EV3 Python

Release 2020

Apr 04, 2020

Contents:

CHAPTER 1

Introduction to the EV3

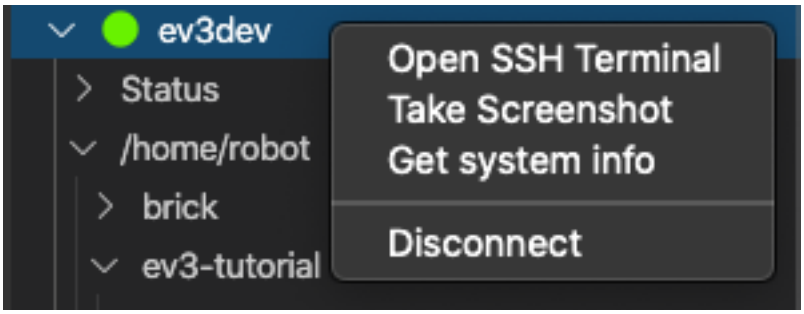
1.1 Installation

Flollow the instructions on the LEGO Education site to use Python on your EV3

- download the microSD image to your computer
- flash the image to the microSD card using a tool such as Etcher
- insert the microSD card into EV3 brick
- download the **VS Code** editor to your compouter
- install the EV3 extensions

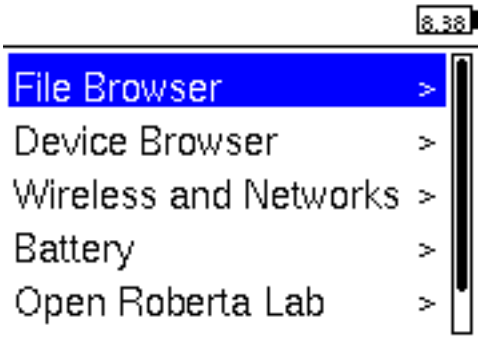
1.2 Get system information

In the **EV3DEV device browser** you have access to a context menu.



It allows you to :

- open the SSH Terminal (entering automatically the password)
- take a (color) screenshop of the EV3 display



This is the system info you get:

```
========== ev3dev-sysinfo ==========
Image file:         ev3-micropython-v1.0.0-sd-card-image
Kernel version:     4.14.96-ev3dev-2.3.2-ev3
Brickman:           0.10.0
BogoMIPS:           148.88
Bluetooth:
Board:              board0
BOARD_INFO_HW_REV=8
BOARD_INFO_MODEL=LEGO MINDSTORMS EV3
BOARD_INFO_ROM_REV=6
BOARD_INFO_SERIAL_NUM=001653601922
BOARD_INFO_TYPE=main
```

1.3 Connect to the EV3 using ssh

You can connect to EV3 brick remoteyl via a SSH terminal. Click in the **EV3DEV device browser** to connect to the EV3. Open a terminal and connect via SSH to `robot@ev3dev.local`

The password is **maker**:

```
user@MacBook-Air brick % ssh robot@ev3dev.local
Warning: Permanently added the ECDSA host key for IP address 'fe80::16:53ff:fe60:1922
↪%en4' to the list of known hosts.
Password:
Linux ev3dev 4.14.96-ev3dev-2.3.2-ev3 #1 PREEMPT Sun Jan 27 21:27:35 CST 2019␣
↪armv5tejl
            _____     _
_____   _|___ /  __| | _____   __
/ _ \ \ / / |_ \ / _` |/ _ \ \ / /
|  __/\ V / ___) | (_| |  __/\ V /
\___| \_/ |____/ \__,_|\___| \_/

Debian stretch on LEGO MINDSTORMS EV3!
robot@ev3dev:~$
```

1.4 Execute Linux commands

You can print the working directory:

```
pwd
/home/robot
```

Display the list of current folders:

```
ls
brick  getting_started  sensors
```

Change directory to **brick** and display its content:

```
cd brick/
ls
battery.py  brick.rst  button.py  display2.py  display.py  main.py  sound2.py  sound.
↪py
```

1.5 Run a Python session

Run a Python session:

```
python3
Python 3.5.3 (default, Sep 27 2018, 17:25:39)
[GCC 6.3.0 20170516] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

Execute some Python commands:

```
>>> print('hello world')
hello world
>>> 99**12
886384871716129280658801
```

1.6 Text to speech

From the command line you can start text-to-speech:

```
espeak "hello, I am an EV3.
> I like to talk because I am a robot.
> Did you know that robots like to make sounds?
> Beep. Boop. Dit. Dit. Meep.
> I am just such a chatterbox." --stdout | aplay
```

1.7 Update the system

You can update the Debian operating system:

```
sudo apt-get update

We trust you have received the usual lecture from the local System
```

(continues on next page)

(continued from previous page)

```
Administrator. It usually boils down to these three things:

    #1) Respect the privacy of others.
    #2) Think before you type.
    #3) With great power comes great responsibility.

