NANOPIX DRONE programming in Python

Step by step manual
for Windows users

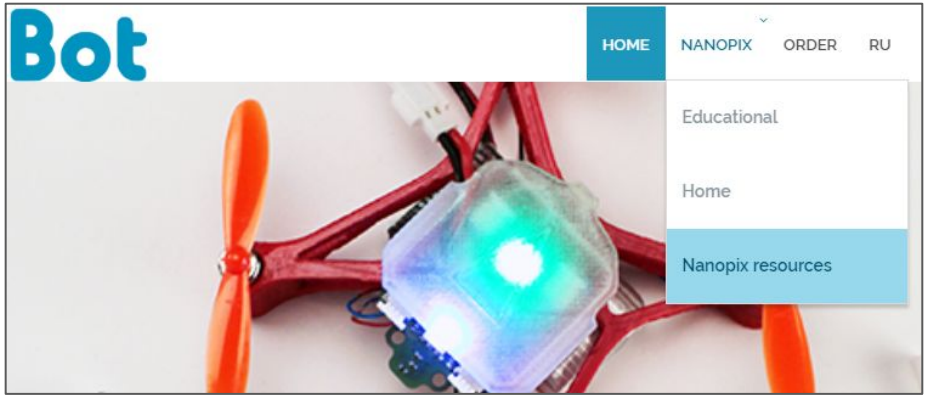
MiniBot

1. Download and install

1.1 Download archive PythonForNanopix.zip via link:
https://www.minibot.tech/programming_tools/PythonForNanopix.zip

1.2 Download archive NanopixGroundStation.zip via link:
https://minibot.tech/programming_tools/NanopixGroundStation.zip

You can also find them on resources page:



1.3 Unpack archive PythonForNanopix.zip

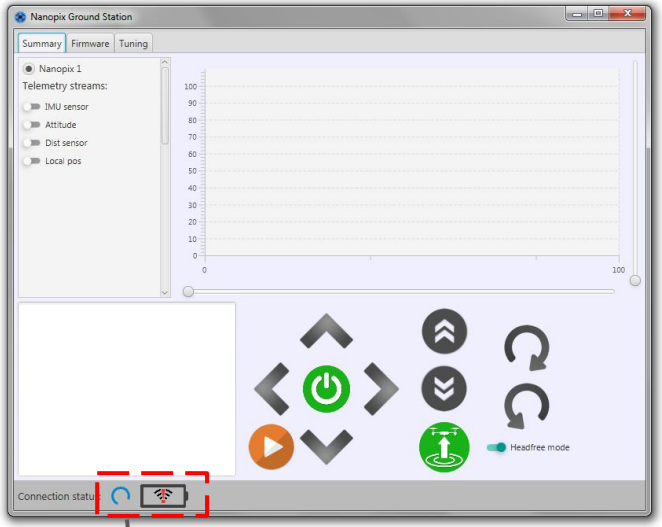
1.4 Unpack archive NanopixGroundStation.zip and run the installer NanopixGroundStation-x.x.exe

