

# DATA SCIENCE 1

TUTORIALDAY - LEGO SPIKE EDUCATION

PROF. DR. CHRISTIAN BOCKERMANN

HOCHSCHULE BOCHUM

WINTERSEMESTER 2023/2024

## LEGO Spike Legacy App

- Programmierumgebung für LEGO Hub
- USB-Verbindung mit HUB möglich
- Unterstützt graphische Programmierung + Python
- direktes Auslesen der Sensoren



The screenshot shows the LEGO Spike Legacy software interface. The main window displays a Python script for controlling a LEGO Spike robot. The script includes imports for PrimeHub, LightMatrix, Button, StatusLight, ForceSensor, MotionSensor, Speaker, ColorSensor, wait\_for\_seconds, wait\_until, and Timer. It initializes a PrimeHub, sets a light matrix to 'HAPPY', and configures a motor (gabel) with a default speed of 20. A timer is also initialized.

```
1 from spike import PrimeHub, LightMatrix, Button, StatusLight, ForceSensor, MotionSensor, Speaker, ColorSensor, A
2 from spike.control import wait_for_seconds, wait_until, Timer
3 from math import *
4 import hub
5
6 #hub = PrimeHub()
7 #hub.light_matrix.show_image('HAPPY')
8
9 timer = Timer()
10
11 #button = ForceSensor('D')
12 aufzug = Motor('B')
13 aufzug.set_default_speed(100)
14 gabel = Motor('E')
15 gabel.set_default_speed(20)
16
17 gabelRaus = False
18
```

Below the script is a console area labeled "Konsole". On the right side, a sidebar menu titled "Wissensdatenbank" (Knowledge Database) is open, listing various sensor and actuator categories with expandable arrows:

- Erste Schritte
- App
- Tasten
- Farbsensor
- Abstandssensor
- Kraftsensor
- Lichtmatrix
- Mathematische Funktionen
- Bewegungssensor

The interface also features a top navigation bar with a home icon, a tab labeled "LagerRobotik", and a plus sign. At the bottom, there are navigation icons for zooming, undo, redo, and a play button.

## Python Modul **spike**

- Python Modul zum Zugriff auf den Hub
- Funktionen für Hardware (Farbsensor, Motor, usw.)

```
from spike import PrimeHub

# Hub Objekt erzeugen
hub = PrimeHub()

# Pixel in der Mitte anschalten:
hub.light_matrix.set_pixel(3,3, brightness=100)
```

## LEGO Winkelmotor

- Winkelgenauer Motor
- Anschluss an einen der 6 Ports
- Drehen mit Winkel, Geschwindigkeit
- aktuellen Winkel auslesen



```
m = Motor('A')  
  
# Winkel abfragen  
winkel = m.get_position()  
  
# Um 90 Grad drehen mit Geschwindigkeit 10  
m.run_for_degrees(90, speed=10)
```

## LEGO Entfernungsmesser

- Entfernungsmessung über Schallwellen
- Meßbereich zwischen 1cm und 200cm
- integrierte LEDs in den “Augen”



```
from spike import DistanceSensor

sensor = DistanceSensor('A')

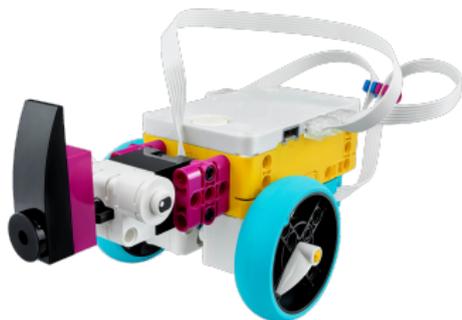
entfernung = sensor.get_distance_cm()
print("Entfernung in cm: ", entfernung)
```

## L2.1 Entfernung auslesen

- Verbinden Sie den Entfernungsmesser mit einem Port
- Schreiben Sie ein Programm, das kontinuierlich die Entfernung abfragt und mit `print` ausgibt.
- Erweitern Sie ihr Programm so, dass bei einem Entfernungswert unter 10 der mittlere Pixel auf dem Hub leuchtet, sonst soll er nicht leuchten.

## Motor-Paare

- Ansteuern zweier Motoren gleichzeitig
- Links/Rechts-Lenkung möglich



```
from spike import MotorPair  
  
paar = MotorPair('A', 'B')  
  
# geradeaus:  
paar.move(20)  
  
# Linkskurve:  
paar.move(10, steering=-10)
```

## L2.2: Einfaches Fahrzeug

- Bauen Sie die beiden Motoren an den Hub (vgl. Modell auf der vorherigen Seite)
- Verbinden Sie die Motoren mit Ports am Hub
- Schreiben Sie ein Programm, dass erst kurz gerade aus und dann eine Linkskurve fährt.

## L2.3: Figuren fahren

- Erweitern Sie Ihr Programm so, dass das Fahrzeug ein Quadrat abfährt.

## L2.3: Hinderniss-Erkennung

- Erweitern Sie Ihr Fahrzeug um den Entfernungsmesser
- Ihr Fahrzeug soll nun so lange geradeaus fahren, bis es ein Hinderniss vor sich erkennt (z.B. Abstand kleiner  $X$ )
- Wenn das Hinderniss nicht mehr da ist, soll das Fahrzeug geradeaus weiterfahren.