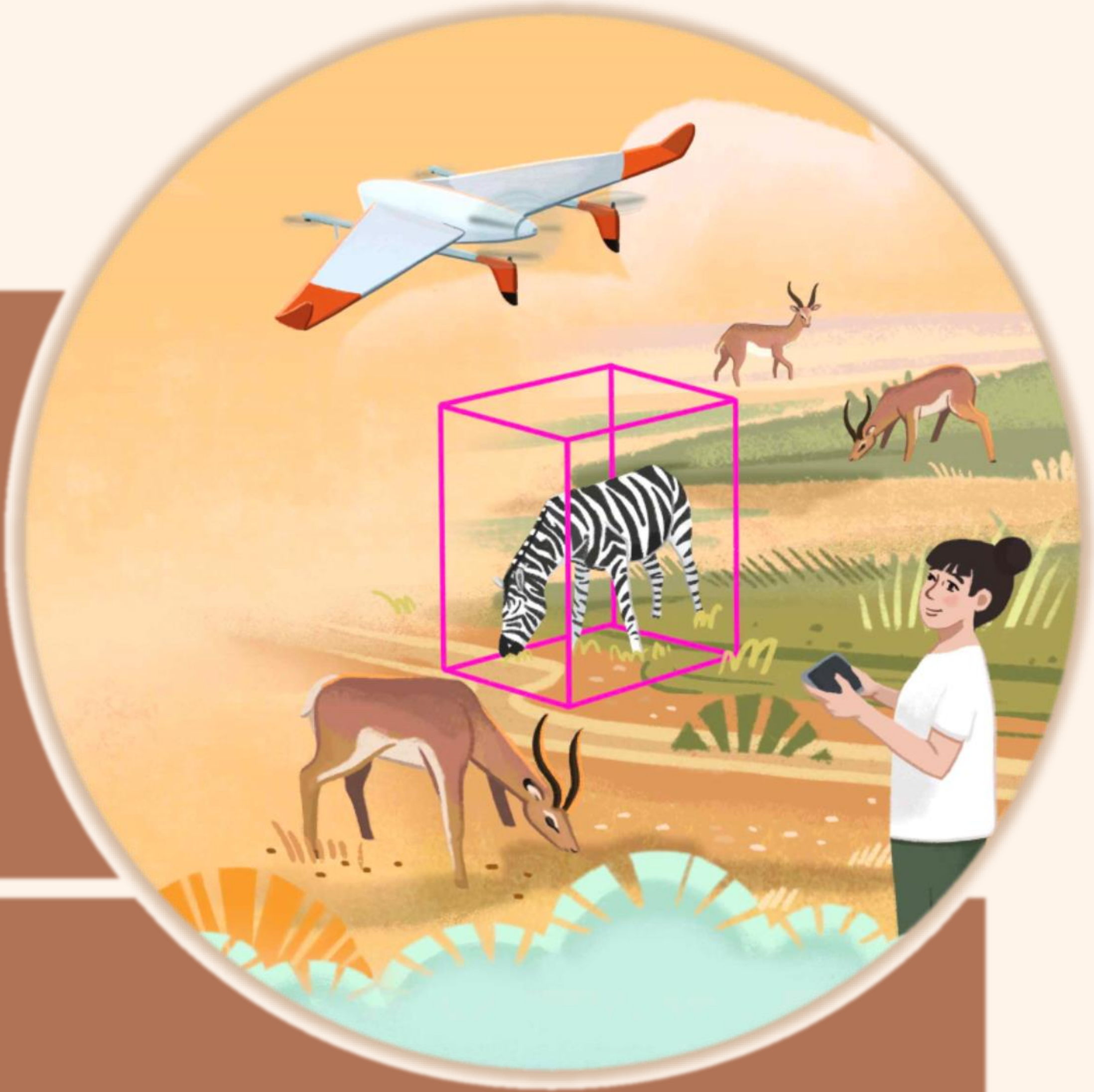
Funded by
the European Union

Mavlink

Kenyatta University, 17.01.2025

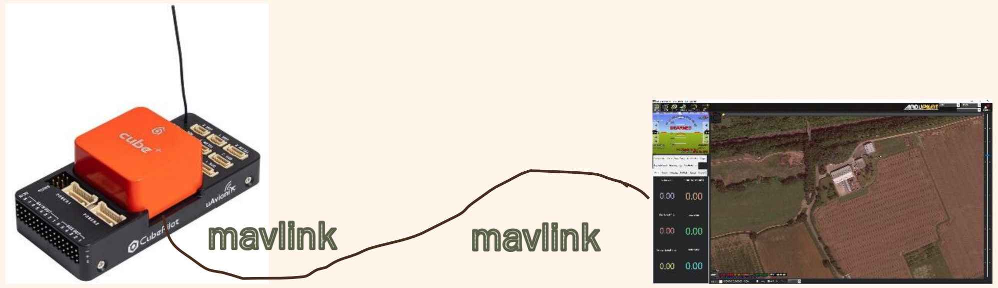
Communicating with your UAV

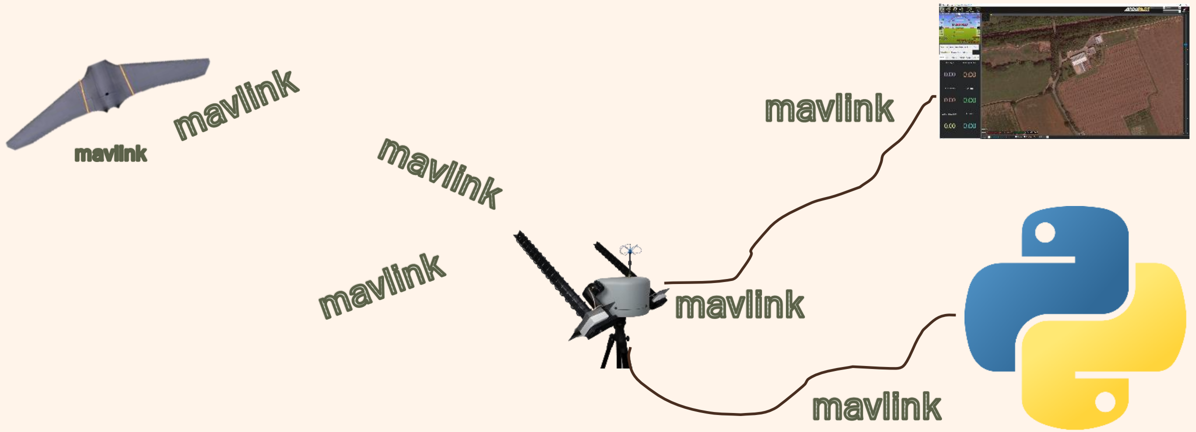


MAVLink

MAVLINK
MICRO AIR VEHICLE COMMUNICATION PROTOCOL

- Open-source messaging protocol for drones
- Use for telemetry and command
- Communication with ground control stations such as Mission Planner
- Wired or wireless connections
- Talk mavlink in Python using Pymavlink
- Pymavlink
  - Request and set parameters
  - Receive state information
  - Send servo control commands
  - Arm/disarm, change flight mode
  - Set waypoints
  - **Everything you can do with MP**

