hochschule mannheim

Fakultät für Informatik, Institut für Robotik

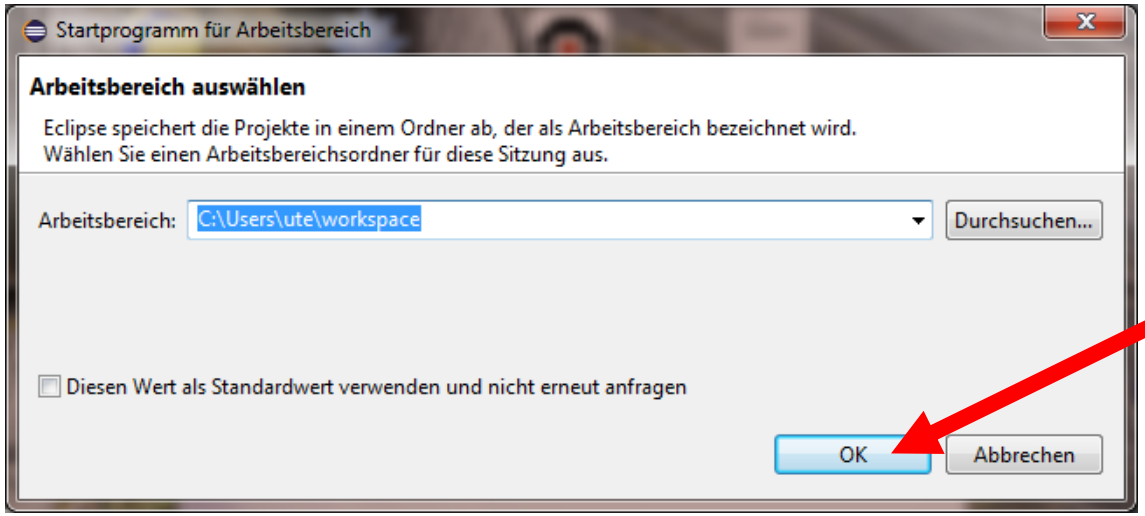
Lego Mindstorms NXT - Programmierung mit JAVA Einführung

Ute Ihme

Start der Entwicklungsumgebung

Starten von Eclipse

1. Starten der 32-bit Version von Eclipse (Eclipse Luna)
2. Auswahl des Arbeitsbereiches
   Standardeinstellungen übernehmen