1.5 Download and install the tool Thonny
(or choose any other, if you prefer)

https://thonny.org/

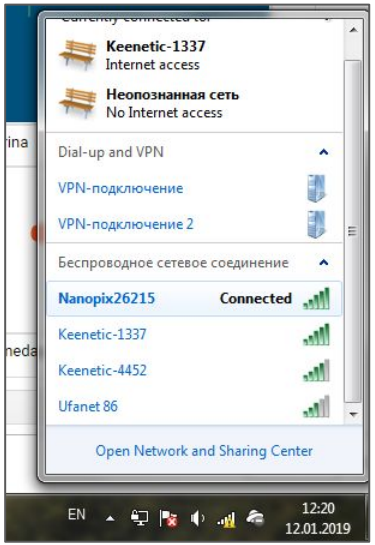
2. NanopixGroundStation setup

2.1 Run installed program **NanopixGroundStation**:



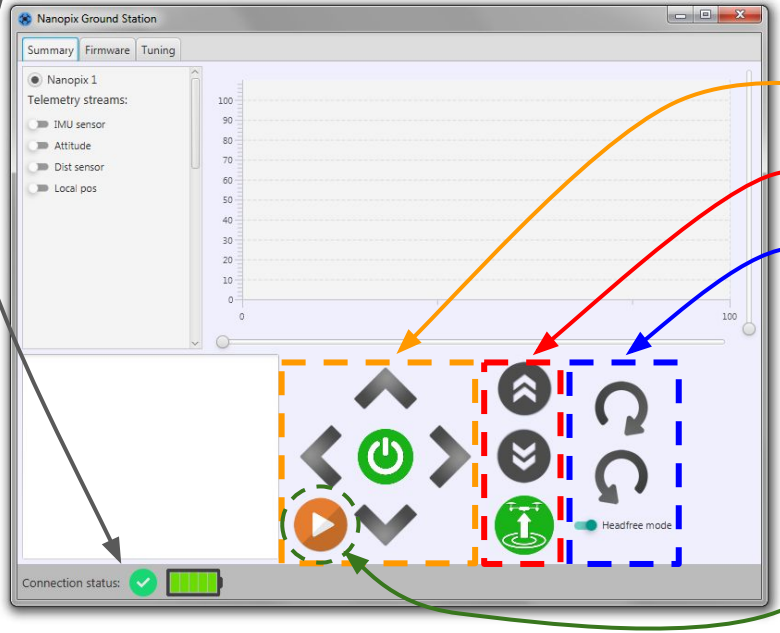
While PC is not connected with Nanopix - you will see “Disconnected” icon here.

2.2 Plug the battery to Nanopix board and connect your PC to Nanopix Wi-Fi network. It will be called like this: Nanopix<number> (default password: 12345678)



2. NanopixGroundStation setup

2.3 After Wi-Fi initialization you will see successfully connection icon in NanopixGroundStation



2.4 Here you can control your NANOPIX, modify parameters, calibrate, flash firmwares, etc.

UI elements for controlling the drone:

- Forward / backward / left / right movements and arm/disarm motors button
- Automatic takeoff/land and altitude control
- Yaw controls and headfree checkbox
- Mission start/stop button (mission = a program that you wrote (in Scratch or Arduino) and flashed to NANOPIX.

For Python - please proceed the instructions following below

3. Thonny IDE set up

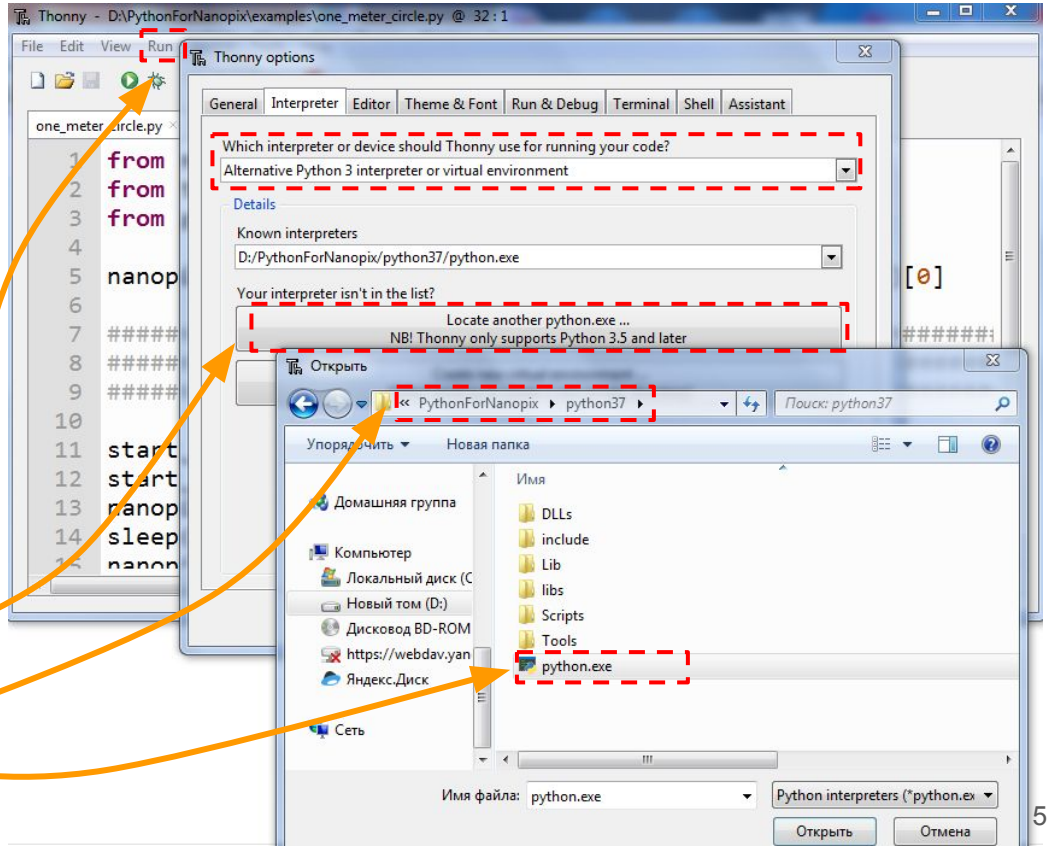
3.1 Run the installed software Thonny and open the file:
<the path, where the archive is extracted>\PythonForNanopix\examples\blink.py