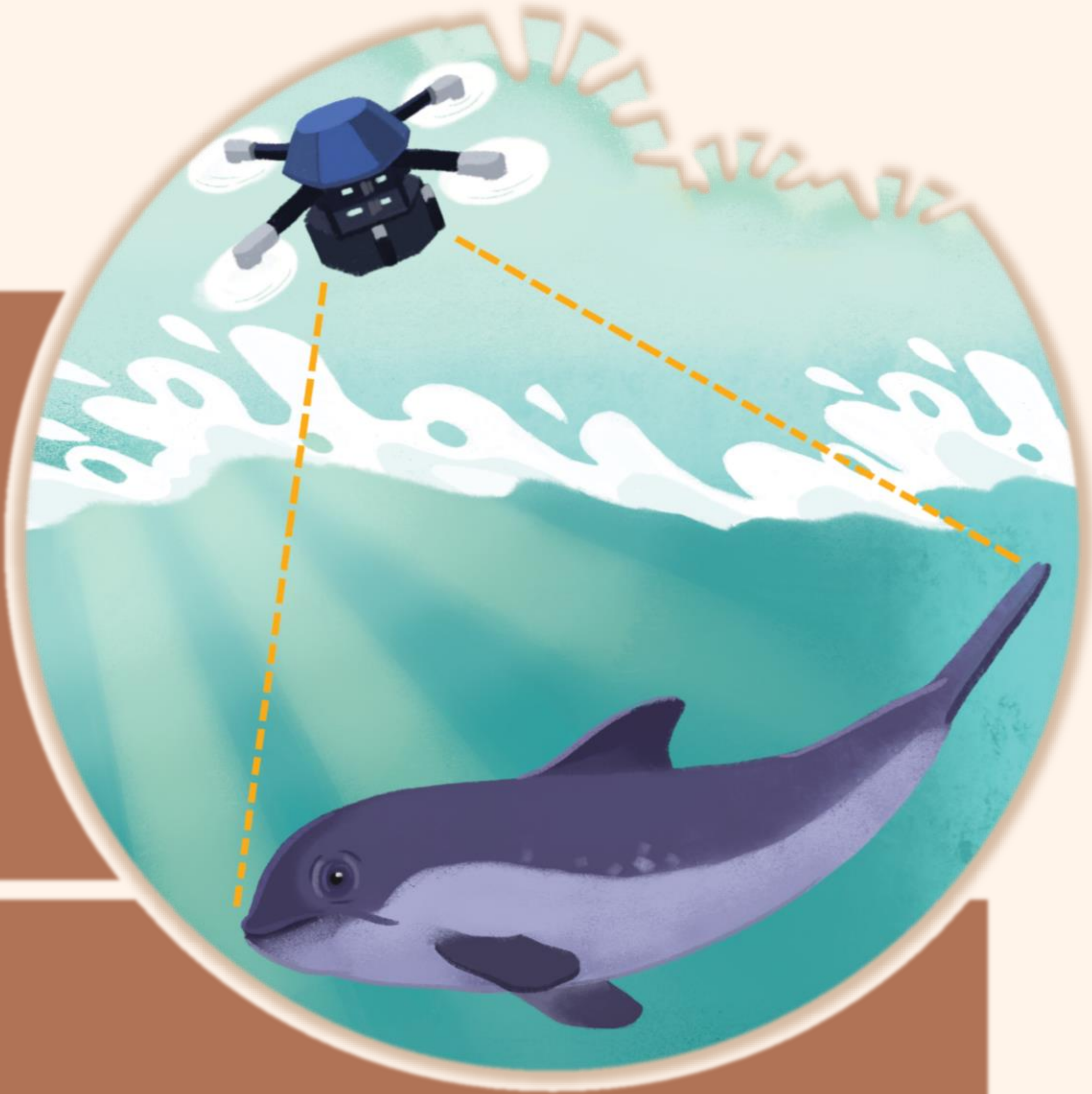




Mavlink Ressources

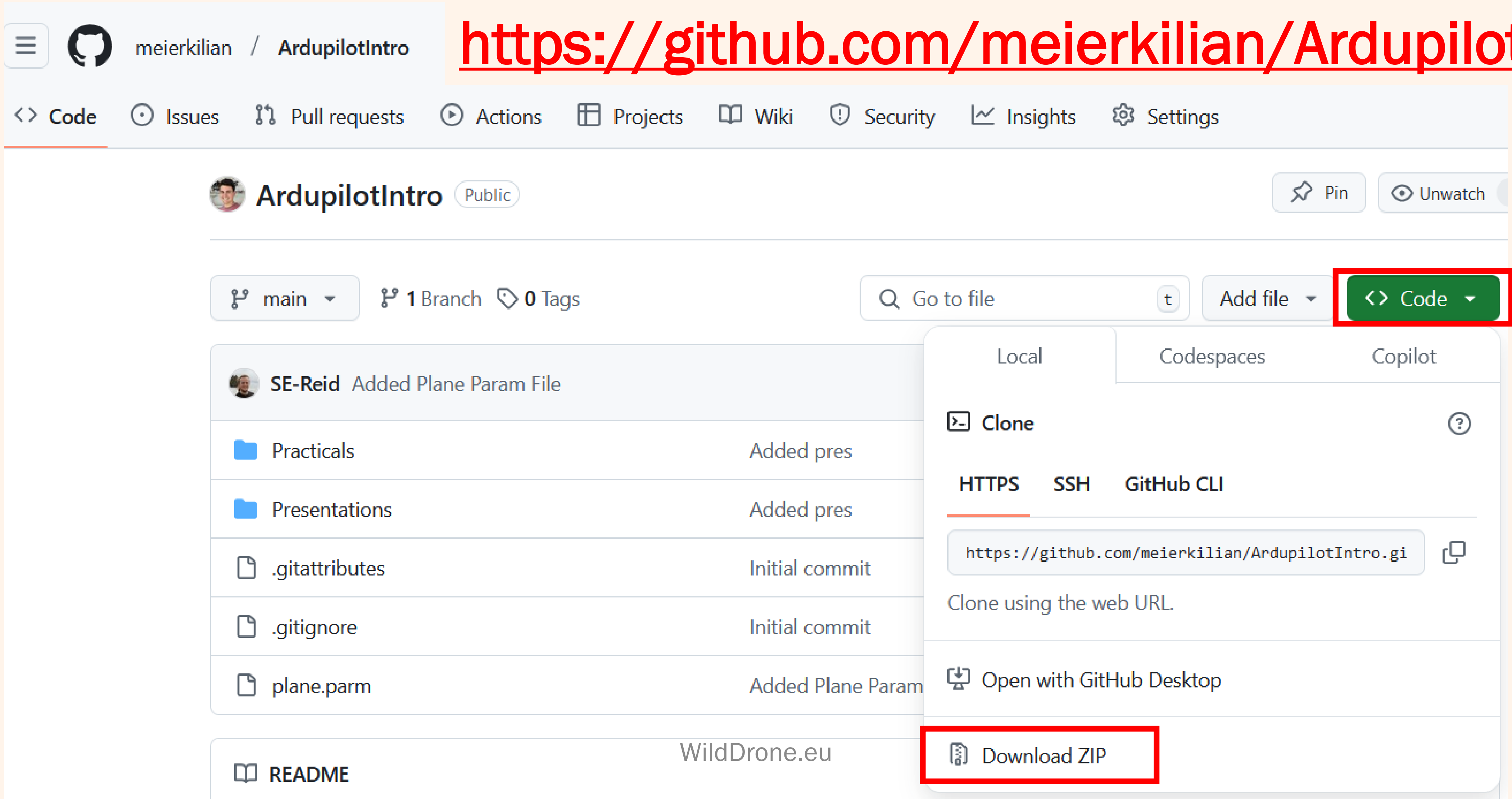
- PyMavlink examples and documentation
  - ArduSub book! ardusub.com/developers/pymavlink.html
- Mavlink Documentation
  - https://mavlink.io/en/messages/common.html
- Today’s presentation and scripts
