Programming paradigms for physical computing and IoT

Ben Nuttall
Raspberry Pi Foundation
UK Charity 1129409
GPIO Pins – General Purpose Input/Output
from gpiozero import LED

led = LED(2)

led.blink()
GPIO Zero – a friendly API for GPIO devices

- Zero-boilerplate Pythonic library
- Intended for use in education
- Simple, guessable API with commonly used names and sensible default values
- Simple introduction, smooth learning curve
- Multi-paradigm
- Extendable
GPIO Zero supports...
from gpiozero import LED, Button

led = LED(17)
button = Button(4)

while True:
    if button.is_pressed:
        led.on()
    else:
        led.off()
from gpiozero import LED, Button

led = LED(17)
button = Button(4)

while True:
    button.wait_for_press()
    led.on()
    button.wait_for_release()
    led.off()
from gpiozero import LED, Button

led = LED(17)
button = Button(4)

button.when_pressed = led.on
button.when_released = led.off
from gpiozero import LED, Button

led = LED(17)
button = Button(4)

led.source = button.values
```python
>>> led = PWMLED(17)
>>> led.value
0.0
>>> led.on()
>>> led.value
1.0
>>> led.value = 0
```
>>> led = PWMLED(17)
>>> pot = MCP3008()
>>> led.value
0.0
>>> pot.value
0.510145879738202
>>> led.value = pot.value
>>> while True:
...    led.value = pot.value
Source / Values

Output Device
.value
.values
.source

Input Device
.value
.values
from gpiozero import LED, Button

led = LED(17)
button = Button(2)

led.source = button.values
Processing values

Output Device

.value
.values
.source

function

Input Device

.value
.values
from gpiozero import Button, LED
from gpiozero.tools import negated

led = LED(4)
btn = Button(17)

led.source = negated(btn.values)
Source tools – single source conversions

- absoluted
- booleanized
- clamped
- inverted
- negated
- post_delayed
- post_periodic_filtered
- pre_delayed
- pre_periodic_filtered
- quantized
- queued
- smoothed
- scaled
Combining values

Output Device
.value
.values
.source

Source tool

Input Device
.value
.values

Input Device
.value
.values
Source tools – combining sources

• all_values
• any_values
• averaged
• multiplied
• summed
Artificial values

Output Device

- value
- values
- source

function
Source tools – artificial sources

- alternating_values
- cos_values
- ramping_values
- random_values
- sin_values
Internal Devices

- TimeOfDay
- CPUTemperature
- PingServer
- More coming soon
- Make your own!
from gpiozero import Energenie, TimeOfDay
from datetime import time

lamp = Energenie(1)
daytime = TimeOfDay(time(9), time(18))

lamp.source = daytime.values
from gpiozero import LEDBarGraph, CPUTemperature

cpu = CPUTemperature(min_temp=50, max_temp=90)
leds = LEDBarGraph(2, 3, 4, 5, 6, 7, 8, pwm=True)

leds.source = cpu.values
Is the internet working?

```python
from gpiozero import LED, PingServer
from gpiozero.tools import negated

green = LED(17)
red = LED(18)
goog = PingServer('google.com')

green.source = goog.values
green.source_delay = 60
red.source = negated(green.values)
```
from gpiozero import InternalDevice

class FileReader(InternalDevice):
    @property
    def value(self):
        with open('value.txt') as f:
            return int(f.read().strip())
Blue Dot
from gpiozero import LED
from bluedot import BlueDot

led = LED(17)
bd = BlueDot()

while True:
    if bd.is_pressed:
        led.on()
    else:
        led.off()
from gpiozero import LED
from bluedot import BlueDot

led = LED(17)
bd = BlueDot()

while True:
    bd.wait_for_press()
    led.on()
    bd.wait_for_release()
    led.off()
from gpiozero import LED
from bluedot import BlueDot

led = LED(17)
bd = BlueDot()

bd.when_pressed = led.on
bd.when_released = led.off
from gpiozero import LED
from bluedot import BlueDot

led = LED(17)
bd = BlueDot()

led.source = bd.values
GPIO Zero: cross-platform – distributed via apt/pip

- **Raspberry Pi**
  - Raspbian, Debian, Ubuntu, etc

- **PC & Mac**
  - Raspberry Pi Desktop x86
  - Linux
  - Mac OS
  - Windows
Supporting multiple back-ends

