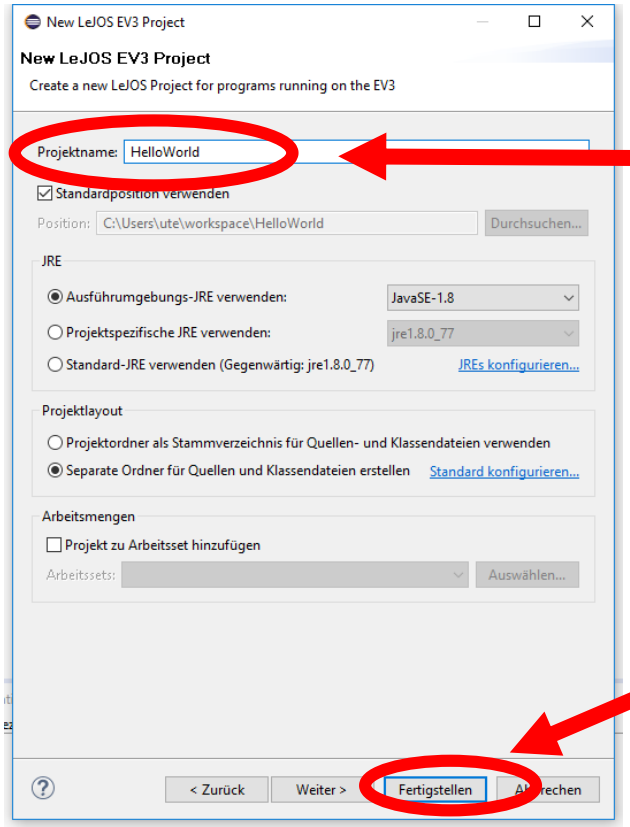


LeJOS EV3 Projekt wählen

auf Weiter> drücken

Start der Entwicklungsumgebung

Erstellen von Projekten und Klassen

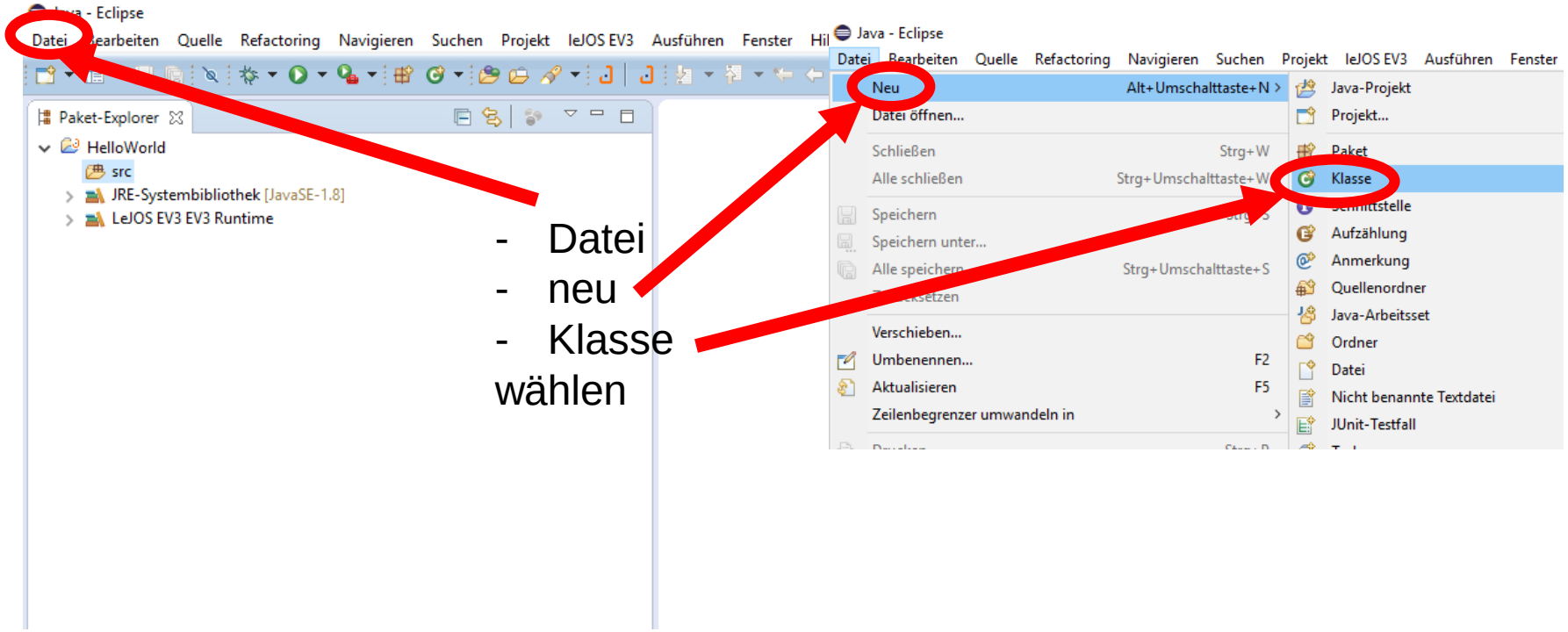


Projektname eingeben

Fertigstellen drücken

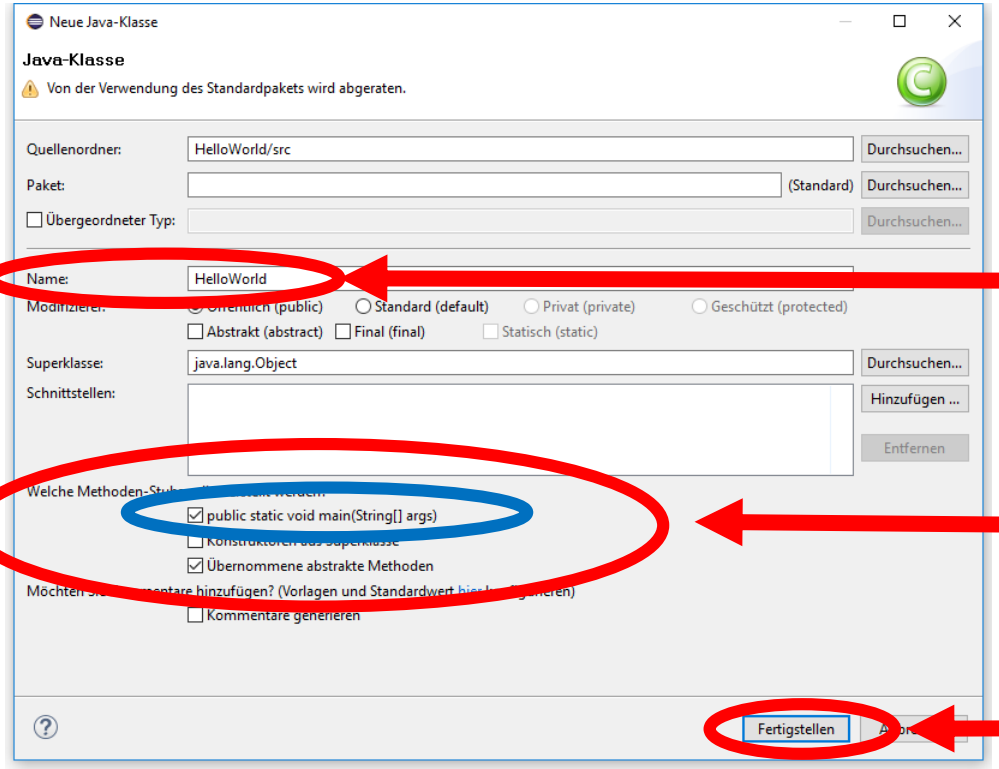
Start der Entwicklungsumgebung

Erstellen von Projekten und Klassen



Start der Entwicklungsumgebung

Erstellen von Projekten und Klassen



1. Klassennamen eingeben

2. Hier zusätzlich **public static void main** ankreuzen, wenn Klasse main Methode enthalten soll.

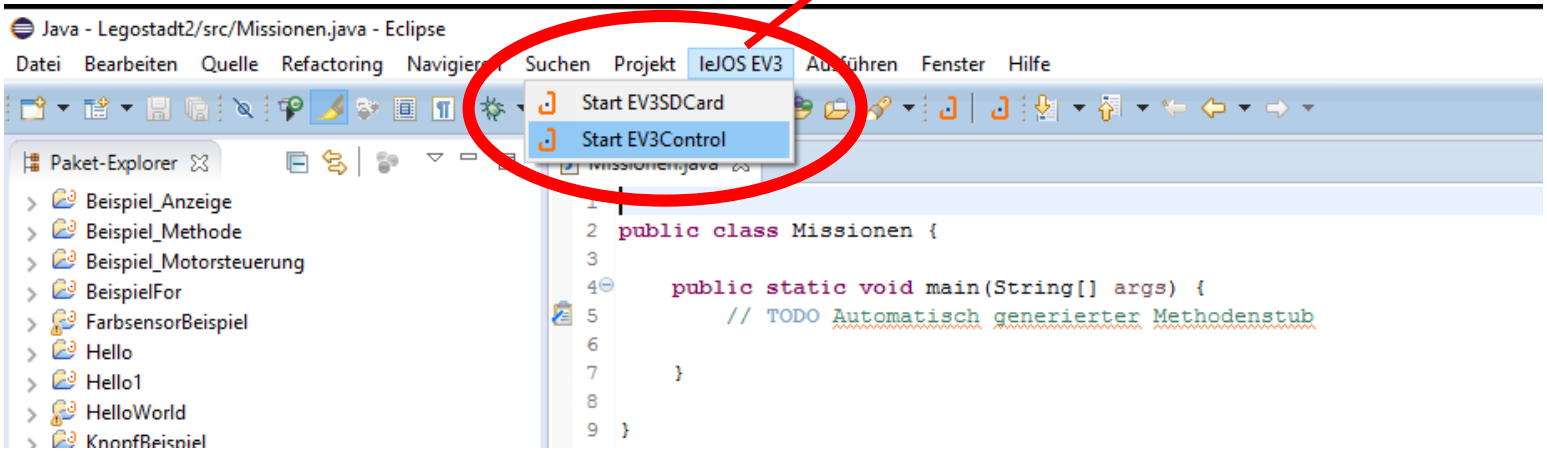
3. Fertigstellen drücken