3.2 Change interpreter from default to the next:
<the path, where the archive is extracted>\PythonForNanopix\python37\python.exe

1. For this choose the tab “Run” -> “Select interpreter...”

2. In the following window choose “Locate another python.exe...”

3. Click the button and choose the file “...\PythonForNanopix\python37\python.exe”



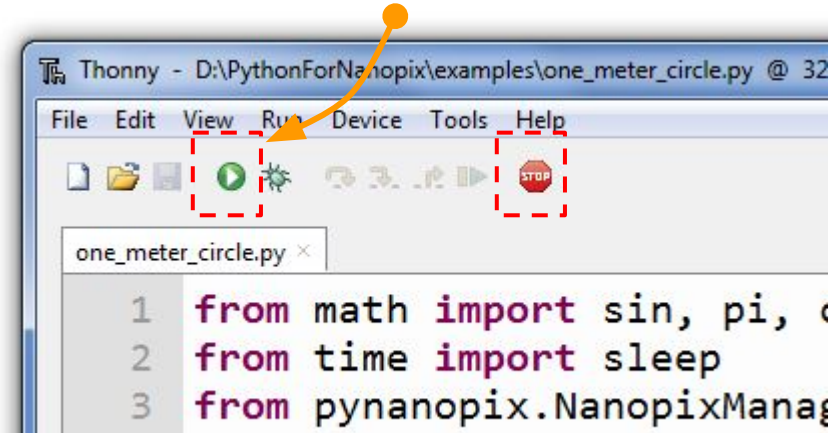
4. Running the program

4.1 Make sure your PC is connected to the Wi-Fi of the drone Nanopix <number> (default password 12345678)

4.2 Example from the file blink.py will make the front LED of the drone blink:

```
6  ##############################################
7  ######################## Script for Nanopix:
8  ##############################################
9
10 nanopix.set_back_led(0, 0, 0)
11 for i in range(0, 15):
12     nanopix.set_front_led(R=255, G=0, B=0)
13     print('RED')
14     sleep(1)
15     nanopix.set_front_led(R=0, G=0, B=255)
16     print('BLUE')
17     sleep(1)
```

4.3 To run the script press green Play button:



To stop the script while it's running press the STOP button.

During 1-2 seconds after you press it, the drone will start the automatic landing (if it was in the air), independent from if there was a program running previously.

Troubleshooting

The smartphone is losing Wifi connection with NANOPIX drone all the time
(Android 8+ frequent problem)

Before Wi-Fi connection disable next things:

1. mobile data
2. hotspot Wi-Fi point

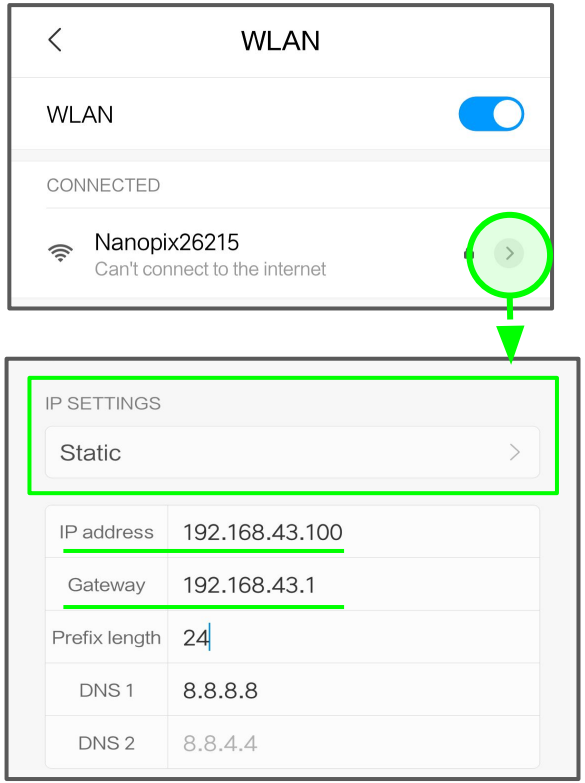
This should fix it:

- If your smartphone loses internet-less connections, try to disable such behavior in Wi-Fi settings.

