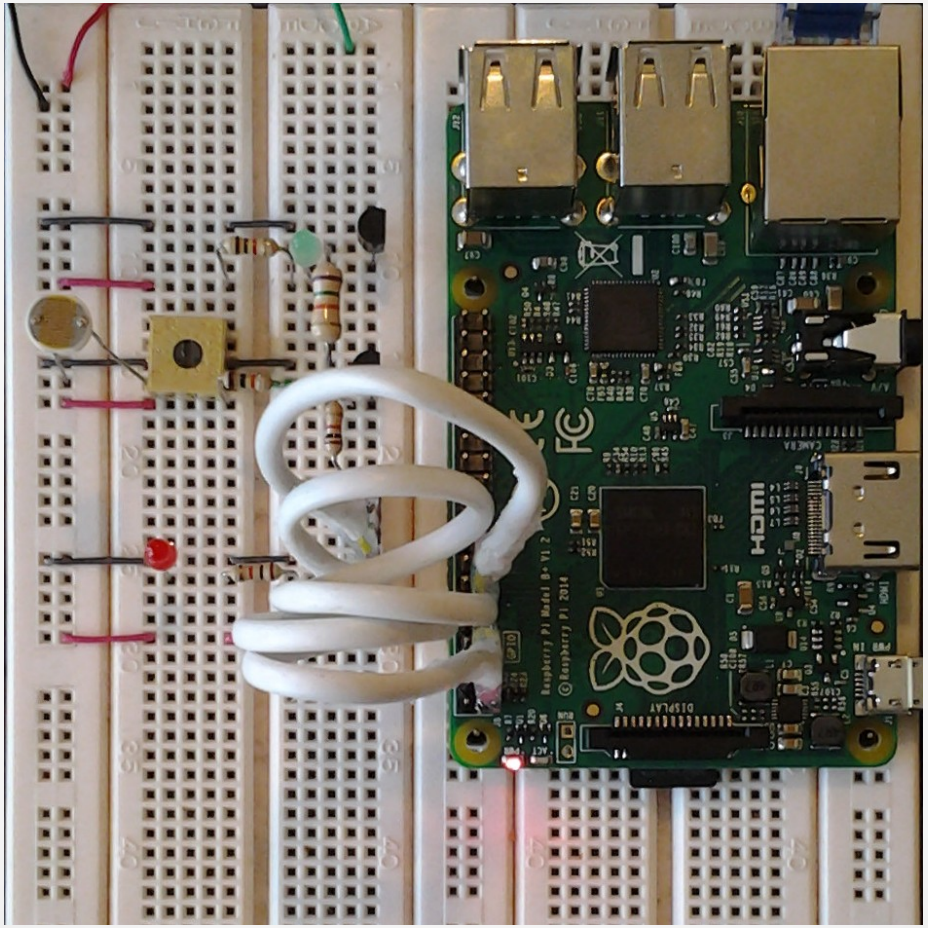
Python en Raspberry Pi

Demo



Indentación

```
for i in range(2):
    print('A')
    print('B')

print('----------')

for i in range(2):
    print('A')
print('B')
```

```
A
B
A
B
----------
A
A
B
```

Archivos

- Archivos de texto plano con extensión `.py`.
- Se ejecutan con el comando:
  `python3 miarchivo.py`