Arbeit mit dem EV3 Stein

Verbindung mit dem EV3 Stein herstellen / Überprüfen

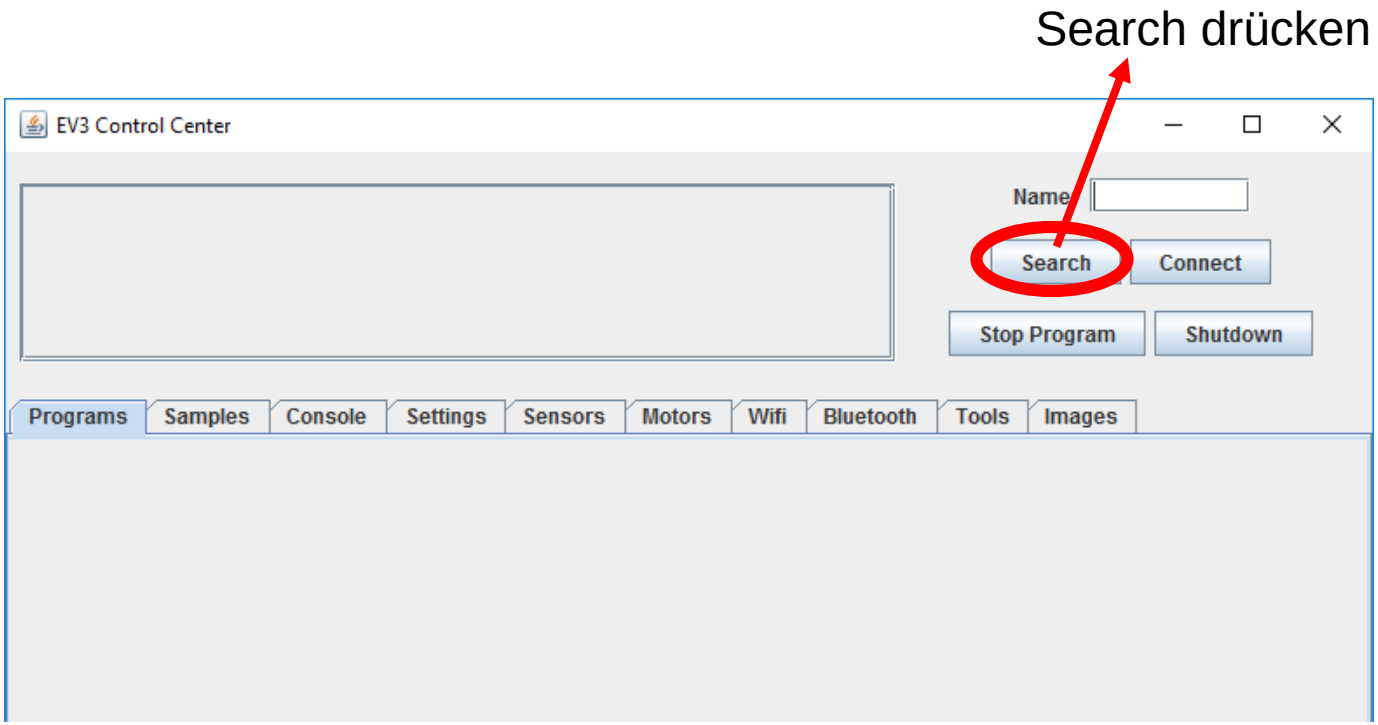
Start des EV3 ControlCenters

1. leJOS EV3 wählen
2. Start EV3 Control



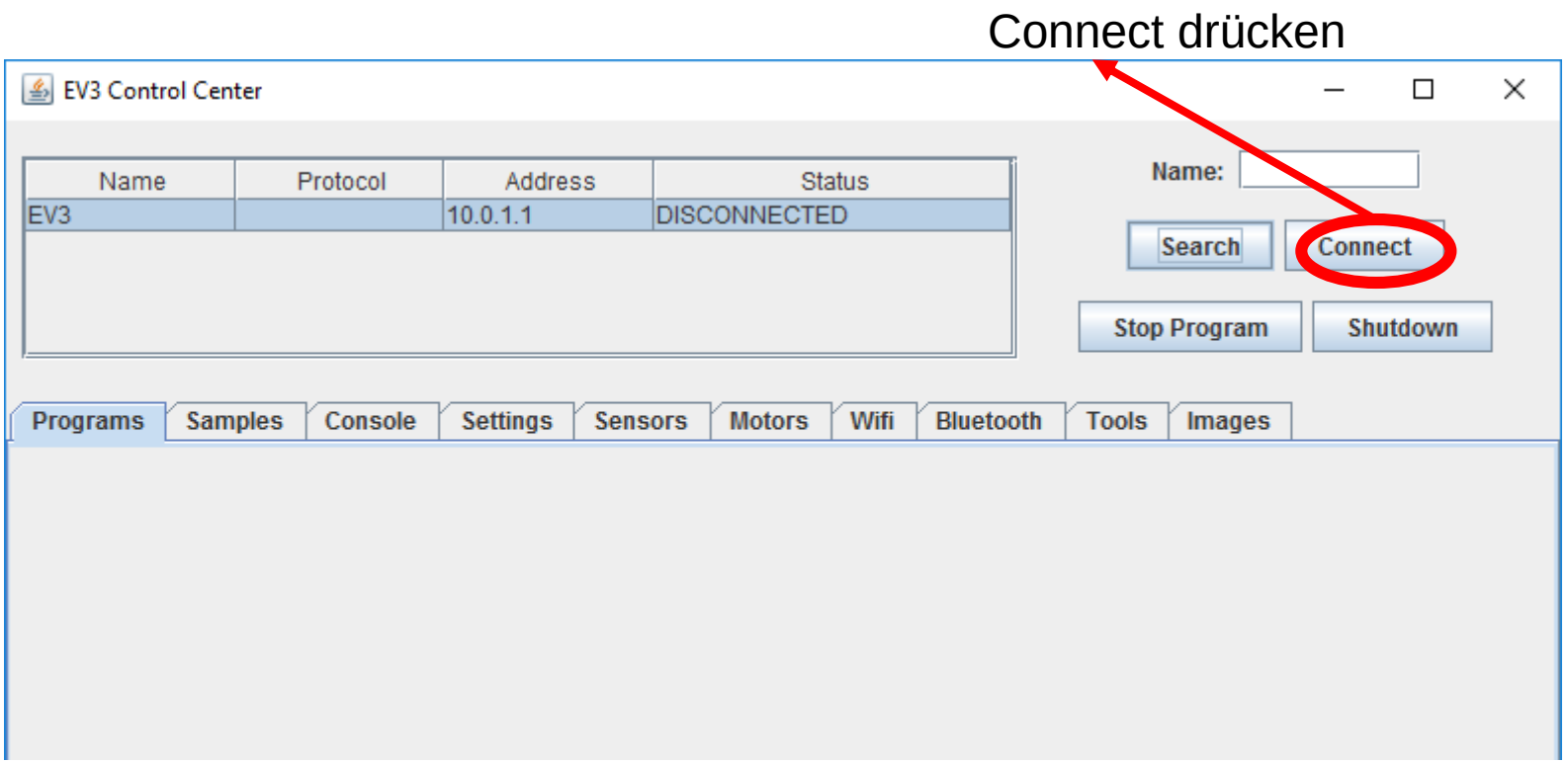
Arbeit mit dem EV3 Stein

Verbindung mit dem EV3 Stein herstellen / Überprüfen



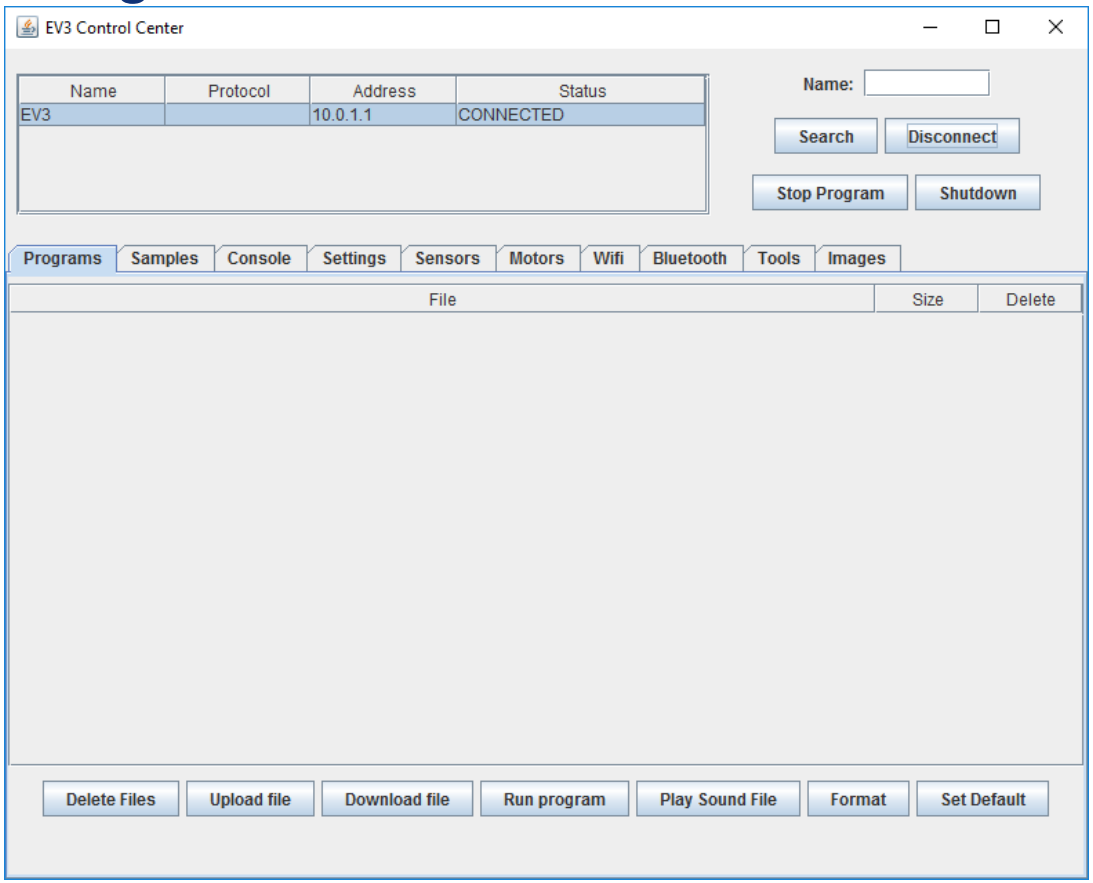
Arbeit mit dem EV3 Stein

Verbindung mit dem EV3 Stein herstellen / Überprüfen



Arbeit mit dem EV3 Stein

Verbindung mit dem EV3 Stein herstellen / Überprüfen



JAVA Code

Projekten und Klassen

- Jedes JAVA Programm besteht aus Klassen.

```
public class Berechnung {
