

# LEGO EV3 PROGRAMMING

16级计科基地班

刘安

pacer97@qq.com

2018/10

# Content

- Intro
- Examples
- Videos

# Intro to EV3

- LEGO MINDSTORMS EV3 is based on a brick
- includes an ARM®9-based processor, micro SD card reader, and USB port for Wi-Fi connectivity.
- It connects to a variety of sensors, such as ultrasound, color/light, gyroscope, and touch.
- It also connects to up to four servo motors, so you can use it to build mobile robots.

# Brick



	EV3	NXT	RCX
Release Date	September 2013	July 2006	1998
Display	178×128 pixel Monochrome LCD	100×64 pixel Monochrome LCD	<a href="#">segmented</a> Monochrome LCD
Main Processor	<a href="#">TI Sitara AM1808</a> ( <a href="#">ARM926EJ-S</a> core) @300 MHz	<a href="#">Atmel AT91SAM7S256</a> ( <a href="#">ARM7TDMI</a> core) @48 MHz	<a href="#">Hitachi H8/300</a> @16 MHz
Main Memory	64 MB RAM 16 MB Flash <a href="#">microSDHC</a> Slot	64 KB RAM 256 KB Flash	32 KB RAM 16 KB ROM
<a href="#">USB</a> Host Port	Yes	No	No
WiFi	Optional dongle via USB port	No	No
Bluetooth	Yes	Yes	No
Connects to Apple devices	Yes	No	No

# Sensors



触动传感器



颜色传感器



远程红外信标



红外/超声波传感器

# Program with Matlab

- % LEGOEV3 - The class to represent the LEGO EV3 brick
- % You can use LEGOEV3 to interact with EV3 brick.
- %
- % myev3 = legoev3('usb')
- % set up USB connection between host and EV3 brick.
- %
- % myev3 = legoev3('wifi', ip\_address, hardware\_id)
- % set up WiFi connection between host and EV3 brick
- % For example:
- % myev3 = legoev3('wifi', '192.168.1.7', '00165340e49b')
- %
- % myev3 = legoev3('bluetooth', com\_port)
- % set up Bluetooth connection between host and EV3 brick
- % For example:
- % myev3 = legoev3('bluetooth', 'COM19')
- %

```
>> mylego = legoev3('usb')
```

```
mylego =
```

legoev3 - 属性:

```
FirmwareVersion: 'V1.09H'  
HardwareID: []  
IPAddress: []  
CommunicationType: 'USB'  
BatteryLevel: 100  
ConnectedSensors: {'' 'gyro' 'color' 'sonic'}
```

# Demo1-Control Sensors

- %%
- % Create a connection to the gyroscopic sensor called mygyrosensor.
- % Measure the rotation, in degrees,
- % since the creation of the connection to the sensor.
- % Reset the measurement to zero.
- % Measure the current rate of rotation, in degrees per second.
- %%
- myev3 = legoev3;
- mygyrosensor = gyroSensor(myev3);
- angle = readRotationAngle(mygyrosensor);
- resetRotationAngle(mygyrosensor);
- rate = readRotationRate(mygyrosensor);

# Demo2-Collision Alarm Template

- `%----- Change ME -----`
- `mylego = legoev3; % Set up MATLAB and EV3 communication`
- `RANGE = 0.3; % Detection range in meters`
- `%-----`
- `mysensor = sonicSensor(mylego); % Set up ultrasonic sensor`
- `%% Operations %%%%%%%%%%`
- `while ~readButton(mylego, 'up') % Exit if UP button is pressed`
- `dis = readDistance(mysensor); % Read ultrasonic sensor value`
- `freq = 5000*(RANGE-dis)/RANGE; % Increase frequency as getting closer`
- `volume = 100*(RANGE-dis)/RANGE; % Increase volume as getting closer`
- `playTone(mylego, freq, 1, volume); % Play tone`
- `end`