  - https://github.com/meierkilian/ArdupilotIntro

Practicals



Download today’s scripts

https://github.com/meierkilian/ArdupilotIntro

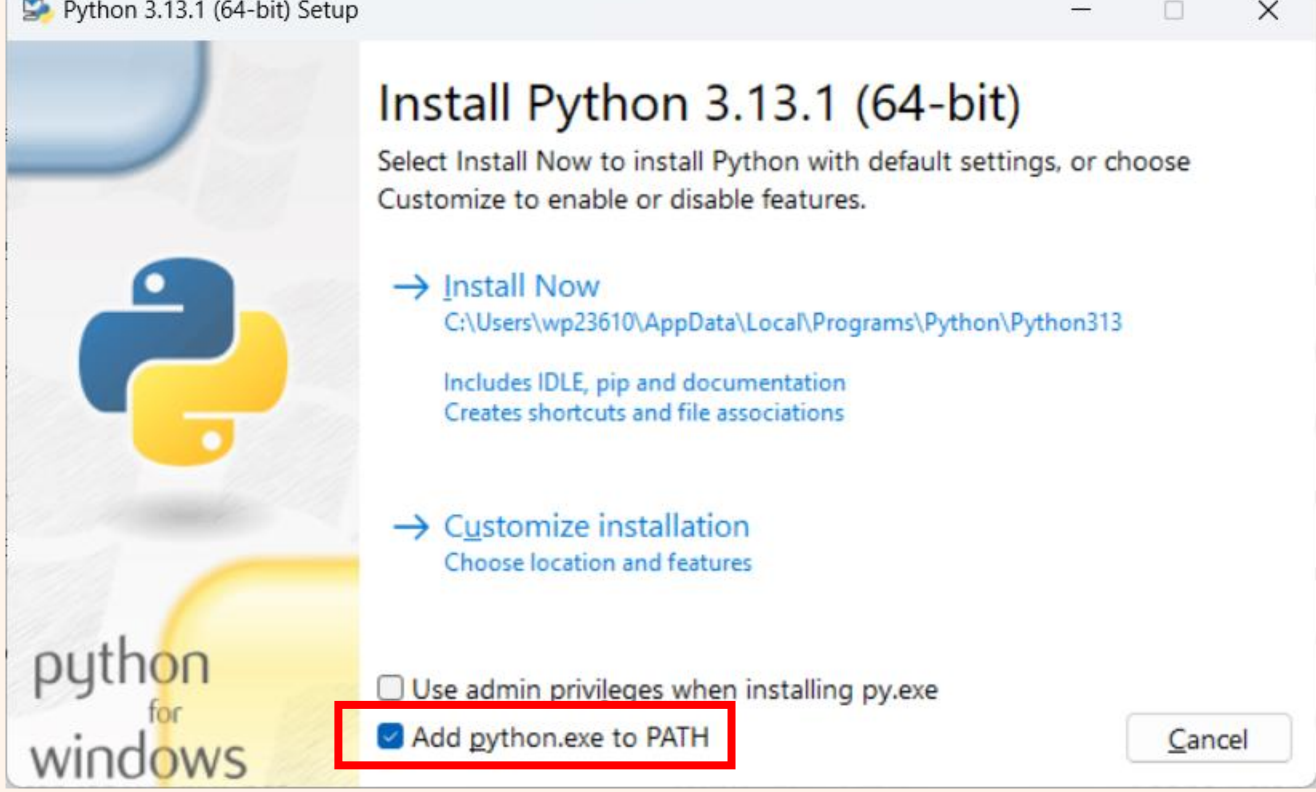


SE-Reid Added Plane Param File

Practicals	Added pres
Presentations	Added pres
.gitattributes	Initial commit
.gitignore	Initial commit
plane.parm	Added Plane Param