    // hier wird der Programmcode eingefügt
}
```

- Eine der Klassen muss eine **main Methode** besitzen.
  Nur eine Klasse im Projekt darf eine **main Methode** besitzen.

```
public class Berechnung {
    public static void main(String[] args) {
        // hier wird der Programmcode eingefügt
    }
}
```

Erste Schritte: Bildschirmanzeigen

1. Nutzung des Standard JAVA Befehls

```
System.out.println("Hello World");
```

2. Nutzung des lejos Befehls
   a) für Strings `LCD.drawString(String, Spalte, Zeile);`

```
LCD.drawString("Hello Friend", 0, 1);
```

   b) Für Zahlen `LCD.drawInt(zahl, Spalte, Zeile);`

```
LCD.drawInt(7, 0, 2);
```

Für die Nutzung der lejos LCD Befehl ist folgende import-Funktion notwendig:

```
import lejos.hardware.lcd.LCD;
```

Erste Schritte: Bildschirmanzeigen

3. Löschen des Displays

```
LCD.clearDisplay();
```

Erste Schritte: Pausenbefehle

1. Warten darauf, dass ein Knopf des EV3 Steins gedrückt wird

```
Button.waitForAnyPress();
```

Für die Nutzung dieses lejos Befehls wird die import-Funktion benötigt:

```
import lejos.hardware.Button;
```

2. Nutzung eines leJos Pausen – Befehls: msDelay

```
Delay.msDelay(1000);
```

Für die Nutzung dieses lejos Befehls wird die import-Funktion benötigt:

```
import lejos.utility.Delay;
```