# Demo3-How to Go Straight

- `mylego = legoev3; % Set up MATLAB and EV3 communication`
- `mymotor1 = motor(mylego, 'B'); % Set up motor`
- `mymotor2 = motor(mylego, 'C');`
- `% Application parameters`
- `EXE_TIME = 10; % Application running time in seconds`
- `SPEED = 30; % Motor speed`
- `mymotor1.Speed = SPEED; % Set motor speed`
- `mymotor2.Speed = SPEED;`
- `start(mymotor1); % Start motor`
- `start(mymotor2);`
- `pause(EXE_TIME); % Wait`
- `stop(mymotor1); % Stop motor`
- `stop(mymotor2);`

# Demo3-How to Go Straight (closeloop)

- `%% setup %%%`
- `mylego = legoev3; % Set up MATLAB and EV3 communication`
- `% Change based on your motor port numbers`
- `mymotor1 = motor(mylego, 'B'); % Set up motor`
- `mymotor2 = motor(mylego, 'C');`
- `% Application parameters`
- `EXE_TIME = 10; % Application running time in seconds`
- `PERIOD = 0.1; % Sampling period`
- `SPEED = 20; % Motor speed`
- `P = 0.01; % P controller parameter`
- `%-----`
- `mymotor1.Speed = SPEED; % Set motor speed`
- `mymotor2.Speed = SPEED;`
- `resetRotation(mymotor1); % Reset motor rotation counter`
- `resetRotation(mymotor2);`
- `start(mymotor1); % Start motor`
- `start(mymotor2);`
- `t = timer('TimerFcn', 'stat=false;', 'StartDelay', EXE_TIME);`
- `start(t);`

- `%% Operations %%`
- `stat = true;`
- `lastR1 = 0;`
- `lastR2 = 0;`
- `while stat == true % Quit when times up`
- `r1 = readRotation(mymotor1); % Read rotation counter in degrees`
- `r2 = readRotation(mymotor2);`
- `speed1 = (r1 - lastR1)/PERIOD; % Calculate the real speed in d/s`
- `speed2 = (r2 - lastR2)/PERIOD;`
- `diff = speed1 - speed2; % P controller`
- `mymotor1.Speed = mymotor1.Speed - int8(diff * P);`
- `lastR1 = r1;`
- `lastR2 = r2;`
- `pause(PERIOD); % Wait for next sampling period`
- `end`
- `%% Clean up %%`
- `stop(mymotor1); % Stop motor`
- `stop(mymotor2);`

# ev3dev is your EV3 *re-imagined*

ev3dev is a **Debian Linux**-based operating system that runs on several LEGO® MINDSTORMS compatible platforms including the **LEGO® MINDSTORMS EV3** and **Raspberry Pi**-powered **BrickPi**.

Just like you can take apart your LEGO® models and build something completely different, we have reverse-engineered the EV3 and created a new software platform for programming your robots.



## 🔧 EV3 programming unlocked

ev3dev gives you the power to program how you want to. We have created a low-level driver framework for controlling sensors, motors and pretty much everything else. It's as easy as reading from and writing to a file.

ev3dev supports many popular scripting languages out-of-the-box, so you can get started right away with your favorite language and libraries.

## 🐧 Backed by the full power of Linux

Since ev3dev is built on Debian Linux, there are over 43,000 free software packages available for you to install.

And with the Linux kernel at its core, many USB and Bluetooth devices, like Wi-Fi dongles, keyboards, keypads, joysticks and cameras work too.

## ⬇️ It's not firmware

It's more like dual-boot. ev3dev runs from a microSD card and doesn't ever touch the firmware installed on the EV3. To switch back, you just shut down and remove the microSD card - no flashing required.