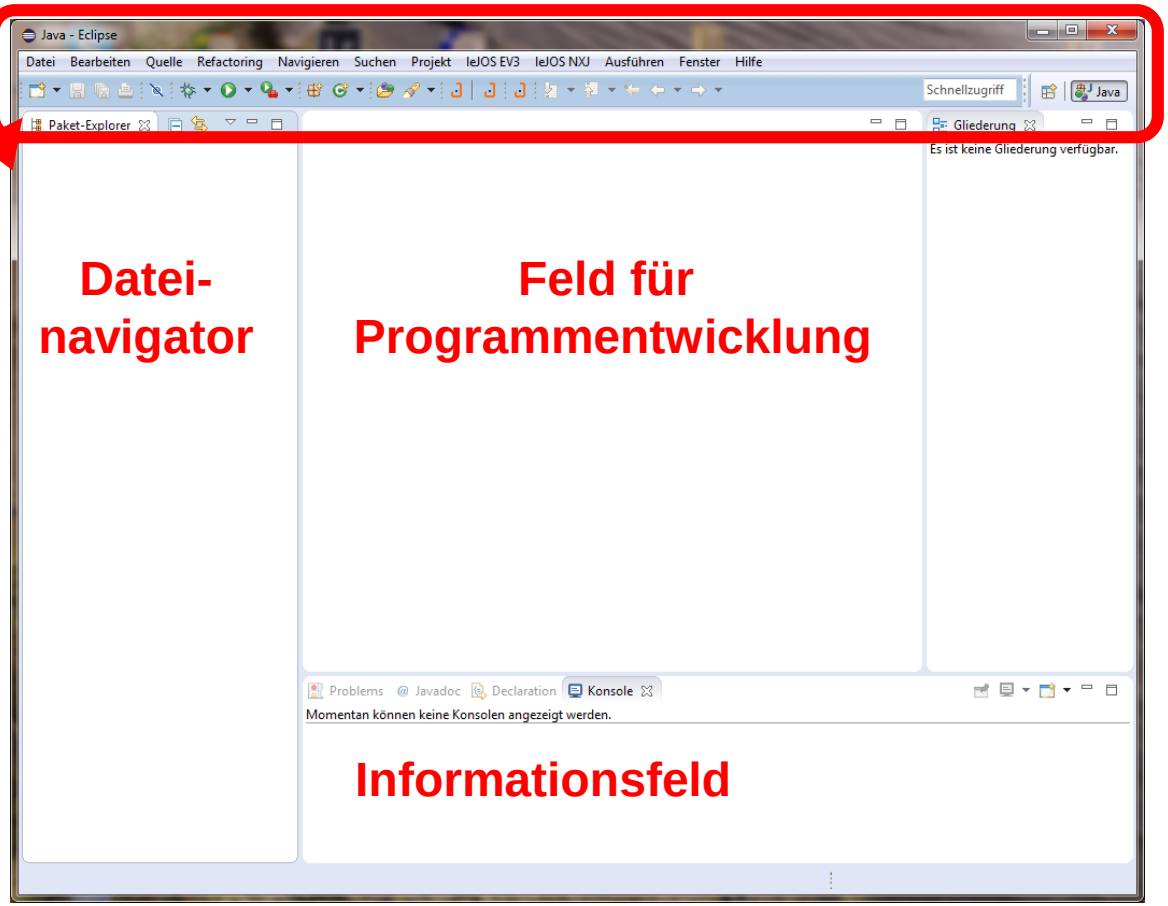


auf OK-Button drücken

Start der Entwicklungsumgebung

Starten von Eclipse

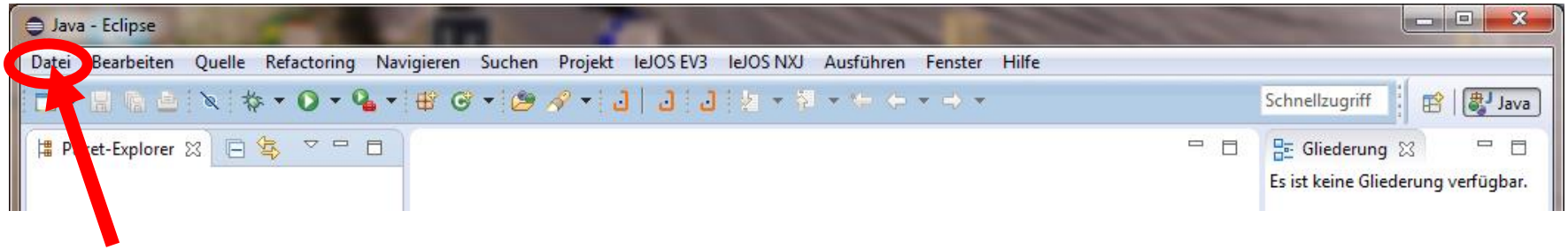
3. Programmierumgebung Eclipse wird gestartet