Installing python

- Installing python
  - Download Python from: www.python.org/downloads/
  - Add python.exe to PATH
  - Run and click “Install now”
- Installing Pymavlink
  - Open terminal (WIN+R, then type “cmd” and press Enter)
  - Run “pip install pymavlink”
  - Run “pip install requests”
- Hello world script
  - Create a script “helloworld.py”
  - Run “python helloworld.py”


Python 3.13.1 (64-bit) Setup

Install Python 3.13.1 (64-bit)

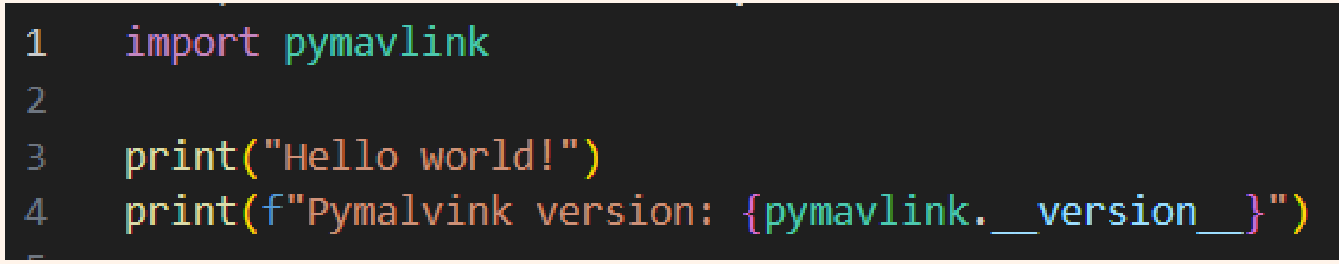
Select Install Now to install Python with default settings, or choose Customize to enable or disable features.

→ Install Now
C:\Users\wp23610\AppData\Local\Programs\Python\Python313

Includes IDLE, pip and documentation
Creates shortcuts and file associations

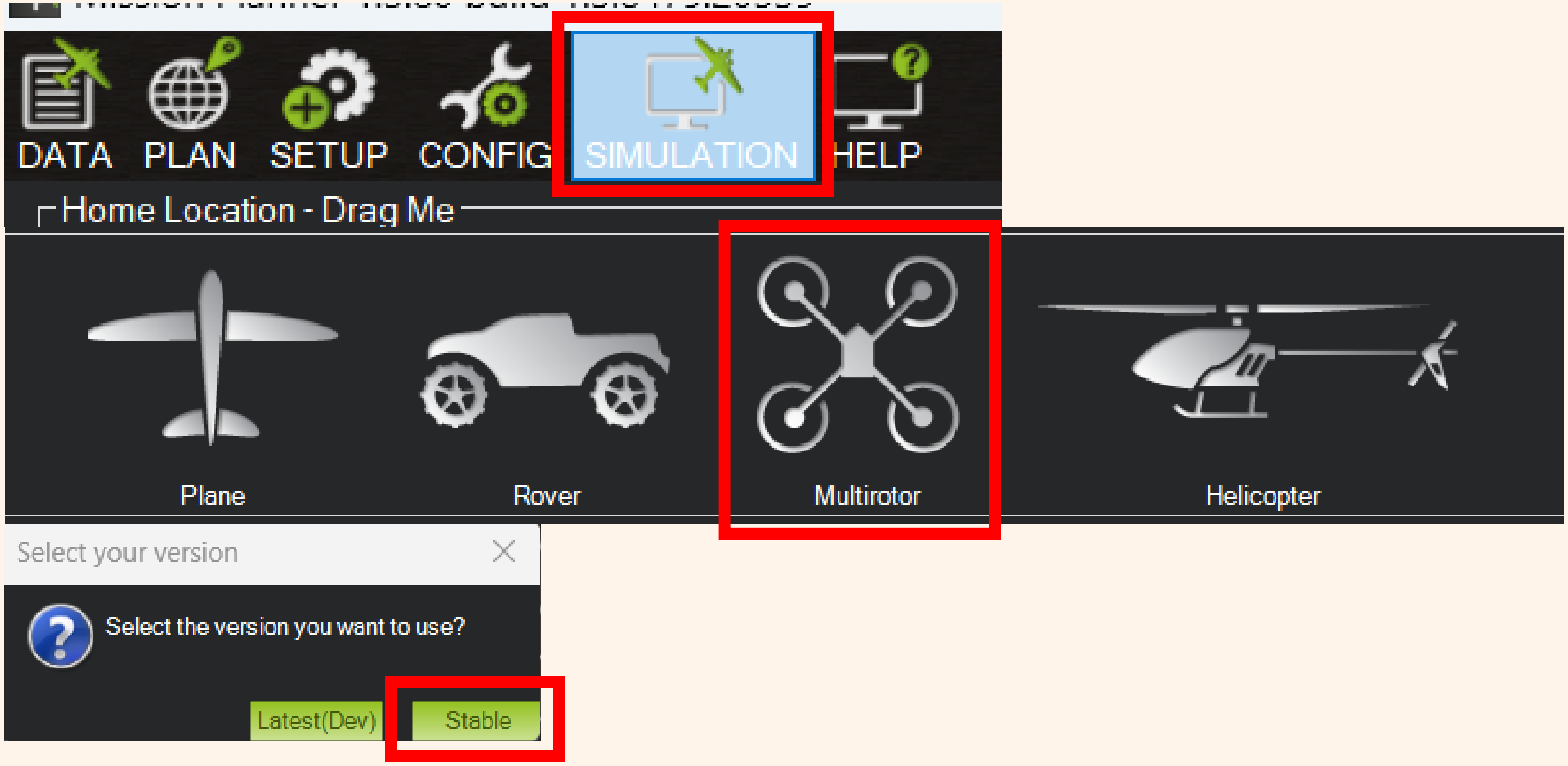
→ Customize installation
Choose location and features