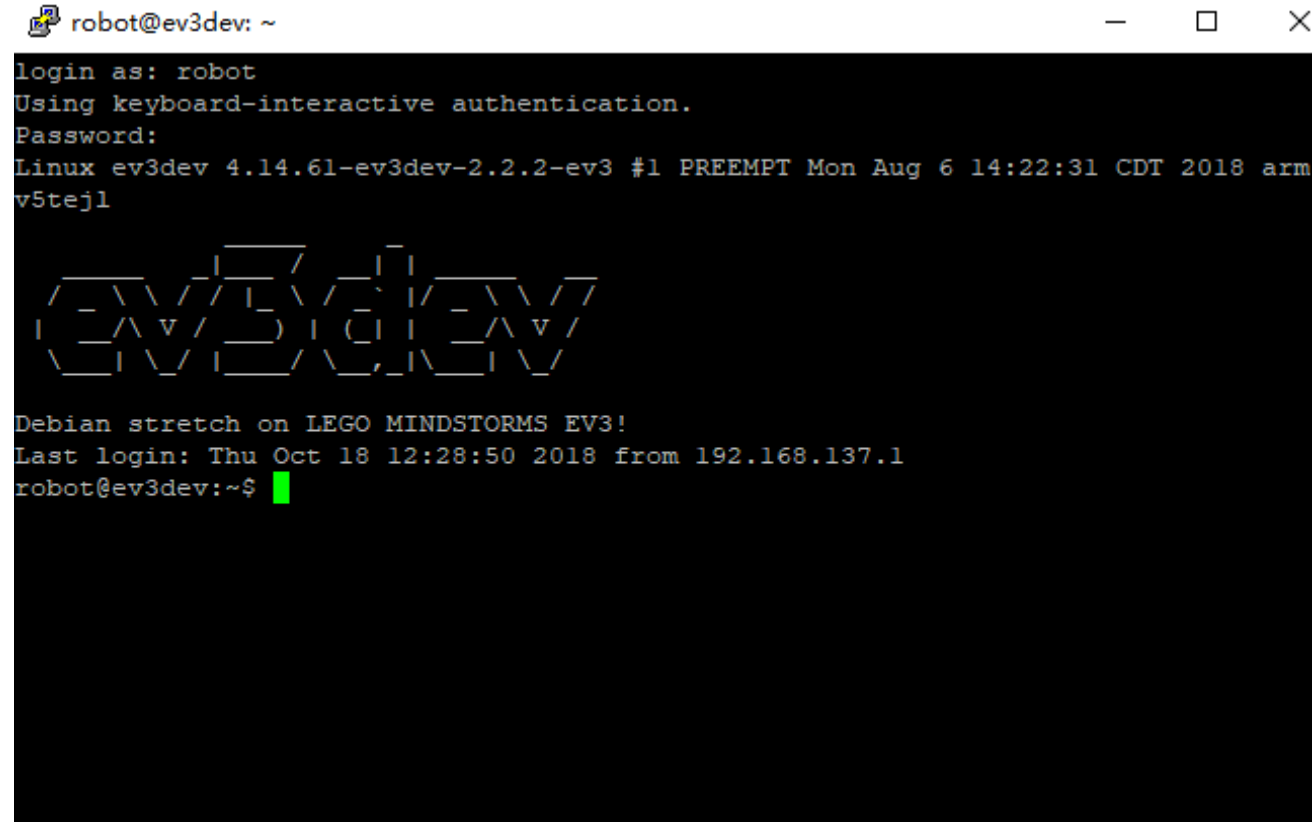
# Program in other Languages

- Python
- JavaScript
- C/C++
- Go
- Ruby
- Java
- .....