Start der Entwicklungsumgebung

Starten von Eclipse

3. Programmierung vorbereiten

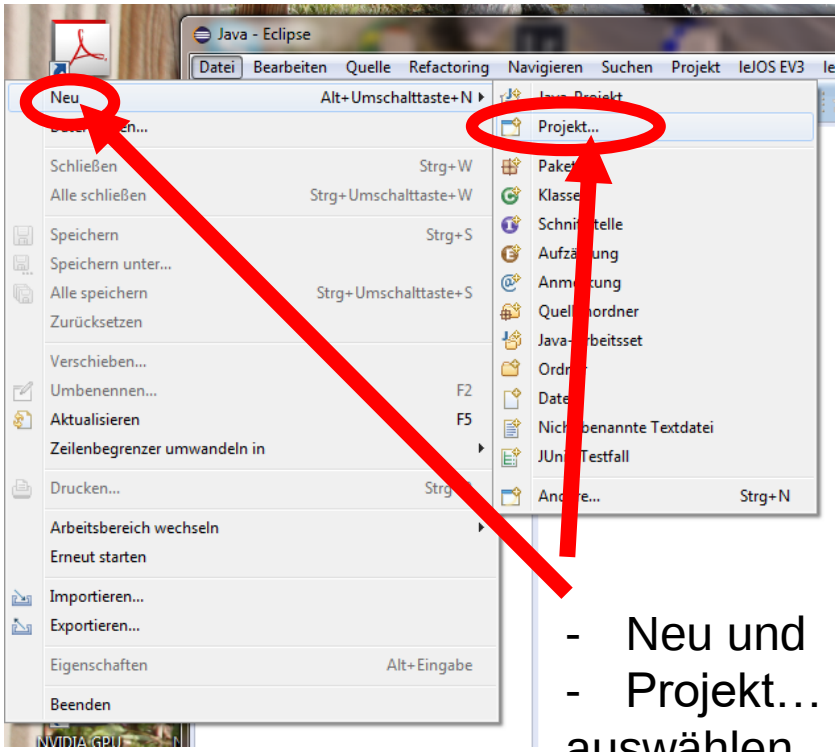


Datei wählen

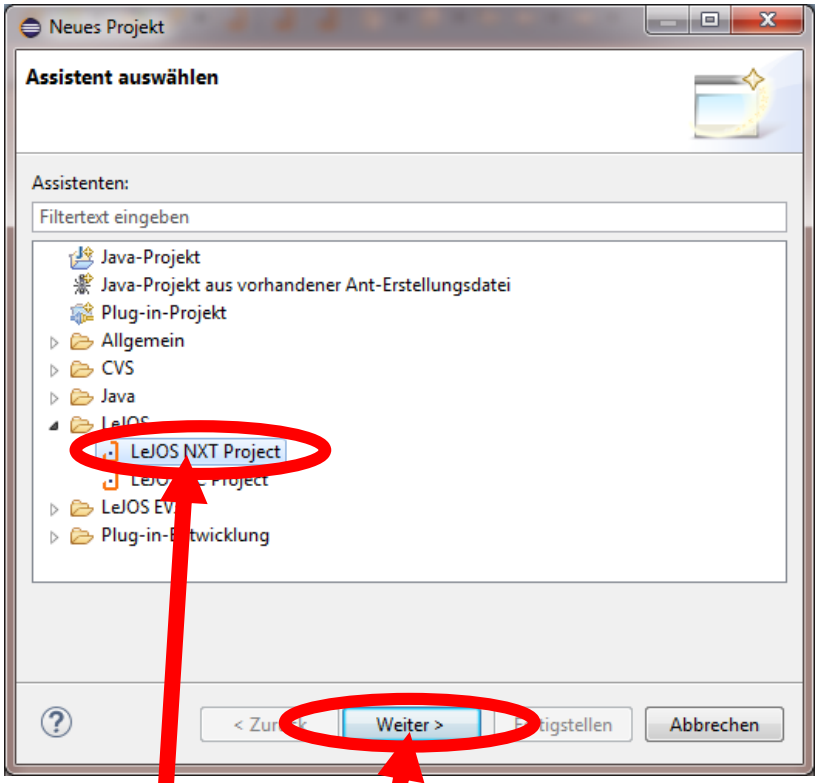
Start der Entwicklungsumgebung

Starten von Eclipse

3. Programmierung vorbereiten



- Neu und
- Projekt… auswählen

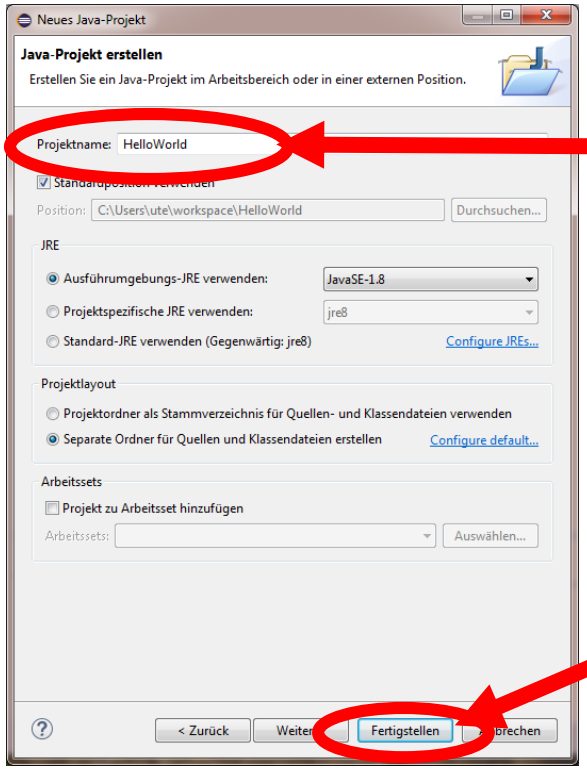


- LeJOS NXT Projekt wählen
- auf Weiter> drücken

Start der Entwicklungsumgebung

Starten von Eclipse

3. Programmierung vorbereiten



Projektname eingeben

Fertigstellen drücken

JAVA Code

JAVA Basics I

- Jedes JAVA Programm besteht aus Klassen.