☐ Use admin privileges when installing py.exe
☑ Add python.exe to PATH


```python
import pymavlink

print("Hello world!")
print(f"Pymalvink version: {pymavlink.__version__}")
```

Start SITL

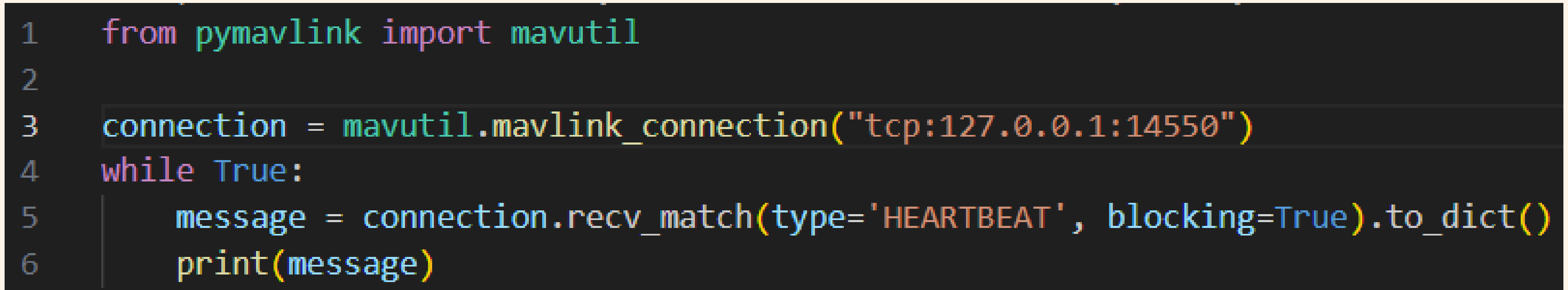


Select your version

Select the version you want to use?

Task 1: connection and heartbeat

- Start SITL
  - Create a quad copter
  - Ctrl + F, Mavlink, TCP Host, Write access, Connect
- Connection
  - Depends on the scenario
  - connection = mavutil.mavlink_connection(‘tcp:127.0.0.1:14550’)
- Heartbeat
  - message = connection.recv_match(type='HEARTBEAT', blocking=True).to_dict()
  - Waits for a message of type HEARTBEAT

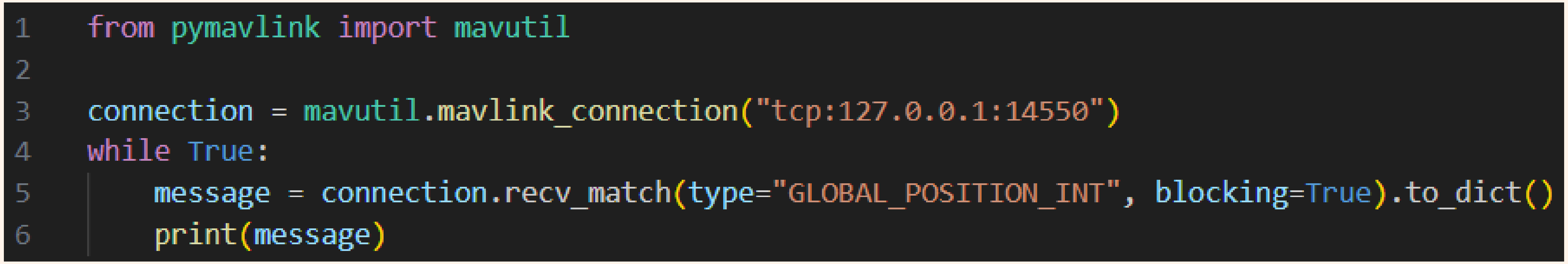


```
from pymavlink import mavutil

connection = mavutil.mavlink_connection("tcp:127.0.0.1:14550")
while True:
    message = connection.recv_match(type='HEARTBEAT', blocking=True).to_dict()
    print(message)
```

Task 2: get telemetry (or any message)

- Have a look at common mavlink message and find a message that will give you the global position of the drone.
  - https://mavlink.io/en/messages/common.html



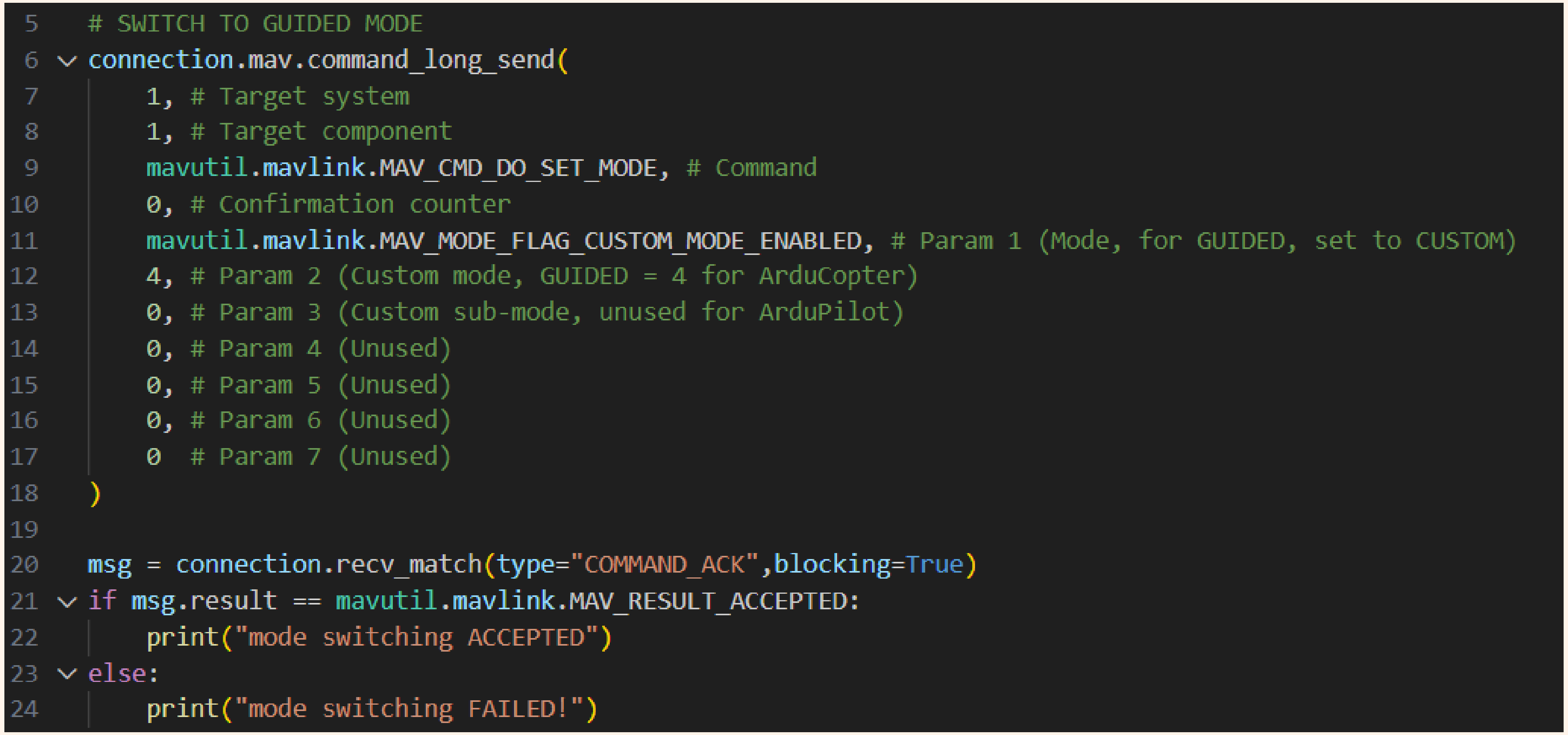
```
from pymavlink import mavutil

connection = mavutil.mavlink_connection("tcp:127.0.0.1:14550")
while True:
    message = connection.recv_match(type="GLOBAL_POSITION_INT", blocking=True).to_dict()
    print(message)
```