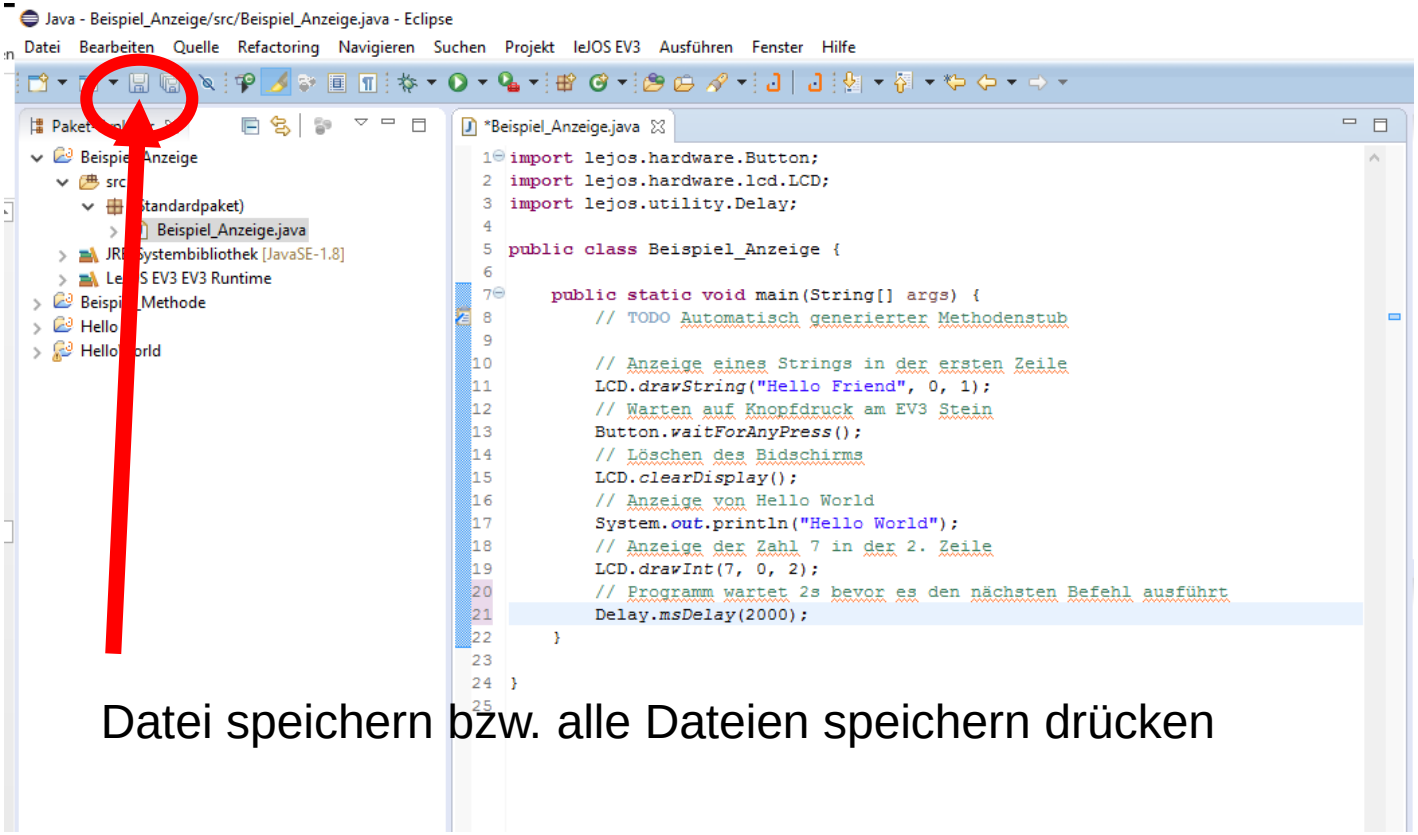
Erste Schritte: Beispielprogramm

```
import lejos.hardware.Button;
import lejos.hardware.lcd.LCD;
import lejos.utility.Delay;

public class Beispiel_Anzeige {

   public static void main(String[] args) {

      // Inhalt nächste Folie
   }

}
```

Erste Schritte: Beispielprogramm

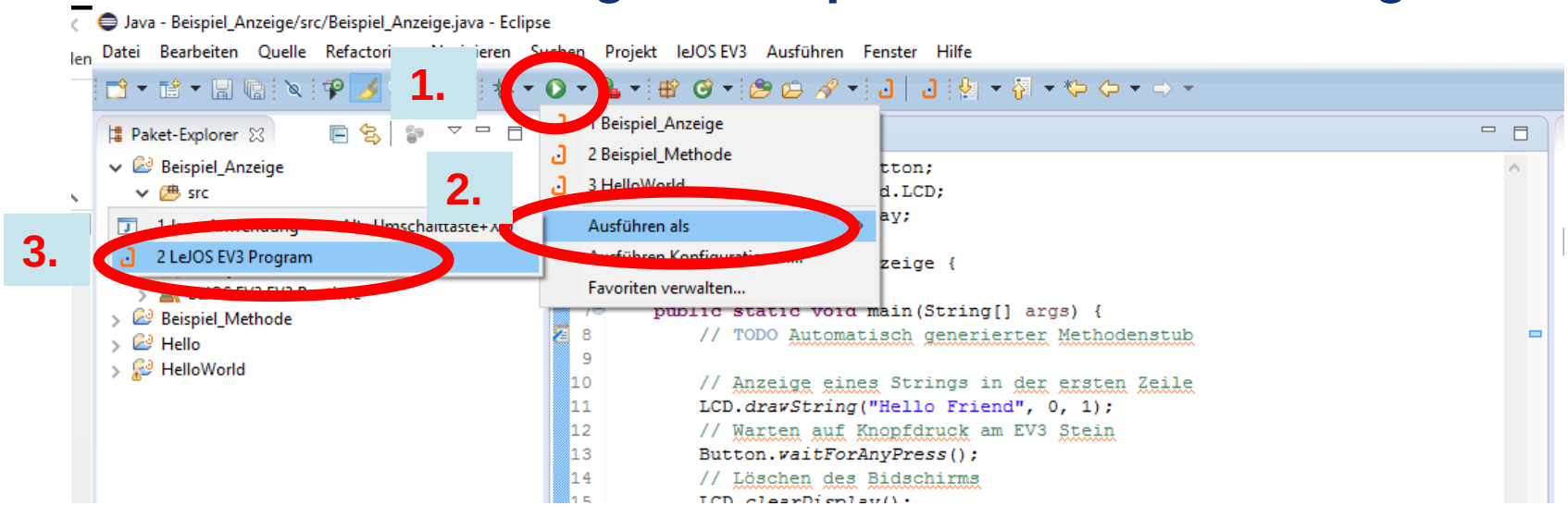
```
public static void main(String[] args) {
    // Anzeige eines Strings in der ersten Zeile
    LCD.drawString("Hello Friend", 0, 1);
    // Warten auf Knopfdruck am EV3 Stein
    Button.waitForAnyPress();
    // Löschen des Bidschirms
    LCD.clearDisplay();
    // Anzeige von Hello World
    System.out.println("Hello World");
    // Anzeige der Zahl 7 in der 2. Zeile
    LCD.drawInt(7, 0, 2);
    // Programm wartet 2s bevor es den nächsten Befehl ausführt
    Delay.msDelay(2000);
}
```

Erste Schritte: Programm Speichern und Übertragen



Datei speichern bzw. alle Dateien speichern drücken

Erste Schritte: Programm Speichern und Übertragen



- Ausführen als LeJOS EV3 Programm wählen, zuvor EV3 Stein einschalten

Achtung: Das Programm auf dem EV3 startet selbstständig!!!