- **RPi.GPIO**
  - Low-level GPIO library, implemented in C (current default)

- **RPIO**
  - Low-level GPIO library, implemented in C (only supports Pi 1)

- **pigpio**
  - Low-level GPIO library, implemented in C
  - Runs as a daemon on the Pi, can accept remote commands

- **Native**
  - Pure Python, limited functionality, experimental (included in gpiozero)

- **Mock**
  - Pure Python, used in test suite, useful for testing (included in gpiozero)
$ GPIOZERO_PIN_FACTORY=mock python3
>>> from gpiozero import LED
>>> led = LED(22)
>>> led.blink()
>>> led.value
True
>>> led.value
False
>>> from gpiozero import LED, Button
>>> led = LED(22)
>>> button = Button(23)
>>> led.source = button.values
>>> led.value
False
>>> button.pin.drive_low()
>>> led.value
True
$ GPIOZERO_PIN_FACTORY=pigpio PIGPIO_ADDR=192.168.0.2 python3 led.py

from gpiozero import LED

led = LED(22)

led.blink()
from gpiozero import LED
from gpiozero.pins.pigpio import PiGPIOFactory

factory = PiGPIOFactory('192.168.0.2')

led = LED(22, pin_factory=factory)

led.blink()
from gpiozero import LED, Button
from gpiozero.pins.pigpio import PiGPIOFactory

remote = PiGPIOFactory('192.168.0.2')

led = LED(22)
btn = Button(22, pin_factory=remote)

led.source = btn.values
Pi Zero GPIO Expander

A Raspberry Pi has been connected
Type: BCM2708

Please select the role you want it to have:
- GPIO expansion board
- eMMC / SD card reader
- Custom application

☐ Remember selection

[Image: Raspberry Pi connected screen with options]
from gpiozero import LED
from gpiozero.pins.pigpio import PiGPIOFactory

pizero = PiGPIOFactory('fe80::1%usb0')

led = LED(22, pin_factory=pizero)

led.blink()
from somelib import GardenLight, LightSensor, MotionSensor
from gpiozero.tools import all_values, negated

garden = GardenLight()
light = LightSensor()
motion = MotionSensor()

garden.source = all_values(negated(light.values), motion.values)
Z-Wave devices & asyncio
GPIO Zero on GitHub & ReadTheDocs

A simple interface to GPIO devices with Raspberry Pi

https://gpiozero.readthedocs.io/

Created by Ben Nuttall of the Raspberry Pi Foundation, Dave Jones, and other contributors.

About

Component interfaces are provided to allow a frictionless way to get started with physical computing:

```python
from gpiozero import LED

led = LED(17)

while True:
    led.on()
    sleep(0.5)
    led.off()
    sleep(0.5)
```

With very little code, you can quickly get going connecting your components together.
• Python package repository providing **Arm platform wheels** for Raspberry Pi

• **Builds automated** from PyPI releases, plus manual builds e.g. opencv & tensorflow

• Raspbian is pre-configured to use piwheels.org as an additional index to PyPI

• Massively reduces **pip install** time for Raspberry Pi users

• **Natively compiled** on Raspberry Pi 3 hardware (Mythic Beasts Pi cloud)

• Repo hosted on single Raspberry Pi serving **300-400k packages per month**
Raspberry Jam

- Independently organised community events around the world
- Family-friendly
- Mix of meetup / conference / workshop styles
- Makers, hackers, programmers & beginners come together
- Find one near you – or start your own!
- raspberrypi.org/jam
• Free coding clubs for young people
• Find one near you and volunteer as a mentor – or start a new Dojo in your area
• coderdojo.com
Raspberry Pi & Python poster session today!
Programming paradigms for physical computing and IoT

Ben Nuttall
Raspberry Pi Foundation
UK Charity 1129409