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

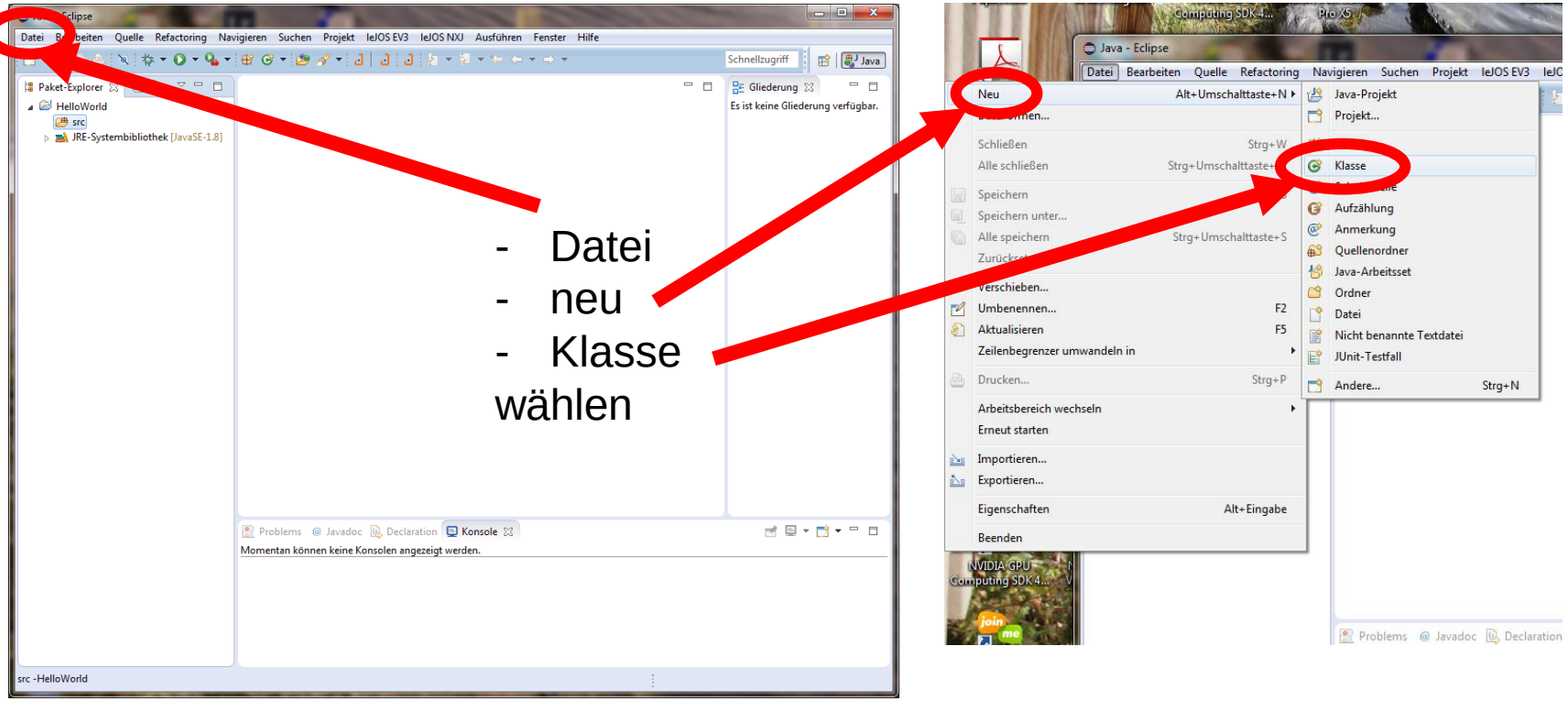
- Eine der Klassen muss eine **main Methode** besitzen.

