

# How To MobRob


## Access-Point

1. MobRob starten (Kippschalter auf der Hinterseite)
2. Bei verfügbaren WiFi-Netzwerken sollte nach kurzer Zeit das Netzwerk "MobRob\_AP" auftauchen
3. "Edit Connection" im Netzwerk-Manager auswählen
4. Auf den Reiter IPv4 klicken
5. Bei Methode Manual bzw. statisch auswählen
6. Auf "Add" klicken und folgende Adresse hinzufügen (siehe Bild)
7. Einstellung speichern und verbinden
8. Passwort "**mobrules**" eingeben


Edit your Network Connections


Type here to search connections...


Wired Ethernet


 **Wired connection 1**  
Last used 19 minutes ago

Wi-Fi

 **eduroam**  
Connected

 **MobRob\_AP**  
Last used yesterday

 **fritzbox**  
Last used yesterday

 **NSA\_SURVEILLANCE**  
Last used on 30.11.18

+

⊘

⌵

Connection name: MobRob\_AP

General configuration

Wi-Fi

Wi-Fi Security

IPv4

IPv6

Method: Manual

DNS Servers:

Search Domains:

DHCP Client ID:

Address	Netmask	Gateway
192.168.100.2	255.255.255.0	192.168.100.1

+ Add

⊘ Remove

☐ IPv4 is required for this connection

Routes...

? Help

↶ Reset

⌵ Defaults

✓ OK

✓ Apply

⊘ Cancel

## Odroid Ordner einbinden via sshfs

1. sshfs installieren: `sudo apt-get install sshfs`
2. Ordner erstellen: `mkdir ~/fusessh` (o. ä.)
3. zum einbinden: `sshfs linadm@192.168.100.1:/home/linadm/Codes/mobrob ~/fusessh` (Verbindung zum MobRob\_AP benötigt)
4. zum unmounten: `fusermount -u ~/fusessh`

## SSH-Verbindung mit Odroid

1. ssh installieren falls noch nicht geschehen
2. mit `ssh linadm@192.168.100.1` verbinden

## Code Dependencies

1. Odroid-Software:
  - a. ZMQ Lib
  - b. ArduinoJson Lib
  - c. MobRob Repo
2. µC-Software:
  - a. PlatformIO (VS-Code Extension oder Atom Extension)
  - b. ArduinoJson Lib
  - c. Port Berechtigungen (99-platformio-udev.rules im /etc/udev/rules.d Verzeichnis)
  - d. Embedded Code aus MobRob Repo inkl. Komm.-Board Software
3. GUI-Software:
  - a. npm (JavaScript Packet Manager)
  - b. node-js (JavaScript Lib)

## Sensordaten auslesen

1. Instanz der Klasse Sensor erstellen **`Sensor sensor;`**
2. requestHandler aufrufen
3. Filedescriptor und Argument zum Auslesen der Sensordaten übergeben:  
**`requestHandler(fd, sensorRead);`**
4. Sensordaten in String schreiben: **`std::string json = sensor.requestSensorDataJsonString(fd);`**
5. Sensordaten in Vector parsen: **`std::vector<int> sensordata = sensor.jsonToSensorData(fd);`**

## Servo-Geschwindigkeiten setzen

1. Instanz der Klasse Servo erstellen: **`Servo servo;`**
2. Funktion `setServoVelocities` aufrufen, Filedescriptor und Geschwindigkeitsvector übergeben: **`servo.setVelocities(fd, VelocityVector);`** mit `VelocityVector(VelocityLeft, VelocityRight);`
3. Mögliche Geschwindigkeiten von -1023 (Rückwärts) bis 1023 (Vorwärts)

## Servo-Geschwindigkeiten auslesen

1. Instanz der Klasse Servo erstellen: **`Servo servo;`**
2. Servo-Geschwindigkeiten in String schreiben: **`std::string json = servo.requestServoDataJsonString(fd);`**
3. mit **`servo.parseJsonVelocities(json);`** Servo-Geschwindigkeiten in Vector (private) parsen
4. mit **`servo.velocitiesInMeterPerSec();`** in m/s umwandeln