# Need to Prepare

- A LEGO MINDSTORMS EV3 Intelligent Brick or Raspberry Pi (any model).
- A microSD or microSDHC card (2GB or larger). microSDXC is not supported on the EV3. **All cards larger than 32GB will not work with the EV3!**
- A computer with an adapter for the SD card. You will need administrator user permissions on this computer.
- USB Wi-Fi dongle

# Demo- Connecting to Ev3dev Using SSH



```
robot@ev3dev: ~  
login as: robot  
Using keyboard-interactive authentication.  
Password:  
Linux ev3dev 4.14.61-ev3dev-2.2.2-ev3 #1 PREEMPT Mon Aug 6 14:22:31 CDT 2018 armv5tejl  
  
ev3dev  
  
Debian stretch on LEGO MINDSTORMS EV3!  
Last login: Thu Oct 18 12:28:50 2018 from 192.168.137.1  
robot@ev3dev:~$
```

# Program in Python

- `#!/usr/bin/env python3`
- `from ev3dev2.motor import LargeMotor, OUTPUT_A, OUTPUT_B, SpeedPercent, MoveTank`
- `from ev3dev2.sensor import INPUT_1`
- `from ev3dev2.sensor.lego import TouchSensor`
- `from ev3dev2.led import Leds`
- `# TODO: Add code here`
- `sound = Sound()`
- `sound.speak('Welcome to the E V 3 dev project!')`
- `tank_drive = MoveTank(OUTPUT_A, OUTPUT_B)`
- `# drive in a turn for 5 rotations of the outer motor`
- `# the first two parameters can be unit classes or percentages.`
- `tank_drive.on_for_rotations(SpeedPercent(50), SpeedPercent(75), 10)`
- `# drive in a different turn for 3 seconds`
- `tank_drive.on_for_seconds(SpeedPercent(60), SpeedPercent(30), 3)`

# Demo-Color

- `#!/usr/bin/env python3`
- `"""Make robot say whatever color it observes with the color sensor."""`
- `from ev3dev2.sensor.lego import ColorSensor`
- `from time import sleep`
- `from ev3dev2.sound import Sound`
- `color_sensor = ColorSensor()`
- `sound = Sound()`
- `while True:`
- `color = color_sensor.color`
- `text = ColorSensor.COLORS[color]`
- `sound.speak(text)`
- `sleep(2)`



# Demo-Touch

```
• #!/usr/bin/env python3  
• """  
• Reverse robot if bumps into wall.  
• This script is a simple demonstration of the touch sensor.  
• """  
• from ev3dev2.motor import MoveSteering, OUTPUT_B, OUTPUT_C  
• from ev3dev2.sensor.lego import TouchSensor  
• motor_pair = MoveSteering(OUTPUT_B, OUTPUT_C)  
• touch_sensor = TouchSensor()  
• # Start robot moving forward  
• motor_pair.on(steering=0, speed=10)  
• # Wait until robot touches wall  
• touch_sensor.wait_for_pressed()  
• # Stop moving forward  
• motor_pair.off()  
• # Reverse away from wall  
• motor_pair.on_for_seconds(steering=0, speed=-10, seconds=2)
```

# Demo-Ultrasonic

```
• #!/usr/bin/env python3
• """
• Use robot to reposition cuboid.
• This script is a simple demonstration of the ultrasonic
  sensor and medium
• motor.
• """
• from ev3dev2.motor import (
  MoveSteering, MediumMotor, OUTPUT_A, OUTPUT_B, OUTPUT_C)
• from ev3dev2.sensor.lego import UltrasonicSensor
• from time import sleep
• motor_pair = MoveSteering(OUTPUT_B, OUTPUT_C)
• medium_motor = MediumMotor(OUTPUT_A)
• ultrasonic_sensor = UltrasonicSensor()
• # Start robot moving forward
•
• motor_pair.on(steering=0, speed=10)
• # Wait until robot less than 3.5cm from cuboid
• while ultrasonic_sensor.distance_centimeters > 3.5:
• sleep(0.01)
• # Stop moving forward
• motor_pair.off()
• # Lower robot arm over cuboid
• medium_motor.on_for_degrees(speed=-10, degrees=90)
• # Drag cuboid backwards for 2 seconds
• motor_pair.on_for_seconds(steering=0, speed=-20, seconds=2)
• # Raise robot arm
• medium_motor.on_for_degrees(speed=10, degrees=90)
• # Move robot away from cuboid
• motor_pair.on_for_seconds(steering=0, speed=-20, seconds=2)
```