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

Start der Entwicklungsumgebung

Starten von Eclipse

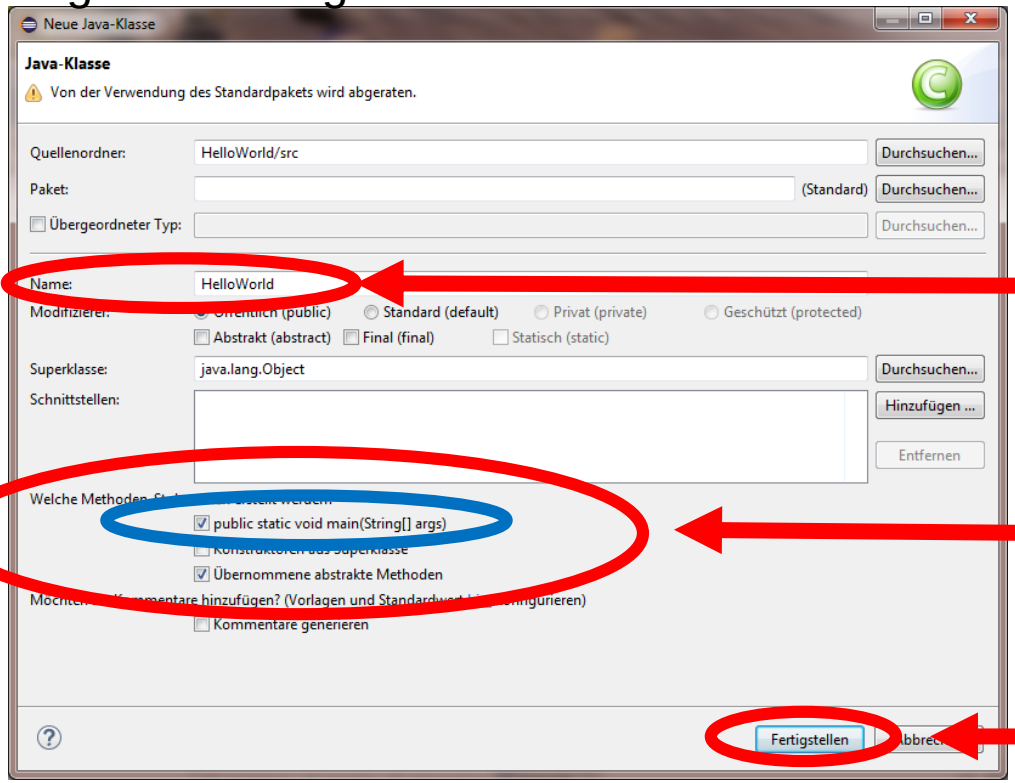
3. Programmierung vorbereiten



Start der Entwicklungsumgebung

Starten von Eclipse

3. Programmierung vorbereiten



1. Klassennamen eingeben

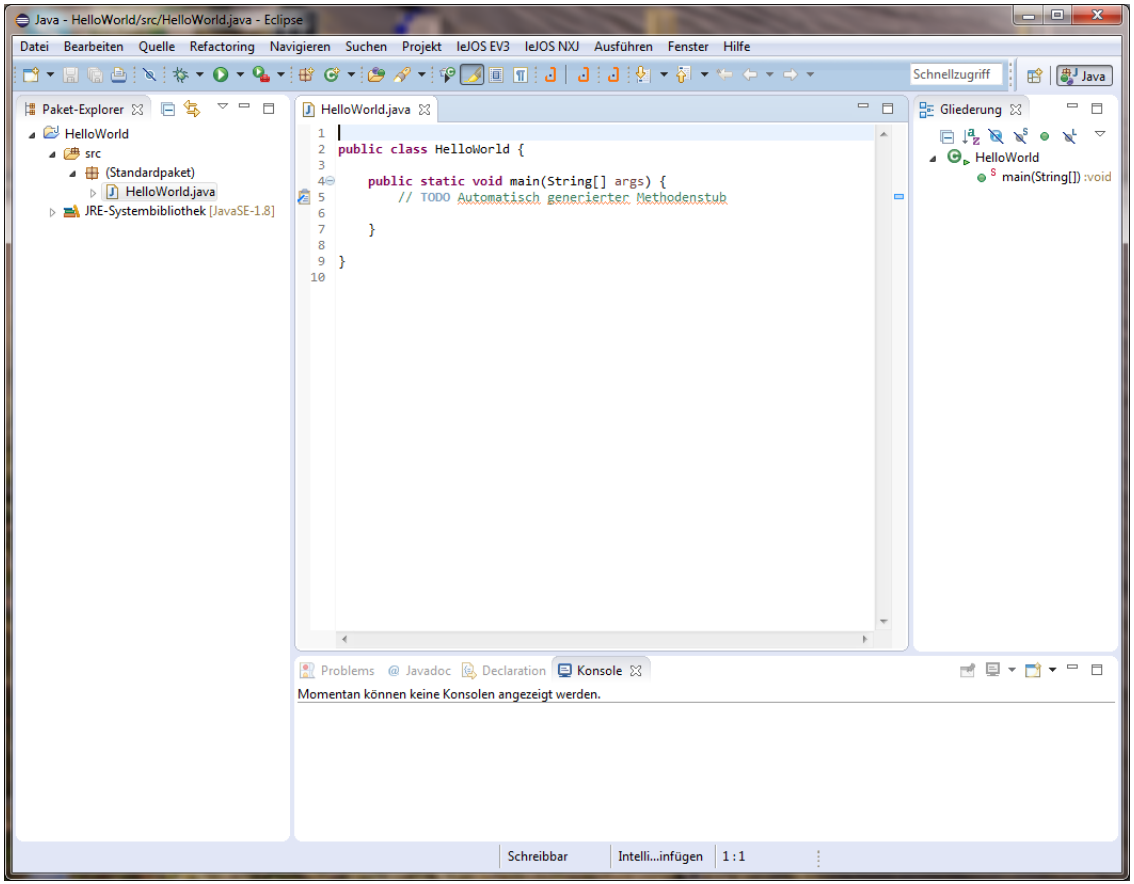
2. Hier zusätzlich **public static void main** ankreuzen, wenn Klasse main Methode enthalten soll.

3. Fertigstellen drücken

Start der Entwicklungsumgebung

Starten von Eclipse

3. Programmierung vorbereiten



Nun kann das
Programmieren beginnen!

JAVA Code

Realisierung eines Pausenbefehls

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

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

2. Nutzung eines leJos Befehls

```
Delay.msDelay(2000);
```

3. Nutzung eines Threads

JAVA Code

Displayanzeige

Syntax:

1. Nutzung des Standard JAVA Befehls

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

2. Nutzung des lejos Befehls
   a) für Strings

```
LCD.drawString("Hello World 1", 0, 2);
```

   b) Für Zahlen

```
LCD.drawInt(7, 0, 4);
```

3. Löschen des Displays

```
LCD.clear();
```

JAVA Code

Das erste Programm: Bildschirmanzeige / Hello World

```
// Importieren von LejosBibliotheken
import lejos.nxt.*;
import lejos.util.*;

public class HelloWorld {

public static void main(String[] args) {
        // TODO Automatisch generierter Methodenstub

        // Nutzung der JAVA Syntax für die Ausgabe eines Strings
        System.out.println("Hello World!");
        Button.waitForAnyPress();
        // Bildschirmlöschen
        LCD.clear();
```

JAVA Code

Das erste Programm: Bildschirmanzeige / Hello World

```
        // Ausgabe eines Strings mit dem LejosBefehl
         LCD.drawString("Hello World 1", 0, 2);
         // Warten für 2000ms
         Delay.msDelay(2000);

        // Ausgabe einer Zahl mit dem LejosBefehl
         LCD.drawInt(7, 0, 4);
         Delay.msDelay(2000);

        }
}
```

JAVA Code

Die Klasse Motor

Klasse enthält Objekte eigenen Typs, die direkt mit
Motor.A, Motor.B und Motor.C
referenziert werden können.