Or / and

- Try to setup static IP for NANOPIX Wi-Fi network Nanopix<number>: change `DHCP` to `Static`. Enter the next IP and Gateway:
  - IP address: 192.168.43.100
  - Gateway: 192.168.43.1

1. Open Wi-Fi settings and setup static IP address:



API description

Controlling the integrated RGB LEDs:

`set_front_led(R: int, G: int, B: int)`

`set_back_led(R: int, G: int, B: int)`

Sets the color value for the front/rear LED.

R, G, B - color channels with the range of values [0..255]

Usage example: ...\PythonForNanopix\examples\blink.py

Run or stop the motors:

`arm_motors()` - run the motors

`disarm_motors()` - stop the motors

Usage example: ...\PythonForNanopix\examples\arm_disarm.py

- `get_roll() -> float` - returns the current **roll** angle in degrees **[-180, 180]**
- `get_pitch() -> float` - returns the current **pitch** angle in degrees **[-180, 180]**
- `get_yaw() -> float` - returns the current **yaw** angle in degrees **[-180, 180]**

Usage example:
...\PythonForNanopix\examples\roll_led.py

API description

Automatic takeoff and landing:

takeoff() - takes off the drone to a height of 60 centimeters
land() - automatic landing (from any height) and engines turn off

Usage example: ...\PythonForNanopix\examples\takeoff_land.py

Yaw angle control functions:

- **rotate_yaw**(yaw_offset_deg: int) - rotate drone by yaw (in range [-180, 180]) **relative** with angle speed (in range [6, 55]). Default speed: 35 deg/sec. Positive angle ~ clockwise rotation.
- **set_yaw**(yaw_value_deg: int) - **set absolute yaw angle** in degrees (in range [-180, 180]) with angle speed (in range [6, 55]). Default speed: 35 deg/sec.

Usage example: ...\PythonForNanopix\examples\takeoff_yaw_rotate_land.py

API description

Altitude control:

- **offset_altitude**(delta_meters: float) - - change target altitude relative (in range [-3.0, 3.0]) based on current altitude with target speed (in range [0.1, 0.75]). Default speed: 0.2 m/s.
- **set_altitude**(z_pos: float) - **set absolute altitude value** in meters (in range [0.15, 3.0]) with target speed (in range [0.1, 0.75]). Default speed: 0.2 m/s.

  Usage example: ...\PythonForNanopix\examples\altitude_control.py

- **get_pos_z**() -> float - returns current altitude position in meters

  Usage example: ...\PythonForNanopix\examples\altitude_led.py

API description

Coordinates based movement:

- **offset_xy_pos**(dx_meters: float, dy_meters: float) - relative movement (in range [-5.0, 5.0]) based on current position with target speed (in range [0.1, 1.0]). Default speed: 0.4 m/s.
- **set_xy_pos**(x_meters: float, y_meters: float) - **set absolute position** in meters with target speed (in range [0.1, 1.0]). Default speed: 0.4 m/s.
- **move_forward**(meters: float) - high level function, moves the drone forward (using **offsetTargetPos** inside)
  Same simple functions: **moveBackward**, **moveLeft**, **moveRight**.
- **get_pos_x**() -> float - returns current X frame-based coordinate system position in meters
- **get_pos_y**() -> float - returns current Y frame-based coordinate system position in meters

Usage example: ...\PythonForNanopix\examples\one_meter_square.py

API description

Manual control functions (like sticks controlling):

- **set_pitch**(pitch_deg: int, pitch_only: bool = true) - set pitch angle in degrees in the range [-35, 35]. **pitch_only** option enables special mode: all other movements (roll, yaw or altitude) will be stopped. Enabled by default.
- **set_roll**(roll_deg: int, roll_only: bool = true) - set roll angle in degrees in range [-35, 35]. **roll_only** option enables special mode: all other movements (pitch, yaw or altitude) will be stopped. Enabled by default.
- **set_yaw_speed**(speed_deg_sec: float) - set yaw rotation speed (around vertical axis) in degrees/second
- **set_altitude_speed**(altitude_speed: float) - set vertical speed in range [-1...1]. Value -1 is equivalent to downward movement with 30 cm/sec vertical speed. Value 1 is equivalent to upward movement with the same vertical speed.

Important: these functions do not wait for the end of the maneuver (non-blocking) and are active until another function is called. To stop movement call, for example: `setYawSpeed(0);`

Usage example: ...\PythonForNanopix\examples\manual_control_square.py