Task 3: taking-off

- Taking-off includes three steps
  - Change the aircraft to the GUIDED mode
  - ARM the aircraft
  - Take-off
- So three commands have to be sent!

Task 3: taking-off: mode change

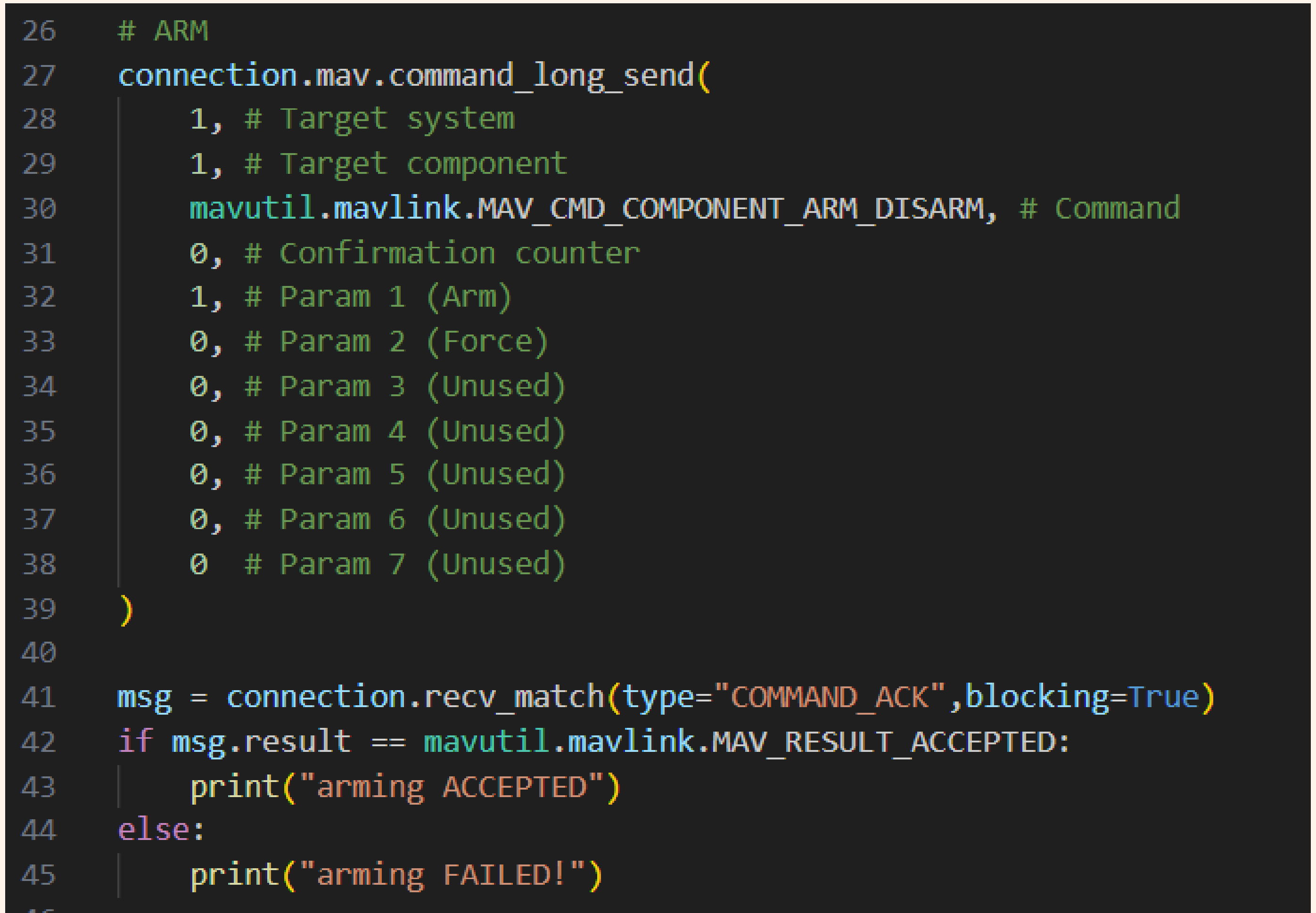


```python
# SWITCH TO GUIDED MODE
connection.mav.command_long_send(
    1, # Target system
    1, # Target component
    mavutil.mavlink.MAV_CMD_DO_SET_MODE, # Command
    0, # Confirmation counter
    mavutil.mavlink.MAV_MODE_FLAG_CUSTOM_MODE_ENABLED, # Param 1 (Mode, for GUIDED, set to CUSTOM)
    4, # Param 2 (Custom mode, GUIDED = 4 for ArduCopter)
    0, # Param 3 (Custom sub-mode, unused for ArduPilot)
    0, # Param 4 (Unused)
    0, # Param 5 (Unused)
    0, # Param 6 (Unused)
    0  # Param 7 (Unused)
)

msg = connection.recv_match(type="COMMAND_ACK",blocking=True)
if msg.result == mavutil.mavlink.MAV_RESULT_ACCEPTED:
    print("mode switching ACCEPTED")
else:
    print("mode switching FAILED!")
```