Wichtige Methoden der Klasse Motor

Vorwärtsfahren:
Motor.***B.forward();***

Rückwärtsfahren:
Motor.***B.backward();***

Anhalten:
Motor.***B.stop();***

JAVA Code

Motorensteuerung I
- zeitgesteuert -

Syntax:

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

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

Anhalten:
Motor.***B.stop();***

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

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

JAVA Code

Motorensteuerung I - zeitgesteuert -

Programmbeispiel:

```
import lejos.nxt.*;
import lejos.util.*;

public class Motoren {

public static void main(String[] args) {

        // Vorwärtsfahren
        Motor.B.forward();
        Motor.C.forward();
        Delay.msDelay(2000);
        Motor.B.stop();
        Motor.C.stop();
```

JAVA Code

Motorensteuerung I

- zeitgesteuert -

Programmbeispiel:

```
// Rückwärtsfahren
Motor.B.backward();
Motor.C.backward();
Delay.msDelay(2000);
Motor.B.stop();
Motor.C.stop();

// Kurve mit einem Motor
Motor.B.forward();
Delay.msDelay(2000);
Motor.B.stop();
//Kurve mit 2 Motoren
Motor.B.forward();
Motor.C.backward();
Delay.msDelay(2000);
Motor.B.stop();
Motor.C.stop();
  }
}
```

DAS SPIELFELD: Legostadt

Allgemeiner Aufbau

Hinweise zur Bearbeitung der Aufgaben

- 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

JAVA Code

Die Klasse Motor

Weitere Funktionen der Klasse Motor

- void setSpeed(int speed)
  setzt die Motorgeschwindigkeit
  Parameter speed wird in Grad pro Sekunde angegeben (max. 900)
- void reverseDirection()
  dreht die aktuelle Drehrichtung um
- void flt()
  stopt die Motordrehung, lässt die Motoren dabei aber auslaufen

JAVA Code

Die Klasse Motor

Weitere Funktionen der Klasse Motor

- void rotate(int angle)
  lässt Motor um den angebenen Winkel drehen
- void rotateTo(int limitAngle)
  lässt den Motor zu dem angebenden Winkel drehen
- int getTachoCount()
  gibt den Drehwinkel seit dem letzten Aufruf von resetTachoCount()
- void resetTachoCount()
  setzt den Drehwinkel auf 0
- void lock(int power)
  falls das Anhalten eines Motors mit stop nicht reicht, z.B. Roboterarm
  Verstärkung der Haltekraft zwischen 1 und 100

Motorensteuerung II /- rotationsgesteuert -

```
import lejos.nxt.*;
import lejos.util.*;

public class MotorKlasse {
        // Motorenkann man auch als Variablen deklarien
        // Instanziierung zweier Motoren
        static NXTRegulatedMotor leftMotor = Motor.B;
        static NXTRegulatedMotor rightMotor = Motor.C;

        public static void main(String[] args) {
                //Setzen der Geschwindigkeit
                leftMotor.setSpeed(500);
                rightMotor.setSpeed(500);
                //Vorwärtsfahren bis Drehwinkel von 300 erreicht ist
                leftMotor.rotate(300,true);
                rightMotor.rotate(300);
        }
}
```

DAS SPIELFELD: Legostadt

Übung 2: Bestimmung der Strecke, die der Roboter in einer Umdrehung 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

Aufgabe 1: Der Roboter soll von einem der Übungsplätze auf das Startfeld fahren

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

Ziel: Erreichen des Startfeldes

Methode: Zeitgesteuerter Fahren

DAS SPIELFELD: Legostadt

Aufgabe 2: Fahrt zum Haus

Start: Startfeld

Ende: Parkplatz am Haus

Methode: rotationsgesteuertes Fahren

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

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
}
```

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

Abfrage von NXT Buttons

Warten auf Knopfdruck:

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

Abfrage, ob Knopf links gedrückt ist:

Button.***LEFT.isDown()***

Beispiel: Abfrage von NXT Buttons

```
import lejos.nxt.Button;
import lejos.nxt.LCD;
import lejos.util.Delay;

public class Test {

public static void main(String[] args) {

        Programm siehe nächste Folie

}
```

Beispiel: Abfrage von NXT Buttons

```
public static void main(String[] args) {
        Button.waitForAnyPress();
        if (Button.LEFT.isDown()){
            LCD.drawString("Links", 0, 3);
        }
        else
        {
            LCD.drawString("Rechts", 0, 3);
        }
        Delay.msDelay(1000);
        }
}
```

DAS SPIELFELD: Legostadt

Aufgabe 3: 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 NXT Stein gedruckt wird. Wird der linke 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.

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 4: 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!

JAVA Code

Berührungssensor / Tastsensor

Initialisierung:

```
public static TouchSensor touchSensor =
                    new TouchSensor(SensorPort.S1);
```

Hinweis:
S1 wird verwendet, wenn der Tastsensor am Port S1 angeschlosssen ist .
Wird der Sensor an einem anderen Port angeschlossen, so ist entsprechend S2, S3 oder S4 zu verwenden.

Abfrage:

```
boolean sensorPressed = false;

sensorPressed = touchSensor.isPressed();
```

JAVA Code

Berührungssensor / Tastsensor

Programmbeispiel:

```
import lejos.nxt.*;

public class Tastsensor {
   //Initialisierung des Tastsenors am Port 1
   public static TouchSensor touchSensor = new
        TouchSensor(SensorPort.S1);

   public static void main(String[] args) {

      //Festlegung einer booleschen Variablen für den Tastsensor
      boolean sensorPressed = false;
```

JAVA Code

Berührungssensor / Tastsensor

Programmbeispiel:

```
        //Geradausfahren bis Tastsensor gedrückt wird
        Motor.B.backward();
        Motor.C.backward();
        while(sensorPressed==false){
            //Auslesen des Zustandes des Tastsensor
            sensorPressed = touchSensor.isPressed();
        }
        Motor.B.stop();
        Motor.C.stop();
    }

}
```

DAS SPIELFELD: Legostadt

Aufgabe 5: Anhalten mittels Tastsensor

Start: Startfeld

Ende: Parkplatz Parkplatz Berghütte

Der Roboter soll von der Startfliese zum Parkplatz Berghütte fahren. Dabei soll der Roboter autonom einparken, dass heißt, sobald der Tastsensor des Roboters die Wand am Parkplatz berührt soll der Roboter anhalten.

Die Aufgabe soll als eigenständige Methode in der Hauptklasse realisiert werden. Initialisierung und Sensorabfragen sollen innerhalb dieser Methode realisiert werden.

JAVA Code

Ultraschallsensor

Initialisierung für Port 2:

```
public static UltrasonicSensor sonicSensor =
        new UltrasonicSensor(SensorPort.S2);
```

Wichtig Methode:

- int getDistance()
  Gibt die Entfernung zwischen Sensor und Hindernis in cm an.
  Maximalwert 255

JAVA Code

Ultraschallsensor

Programmbeispiel:

```
import lejos.nxt.*;

public class Ultraschall {

    //Initialsierung Ultraschallsensor an Port 2
    public static UltrasonicSensor sonicSensor = new
       UltrasonicSensor(SensorPort.S2);
    public static void main(String[] args) {
       int distance=255;

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

JAVA Code

Ultraschallsensor

Programmbeispiel:

```
        while(distance>=10){
            //Auslesen des Zustandes des Ultraschallsensor
            distance = sonicSensor.getDistance();
        }
        Motor.B.stop();
        Motor.C.stop();
    }
}
```

Aufgabe 3:
Einparken mittels Ultraschallsensor

Start: Parkplatz Bahnhof

Ende: Parkplatz Flughafen

Der Roboter soll vom Parkplatz Bahnhof zum Parkplatz Flughafen fahren. Dabei soll der Roboter autonom in einer Distanz von ca. 5 cm der Wand anhalten.

Die Aufgabe soll als eigenständige Methode in der Hauptklasse realisiert werden. Initialisierung und Sensorabfragen sollen innerhalb dieser Methode realisiert werden.

Ultraschallsensor an Port 4 anschließen!

JAVA Code

Lichtsensor

Initialisierung:

```
public static LightSensor lightSensor =
        new LightSensor(SensorPort.S3);
```

Wichtige Methoden:

- int getLightValue()
  gibt den aktuellen Sensorwert zurück in % (0-100%)
- void setFloodlight(boolean floodlight)
  Ein- und Ausschalten der aktiven Beleuchtung

JAVA Code

Lichtsensor

```
import lejos.nxt.*;

public class Lichtsensor {
    public static LightSensor lightSensor =
        new LightSensor(SensorPort.S3);

        public static void main(String[] args) {
            int licht = 100;

            Motor.B.forward();
            Motor.C.forward();
            while(licht>=40){
                //Auslesen des Zustandes des
                //Lichtsensor
                licht = lightSensor.getLightValue();
            }
            Motor.B.stop();
            Motor.C.stop();
    }
}
```