[sudo] password for robot:
```

1.8 Demo example

This is a demo example for a simple Python program.

```
#!/usr/bin/env pybricks-micropython

from pybricks import ev3brick as brick
from pybricks.ev3devices import Motor
from pybricks.parameters import Port

# Play a sound.
brick.sound.beep()

# Initialize a motor at port B.
test_motor = Motor(Port.B)

# Run the motor up to 500 degrees per second. To a target angle of 90 degrees.
test_motor.run_target(500, 90)

# Play another beep sound.
# This time with a higher pitch (1000 Hz) and longer duration (500 ms).
brick.sound.beep(1000, 500)
```

1.9 Import classes and methods

These are all the useful classes and methods.

```
#!/usr/bin/env pybricks-micropython

from pybricks import ev3brick as brick
from pybricks.ev3devices import (Motor, TouchSensor, ColorSensor,
                                 InfraredSensor, UltrasonicSensor, GyroSensor)
from pybricks.parameters import (Port, Stop, Direction, Button, Color,
                                 SoundFile, ImageFile, Align)
from pybricks.tools import print, wait, StopWatch
from pybricks.robotics import DriveBase
```

1.10 Micro-python vs real Python

The first line of the program, the so-called **shebang**, indicates to the EV3 which Python it is going to use. LEGO proposes the use of Micropython, which is starting up slightly faster:

```
#!/usr/bin/env pybricks-micropython
```

To use real Python put this on your first line:

```
#!/usr/bin/env python3
```

You get bigger fonts and get text-to-speech.

```
#!/usr/bin/env python3
from ev3dev.ev3 import *
import os

os.system('setfont Lat15-TerminusBold14')
L = LargeMotor('outB'); mL.stop_action = 'hold'
R = LargeMotor('outC'); mR.stop_action = 'hold'

msg = 'Hello, my name is EV3!'
print(msg)
Sound.speak(msg).wait()

L.run_to_rel_pos(position_sp= 840, speed_sp = 250)
R.run_to_rel_pos(position_sp=-840, speed_sp = 250)
L.wait_while('running')
R.wait_while('running')
```

Sources

- https://sites.google.com/site/ev3devpython/
- https://www.udemy.com/course/ev3-python/

CHAPTER 2

EV3 Brick

This section shows how to program the buttons, lights, sounds and display.

2.1 Buttons

This program associates 3 button presses with 3 different colors:

- LEFT - green
- CENTER - yellow
- RIGHT - red

```
#!/usr/bin/env pybricks-micropython
from pybricks import ev3brick as brick
from pybricks.parameters import Button, Color

while True:
    b = brick.buttons()
    if Button.LEFT in b:
        brick.light(Color.GREEN)

    elif Button.CENTER in b:
        brick.light(Color.YELLOW)

    elif Button.RIGHT in b:
        brick.light(Color.RED)

    else:
        brick.light(None)
```

2.2 Sound

This program uses the 4 buttons to change volume and frequency.

- LEFT/RIGHT - change volume from 0 to 100 in increments of 10
- UP/DOWN - change frequency in increments of 10

```
#!/usr/bin/env pybricks-micropython
from pybricks import ev3brick as brick
from pybricks.parameters import Button

# up/down buttons to change frequency
freq = 500
volume = 30

while True:
    buttons = brick.buttons()
    if Button.UP in buttons:
        freq += 10

    elif Button.DOWN in buttons:
        freq -= 10

    elif Button.LEFT in buttons:
        volume = max(0, volume-10)

    elif Button.RIGHT in buttons:
        volume = min(100, volume+10)

    brick.sound.beep(freq, 300, volume)
```

This program places a couple of sound files into two lists:

- emotions
- numbers

Inside a loop they are played in sequence.

```
#!/usr/bin/env pybricks-micropython
from pybricks import ev3brick as brick
from pybricks.parameters import SoundFile

emotions = ['SHOUTING', 'CHEERING', 'CRYING']
numbers = 'ZERO ONE TWO THREE FOUR FIVE SIX SEVEN EIGHT NINE'.split()

for sound in emotions:
    file = eval('SoundFile.'+sound)
    brick.sound.file(file, 100)

for sound in numbers:
    file = eval('SoundFile.'+sound)
    brick.sound.file(file, 100)
```

2.3 Display

This program displays image files and their name on the screen, during 1 second.