Task 3: taking-off: arming

- Command name: MAV_CMD_COMPONENT_ARM_DISARM

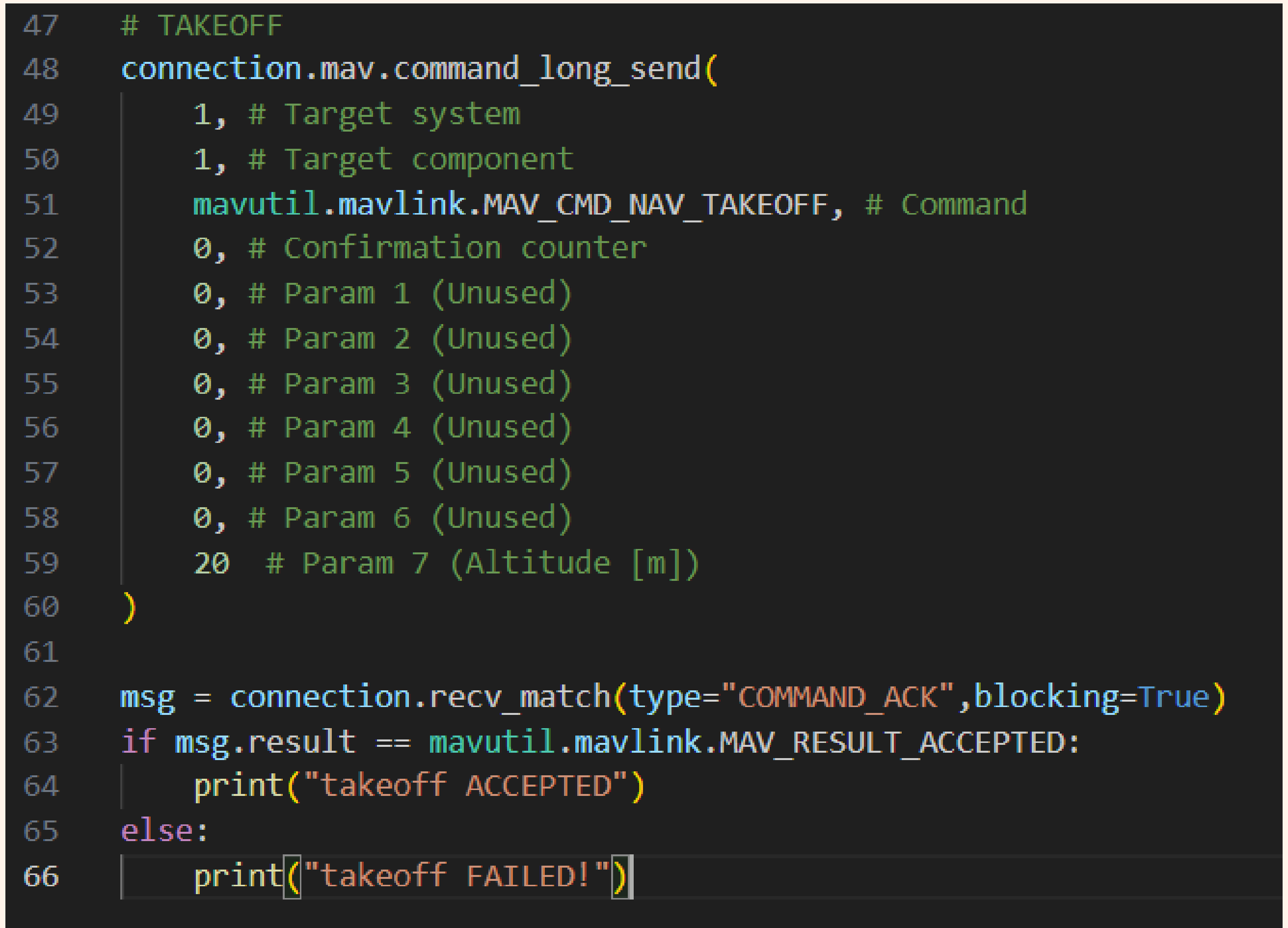


```python
# ARM
connection.mav.command_long_send(
    1, # Target system
    1, # Target component
    mavutil.mavlink.MAV_CMD_COMPONENT_ARM_DISARM, # Command
    0, # Confirmation counter
    1, # Param 1 (Arm)
    0, # Param 2 (Force)
    0, # Param 3 (Unused)
    0, # Param 4 (Unused)
    0, # Param 5 (Unused)
    0, # Param 6 (Unused)
    0  # Param 7 (Unused)
)

msg = connection.recv_match(type="COMMAND_ACK",blocking=True)
if msg.result == mavutil.mavlink.MAV_RESULT_ACCEPTED:
    print("arming ACCEPTED")
else:
    print("arming FAILED!")
```