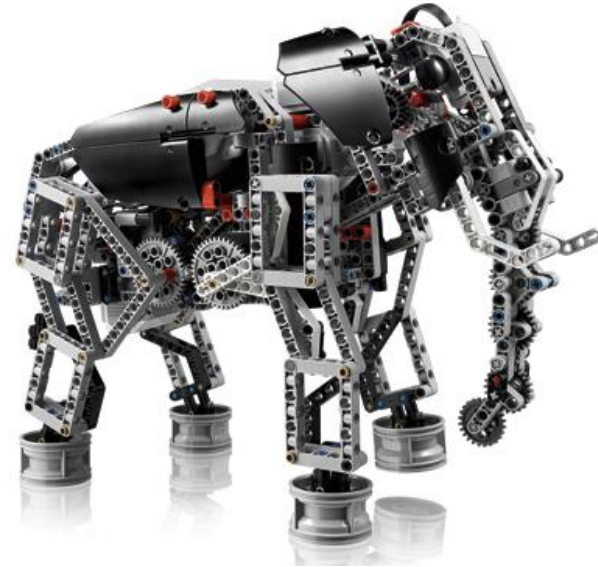
# Demo-Square

- `#!/usr/bin/env python3`
- `"""`
- Move robot in a square path without using the Gyro sensor.
- This script is a simple demonstration of moving forward and turning.
- `"""`
- `from ev3dev2.motor import MoveSteering, OUTPUT_B, OUTPUT_C`
- `motor_pair = MoveSteering(OUTPUT_B, OUTPUT_C)`
- `for i in range(4):`
- `# Move robot forward for 3 seconds`
- `motor_pair.on_for_seconds(steering=0, speed=50, seconds=3)`
- `# Turn robot left 90 degrees (adjust rotations for your particular robot)`
- `motor_pair.on_for_rotations(steering=-100, speed=5, rotations=0.5)`

# Demo-Square-gyro

```
• #!/usr/bin/env python3  
• """  
• Move robot in a square path using the Gyro sensor.  
• This script is a simple demonstration of turning using the Gyro sensor.  
• """  
• from ev3dev2.motor import MoveSteering, OUTPUT_B, OUTPUT_C  
• from ev3dev2.sensor.lego import GyroSensor  
• motor_pair = MoveSteering(OUTPUT_B, OUTPUT_C)  
• gyro = GyroSensor()  
• gyro.mode = GyroSensor.MODE_GYRO_ANG  
• for i in range(4):  
• # Move robot forward for 3 seconds  
• motor_pair.on_for_seconds(steering=0, speed=50, seconds=3)  
• # Spin robot to the left  
• motor_pair.on(steering=-100, speed=5)  
• # Wait until angle changed by 90 degrees  
• gyro.wait_until_angle_changed_by(90)  
• # Stop motors  
• motor_pair.off()
```

# Usage



# Vedios

- <https://www.youtube.com/watch?v=Z4Lz2rYRipQ>
- <https://www.bilibili.com/video/av5740892/>

# Reference

- [https://sites.google.com/site/ev3python/learn\\_ev3\\_python/going-further/auto-drive](https://sites.google.com/site/ev3python/learn_ev3_python/going-further/auto-drive)
- <https://www.ev3dev.org/docs/getting-started/>
- <https://github.com/ev3dev/ev3dev-lang-python>
- <https://python-ev3dev.readthedocs.io/en/ev3dev-stretch/motors.html#units>
- <https://github.com/ev3dev/ev3dev-lang-python-demo>
- <https://github.com/sshopov/pyconau2017/blob/master/final.ipynb>

# THANKS

Q&A