```
#!/usr/bin/env pybricks-micropython
from pybricks import ev3brick as brick
from pybricks.parameters import ImageFile
from pybricks.tools import print, wait

images = 'RIGHT FORWARD ACCEPT QUESTION_MARK STOP_1 LEFT DECLINE \
    THUMBS_DOWN BACKWARD NO_GO WARNING STOP_2 THUMBS_UP'.split()

for image in images:
    brick.display.clear()
    brick.display.text(image, (10, 10))
    file = eval('ImageFile.'+image)
    brick.display.image(file, clear=False)
    wait(1000)
```

This program writes a new line of text to screen, every second.

```
#!/usr/bin/env pybricks-micropython
from pybricks import ev3brick as brick
from pybricks.parameters import (Port, Stop, Direction, Button, Color,
                                 SoundFile, ImageFile, Align)
from pybricks.tools import print, wait

images = 'RIGHT FORWARD ACCEPT QUESTION_MARK STOP_1 LEFT DECLINE \
    THUMBS_DOWN BACKWARD NO_GO WARNING STOP_2 THUMBS_UP'.split()

for image in images:
    brick.display.text(image.lower())
    wait(1000)
```

2.4 Battery

This program displays the battery voltage and current during 5 seconds.

```
#!/usr/bin/env pybricks-micropython
from pybricks import ev3brick as brick
from pybricks.tools import print, wait

voltage = brick.battery.voltage()
current = brick.battery.current()

brick.display.text('voltage = {} mV'.format(voltage))
brick.display.text('current= {} mA'.format(current))

wait(5000)
```

CHAPTER 3

Motor

3.1 Run at a fix Duty cycle

In this example the motor runs at a duty-cycle from -100% to +100%.

```
#!/usr/bin/env pybricks-micropython
from pybricks import ev3brick as brick
from pybricks.ev3devices import Motor
from pybricks.parameters import Port, Button
from pybricks.tools import print, wait

motor = Motor(Port.B)
cycle = 50

while True:
    bts = brick.buttons()
    if Button.LEFT in bts:
        cycle = max(-100, cycle-10)

    elif Button.RIGHT in bts:
        cycle = min(100, cycle+10)

    elif Button.CENTER in bts:
        break

    motor.dc(cycle)
    print(cycle, motor.speed(), motor.angle())
    wait(100)
```

3.2 Run at a fix speed

In this mode the motor uses feedback action to keep the speed constant.

```
#!/usr/bin/env pybricks-micropython
from pybricks import ev3brick as brick
from pybricks.ev3devices import Motor
from pybricks.parameters import Port, Button
from pybricks.tools import print, wait

motor = Motor(Port.B)
speed = 100

while True:
    bts = brick.buttons()
    if Button.LEFT in bts:
        speed = max(-1000, speed-100)

    elif Button.RIGHT in bts:
        speed = min(1000, speed+100)

    elif Button.CENTER in bts:
        break

    motor.run(speed)
    print(speed, motor.speed(), motor.angle())
    wait(100)
```

3.3 Run for a specified time

```
#!/usr/bin/env pybricks-micropython
from pybricks import ev3brick as brick
from pybricks.ev3devices import Motor
from pybricks.parameters import Port, Button, Stop
from pybricks.tools import print, wait

motor = Motor(Port.B)

while True:
    bts = brick.buttons()
    if Button.RIGHT in bts:
        motor.run_time(200, 3000, Stop.COAST, False)

    elif Button.CENTER in bts:
        break

    print(motor.speed(), motor.angle())
    wait(100)
```

3.4 Run for a specified angle

```
#!/usr/bin/env pybricks-micropython
from pybricks import ev3brick as brick
from pybricks.ev3devices import Motor
from pybricks.parameters import Port, Button, Stop
from pybricks.tools import print, wait

motor = Motor(Port.B)

while True:
    bts = brick.buttons()
    if Button.RIGHT in bts:
        motor.run_angle(200, 500, Stop.COAST, False)

    elif Button.CENTER in bts:
        break

    print(motor.speed(), motor.angle())
    wait(100)
```

3.5 Track an angle

```
#!/usr/bin/env pybricks-micropython
from pybricks import ev3brick as brick
from pybricks.ev3devices import Motor
from pybricks.parameters import Port, Button, Stop
from pybricks.tools import print, wait, StopWatch
import math

motor = Motor(Port.B)
watch = StopWatch()
amplitude = 90

while True:
    bts = brick.buttons()

    t = watch.time()/1000
    angle = math.sin(t) * amplitude
    motor.track_target(angle)

    if Button.CENTER in bts:
        break
```

CHAPTER 4

Indices and tables

- genindex
- modindex
- search