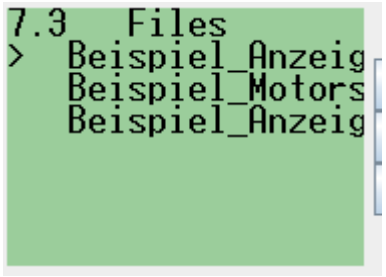
Arbeit mit dem EV3 Stein

Programme auf dem EV3 starten

Startbildschirm

1x Rechts drücken

1x Eingabe drücken



Arbeit mit dem EV3 Stein

Programme auf dem EV3 starten

- Auswahl der Programme mit Oben und Unten Button
- Bestätigung des Programms mit Eingabe Button

Programme auf dem EV3 starten

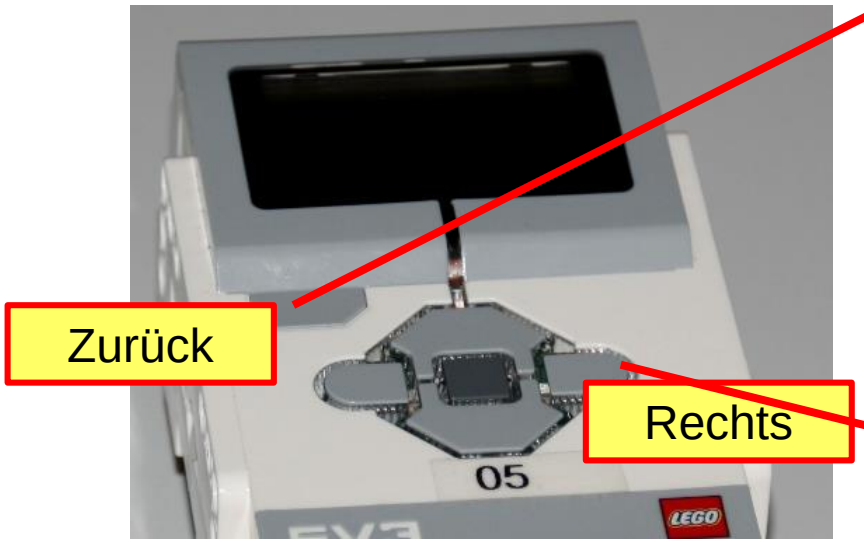
Nochmals auf Eingabe Button drücken

Programm wird gestartet

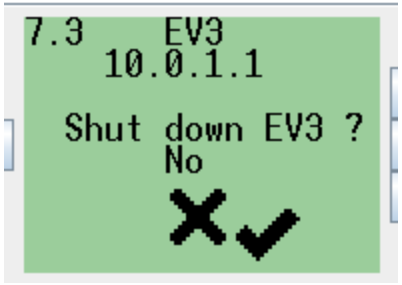
Zum Löschen von Programmen auf Mülleiner mit Links Button navigieren und dann mit Eingabe Button bestätigen

Arbeit mit dem EV3 Stein

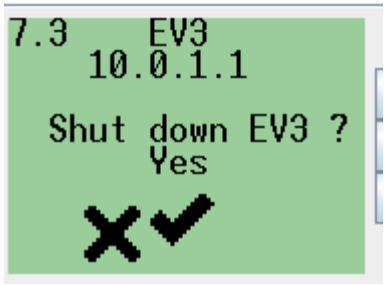
Ausschalten des EV3



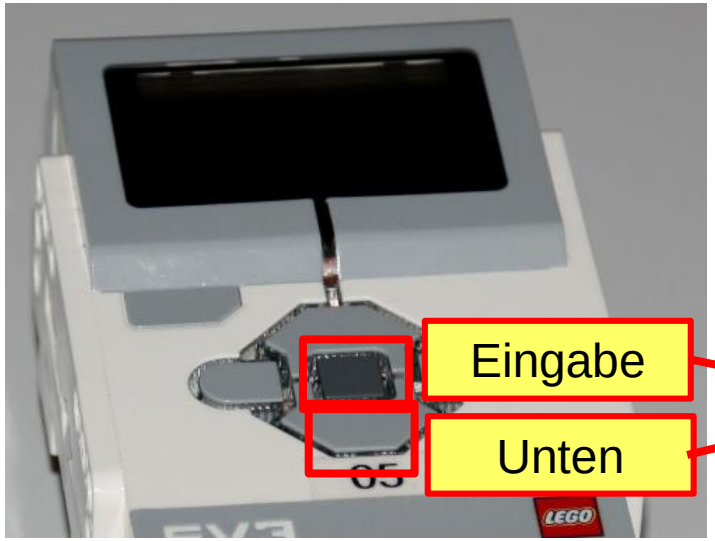
Zurück Button betätigen bis folgender Bildschirm erscheint



Dann auf Rechts Button drücken und mit Eingabe Button bestätigen



Abbruch eines laufenden Programms



Eingabe Button
und
Unten Button
gleichzeitig
drücken

DAS SPIELFELD: Legostadt

Allgemeiner Aufbau

Hinweise zur Bearbeitung der Praktikumsaufgaben

- Jede Aufgabe des Spielfeldes ist eine eigenständige Aufgabe. D. h. jede Aufgabe soll einzeln gelöst werden und muss nicht mit anderen Aufgaben kombiniert werden.
- Erstellen Sie ein Projekt mit einer Klasse, die eine Main Methode
- Schreiben Sie das entsprechende Programm
- Für eine neue Aufgaben, löschen Sie den nicht mehr benötigten Quelltext bzw. kommentieren diesen aus.
- Erstellen Sie für eine neue Aufgabe keine neue Klasse mit einer main-Methode in dem selben Projekt.
- Bei Bedarf erstellen Sie für eine neue Aufgabe ein neues Projekt mit einer neuen Klasse, die eine main-Methode enthält

DAS SPIELFELD: Legostadt

Übung 1: Bestimmung der Strecke, die der Roboter in 1 s vorwärts fährt

Start:
Einer der Übungsplätze Ü1, Ü2, Ü3 oder Ü4

Vorgehensweise:

- Schreiben eines Programm, dass den Roboter bei einer bestimmten Geschwindigkeit 1s vorwärts fahren lässt
- Platzieren des Roboters an der schwarzen Linie in einem der Übungsplätze
- Starten des Programms
- Messen der Strecke und Wert notieren
- Über Verhältsnisgleichungen kann man nun die Zeit bestimmen, die der Roboter braucht, um bestimmte Strecken zurückzulegen