Task 3: taking-off: take-off

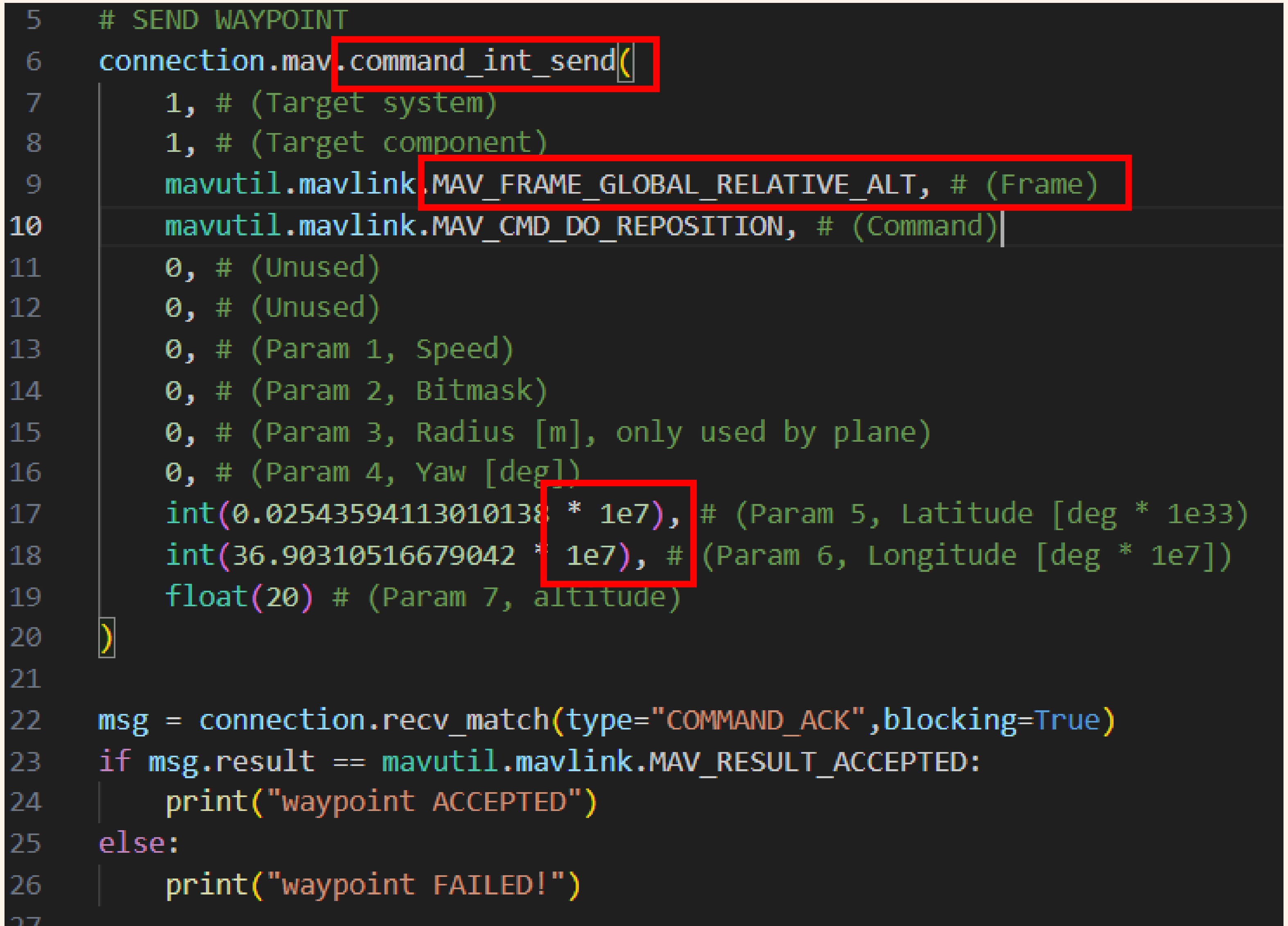
- Command name: MAV_CMD_NAV_TAKEOFF



```python
# TAKEOFF
connection.mav.command_long_send(
    1, # Target system
    1, # Target component
    mavutil.mavlink.MAV_CMD_NAV_TAKEOFF, # Command
    0, # Confirmation counter
    0, # Param 1 (Unused)
    0, # Param 2 (Unused)
    0, # Param 3 (Unused)
    0, # Param 4 (Unused)
    0, # Param 5 (Unused)
    0, # Param 6 (Unused)
    20  # Param 7 (Altitude [m])
)

msg = connection.recv_match(type="COMMAND_ACK",blocking=True)
if msg.result == mavutil.mavlink.MAV_RESULT_ACCEPTED:
    print("takeoff ACCEPTED")
else:
    print("takeoff FAILED!")
```

Task 4: send a waypoint command



```
# SEND WAYPOINT
connection.mav.command_int_send(
    1, # (Target system)
    1, # (Target component)
    mavutil.mavlink.MAV_FRAME_GLOBAL_RELATIVE_ALT, # (Frame)
    mavutil.mavlink.MAV_CMD_DO_REPOSITION, # (Command)
    0, # (Unused)
    0, # (Unused)
    0, # (Param 1, Speed)
    0, # (Param 2, Bitmask)
    0, # (Param 3, Radius [m], only used by plane)
    0, # (Param 4, Yaw [deg])
    int(0.02543594113010138 * 1e7), # (Param 5, Latitude [deg * 1e33)
    int(36.90310516679042 * 1e7), # (Param 6, Longitude [deg * 1e7])
    float(20) # (Param 7, altitude)
)

msg = connection.recv_match(type="COMMAND_ACK",blocking=True)
if msg.result == mavutil.mavlink.MAV_RESULT_ACCEPTED:
    print("waypoint ACCEPTED")
else:
    print("waypoint FAILED!")
```

Task 5: send a WP to remote SITL

- Connect to local network
  - SSID: ANVLAN729
  - Password: ministrylevel369
- Change connection to your groupe IP
- Change the “target system” to your group ID

Engineering and Physical Sciences Research Council

Funded by the European Union

Project funded by

Schweizerische Eidgenossenschaft
Confédération suisse
Confederazione Svizzera
Confederaziun svizra

Swiss Confederation

Federal Department of Economic Affairs, Education and Research EAER
State Secretariat for Education, Research and Innovation SERI

WildDrone is an MSCA Doctoral Network funded by the European Union's Horizon Europe research and innovation funding programme under the Marie Skłodowska-Curie grant agreement no. 101071224. Views and opinions expressed are those of the author(s) only and do not necessarily reflect those of the European Union or the European Research Executive Agency. Neither the European Union nor the granting authority can be held responsible for them.

FOLLOW OUR PROGRESS ON SOCIAL MEDIA!

WildDrone.eu

WildDrone.eu

@WildDrone_eu

WildDrone_eu