DAS SPIELFELD: Legostadt

Steuerung zweier Motoren mittels Zeitangaben

Setzen einer definierten Geschwindigkeit:

```
Motor.B.setSpeed(400);
Motor.C.setSpeed(400);
```

Vorwärtsfahren:

Motor.***B.forward();***
Motor.***C.forward();***

Rückwärtsfahren:

Motor.***B.backward();***
Motor.***C.backward();***

Anhalten mit Bremsen:

Motor.***B.stop();***
Motor.***C.stop();***

Für die Nutzung der Motor-Befehle wird die import-Funktion benötigt:

```
import lejos.hardware.motor.Motor;
```

Überprüfen Sie, dass die Motoren in den Ports B und C angeschlossen sind! Wenn nicht, dann die Stecker in die entsprechenden Ports stecken oder Portangabe im Programm entsprechend ändern!

DAS SPIELFELD: Legostadt

Steuerung zweier Motoren mittels Zeitangaben

Beispielprogramm:

```
import lejos.hardware.motor.Motor;
import lejos.utility.Delay;

public class MotorBeispiel {

    public static void main(String[] args) {

        //Inhalt nächste Folie

    }

}
```

Der Roboter fährt
- Mit einer Geschwindigkeit von 400
- Fährt 1 s vorwärts
- Fährt 1 s rückwärts

DAS SPIELFELD: Legostadt

Steuerung zweier Motoren mittels Zeitangaben

Beispielprogramm:

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

    // Bei Bedarf: Setzen einer Motorgeschwindigkeit
    Motor.B.setSpeed(400);
    Motor.C.setSpeed(400);

    //Vorwärts für 1s
    Motor.B.forward();
    Motor.C.forward();
    Delay.msDelay(1000);

    // Fortsetzung nächste Folie
```

Der Roboter fährt
- Mit einer Geschwindigkeit von 400
- Fährt 1 s vorwärts
- Fährt 1 s rückwärts

DAS SPIELFELD: Legostadt

Steuerung zweier Motoren mittels Zeitangaben

Beispielprogramm:

```
        // Rückwärts für 1 s
        Motor.B.backward();
        Motor.C.backward();
        Delay.msDelay(1000);

        // Anhalten der Motoren
        Motor.B.stop();
        Motor.C.stop();
    }
```

Der Roboter fährt
- Mit einer Geschwindigkeit von 400
- Fährt 1 s vorwärts
- Fährt 1 s rückwärts

DAS SPIELFELD: Legostadt

Übung 2: Fahrt zum Haus

Start: Startfeld

Ende: Parkplatz am Haus

Der Roboter soll vom Startplatz zum Parkfläche am Haus fahren. Dabei soll er der vorgegebenen Straße folgen.

DAS SPIELFELD: Legostadt

Steuerung zweier Motoren mittels Zeitangaben

Kurve
Motor.***B.forward();***
Motor.***C.backward();***

oder
Motor.***B.backward();***
Motor.***C.forward();***

DAS SPIELFELD: Legostadt

Steuerung zweier Motoren mittels Zeitangaben

Beispielprogramm:

```
import lejos.hardware.motor.Motor;
import lejos.utility.Delay;

public class MotorBeispiel {

    public static void main(String[] args) {

        //Inhalt nächste Folie

    }

}
```

Der Roboter fährt eine Kurve

DAS SPIELFELD: Legostadt

Steuerung zweier Motoren mittels Zeitangaben

Beispielprogramm:

```
        // Kurve nach Links bzw. Rechts
        Motor.B.forward();
        Motor.C.backward();
        Delay.msDelay(500);

        // Anhalten der Motoren
        Motor.B.stop();
        Motor.C.stop();
    }
```

Der Roboter fährt eine Kurve

DAS SPIELFELD: Legostadt

Die Übungsphase ist vorbei.
Alles verstanden?
Wenn ja, dann kann mit der Bearbeitung der nachfolgenden Aufgaben begonnen werden.
Die nachfolgenden Aufgaben müssen durch einen Praktikumsbetreuer abgenommen und unterschrieben werden!

DAS SPIELFELD: Legostadt

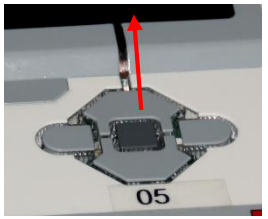
Aufgabe 1: Der Roboter soll entweder zum Krankenhaus oder zur Schule fahren (if … else Abfrage)

Start: Parkplatz am Haus

Ende: Parkplatz Krankenhaus bzw. Ein- und Ausstiegsfeld an der Schule

Der Roboter soll vom Parkplatz am Haus entweder zur Schule oder zum Krankenhaus fahren. Die Auswahl des Ziel erfolgt in Abhängigkeit vom Button, der am EV3 Stein gedruckt wird. Wird der obere Knopf gedrückt, soll der Roboter zum Krankenhaus, in allen anderen Fällen zur Schule fahren. Beide Wege sind gleichzeitig zum implementieren! Das Ziel soll angezeigt werden.

Oben



DAS SPIELFELD: Legostadt

Die if-else Abfrage

```
if(Ausdruck){
   Anweisung
   …
   Anweisung
}
else{
   Anweisung
   …
   Anweisung
}
```

Wenn der Ausdruck erfüllt ist, so werden die Anweisungen im if-Block erfüllt, ansonsten die Anweisung im else-Block.

Beispiel:

```
if(a==10){
   Anweisung
   …
   Anweisung
}
else{
   Anweisung
   …
   Anweisung
}
```

DAS SPIELFELD: Legostadt

Vergleichsoperatoren

Operator	Beispiel	Wirkung
>	a > b	a größer als b
>=	a >= b	a größer oder gleich b
<	a < b	a kleiner als b
<=	a <= b	a kleiner oder gleich b
==	a == b	a ist gleich b
!=	a != b	a ist ungleich b

DAS SPIELFELD: Legostadt

Abfrage von EV3 Buttons

Warten auf Knopfdruck:

```
Button.waitForAnyPress();
```

Abfrage, ob Knopf oben gedrückt ist:

```
Button.getButtons()==Button.ID_UP
```

Löschen des Buttonabfrageergebnisses

```
Button.discardEvents();
```

DAS SPIELFELD: Legostadt

Beispielprogramm: if Abfrage

```
import lejos.hardware.Button;
import lejos.hardware.lcd.LCD;
import lejos.utility.Delay;

public class KnopfBeispiel {

   public static void main(String[] args) {

      // Inhalt nächste Folie

   }

}
```

Das Programm fragt ab, ob der linke oder ein anderer Knopf gedrückt wurde.

DAS SPIELFELD: Legostadt

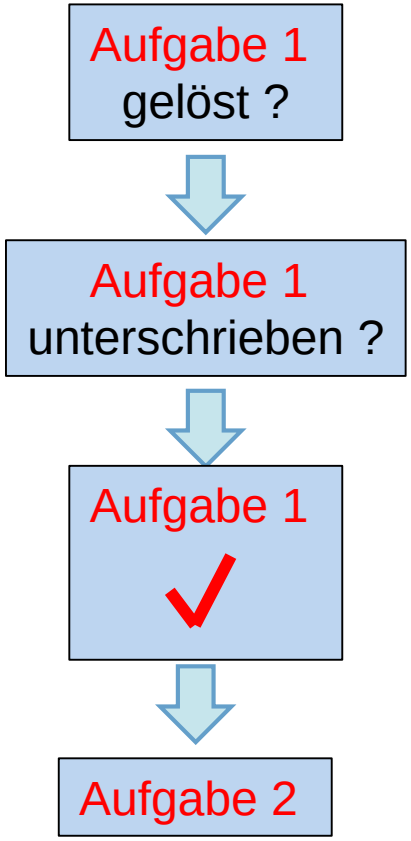
Beispielprogramm:

```
public static void main(String[] args) {
    // Warten auf Knopfdruck
    Button.discardEvents();
    LCD.drawString("Druecke Knopf", 0, 1);
    Button.waitForAnyPress();

    // Abfrage, ob Knopf oben gedrueckt ist
    if(Button.getButtons()==Button.ID_UP){
        LCD.drawString("Oben", 0, 2);
    }
    else{
        LCD.drawString("anderer", 0, 2);
    }
    Delay.msDelay(2000);
}
```

Das Programm fragt ab, ob der linke oder ein anderer Knopf gedrückt wurde.

DAS SPIELFELD: Legostadt

Aufgabe 1: Fahrt zum Krankenhaus oder zur Schule (if-else Abfrage)



DAS SPIELFELD: Legostadt

Aufgabe 2: Beförderung von Fahrgästen zwischen Bahnhof und Airport (for Schleife)

Start und Ende: Parkfläche Bahnhof

Der Roboter soll als Shuttlebus Gäste zwischen Bahnhof und Airport hin und zurück befördern.

Der Roboter startet per Knopfdruck, wenn der Gast eingestiegen ist. Der Roboter fährt die Strecke vom Bahnhof zum Airport vorwärts. Lässt Gäste ein- und aussteigen und fährt nach Knopfdruck die gleiche Strecke rückwärts zurück.

Die Zahl soll angezeigt werden.

Insgesamt soll der Roboter die Strecke 3mal absolvieren!

Auf den Parkflächen darf der Roboter neu ausgerichtet werden!

DAS SPIELFELD: Legostadt

Die for-Schleife

Eine Anweisung bzw. eine Folge von Anweisungen soll mehrfach wiederholt werden.

```
for(Startwert;Endwert;Erhöhung){
    Anweisung
    …
    Anweisung
}
```

Beispiel:

```
for(i=1;i<=7;i++){
    Anweisung
    …
    Anweisung
}
```

DAS SPIELFELD: Legostadt

Beispielprogramm: for Schleife

Das Programm gibt das Wort Test 4mal aus.

```
import lejos.hardware.lcd.LCD;
import lejos.utility.Delay;

public class BeispielFor {
   public static void main(String[] args) {

      LCD.clearDisplay();
      // Das Wort Test wird 4mal ausgegeben
      for(int i=1;i<=4;i++)
      {
         System.out.println("Test");
      }
      Delay.msDelay(4000);
   }
}
```

DAS SPIELFELD: Legostadt

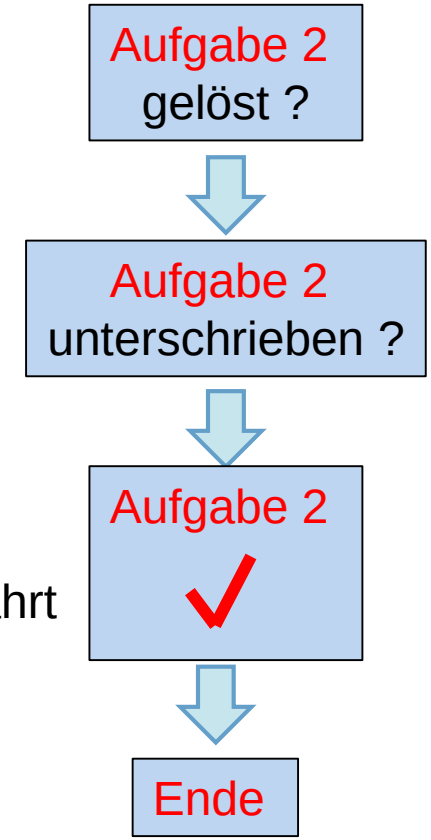
Aufgabe 2: Beförderung von Fahrgästen zwischen Flughafen und Hotel

Start und Ende: Parkfläche Bahnhof

Der Roboter soll als Shuttlebus Gäste zwischen Bahnhof und Airport hin und zurück befördern.

Der Roboter startet per Knopfdruck, wenn der Gast eingestiegen ist. Der Roboter fährt die Strecke vom Bahnhof zum Airport vorwärts. Lässt Gäste ein- und aussteigen und fährt nach Knopfdruck die gleiche Strecke rückwärts zurück.

Insgesamt soll der Roboter die Strecke 3mal absolvieren!



Auf den Parkflächen darf der Roboter neu ausgerichtet werden!