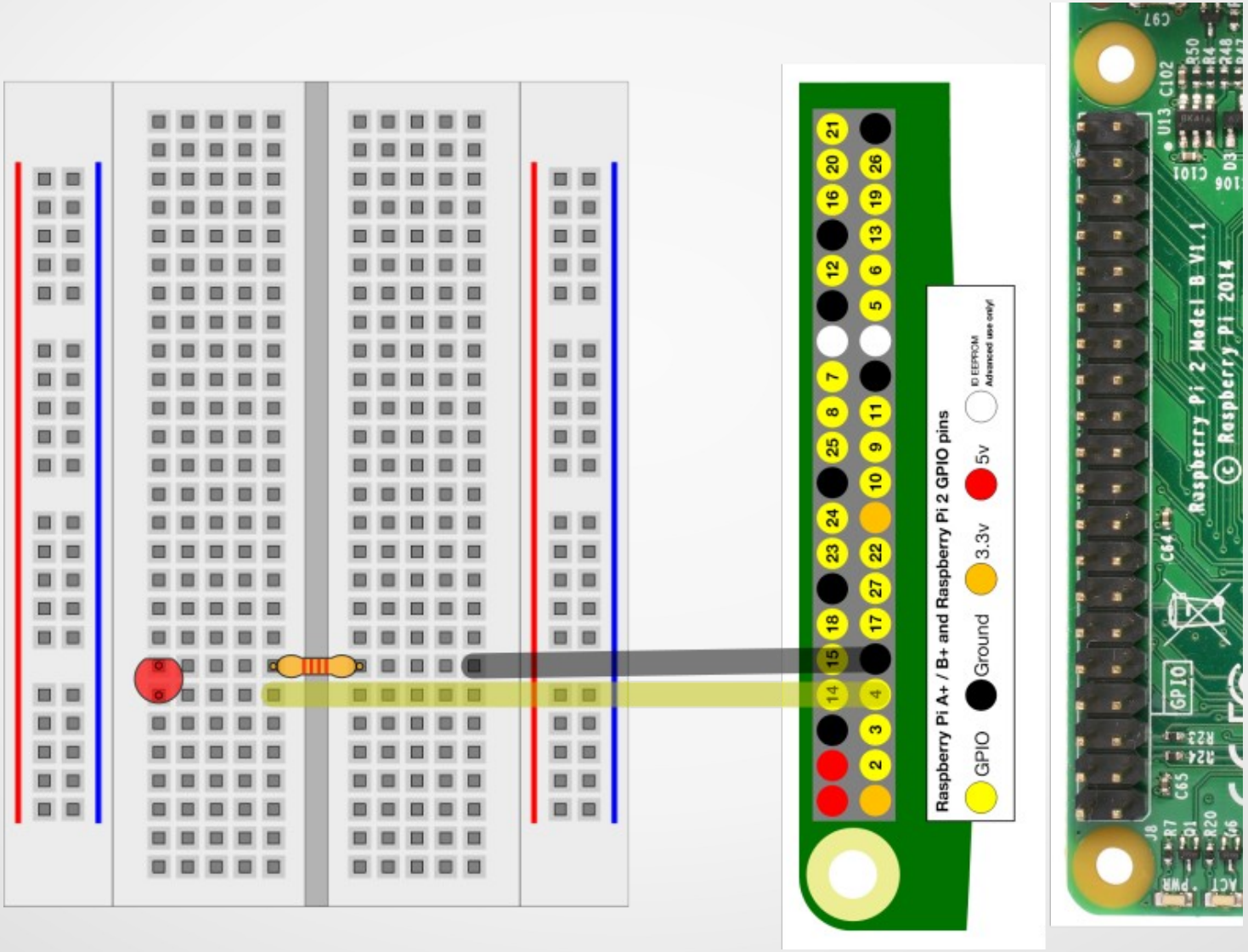
Bibliotecas

- Son programas que sirven como piezas para crear otros programas. (Legos!)
- Biblioteca estándar: datetime, re, os, sqlite3, json, random, …
- Bibliotecas de terceros: Pillow, Scrapy, Matplotlib, PyGame, NumPy, ScyPy, Tornado, Django, PyOpenCV, …

Bibliotecas (continuación)

- Se instalan utilizando pip (es como Google Play para Python).
- Para instalar pip:
```
sudo apt-get install python3-pip
```
- Para instalar una librería (tornado y Rpi.GPIO):
```
sudo pip-3.2 install tornado
sudo pip-3.2 install Rpi.GPIO
```
- Para usar una librería en python:
```
from time import sleep
sleep(3)
```

Armar el circuito de prueba



GPIO (Entradas y salidas de propósito general)

```
from RPi import GPIO

GPIO.setmode(GPIO.BCM)
GPIO.setup(4, GPIO.OUT)
GPIO.setup(22, GPIO.IN)

print(GPIO.input(22))
GPIO.output(4, 1)

GPIO.cleanup()
```

If

```
from RPi import GPIO

GPIO.setmode(GPIO.BCM)
GPIO.setup(4, GPIO.OUT)
GPIO.setup(22, GPIO.IN)

entrada = GPIO.input(22)

if entrada:
    GPIO.output(4, 1)
else:
    GPIO.output(4, 0)

GPIO.cleanup()
```

For

```
for i in range(5):
    print('Hola!')

for i in range(5):
    print(i)
```

For + If + Sleep

```
GPIO.setmode(GPIO.BCM)
GPIO.setup(4, GPIO.OUT)

encendido = False
for i in range(10):
    if encendido:
        GPIO.output(4, 1)
        encendido = False
    else:
        GPIO.output(4, 0)
        encendido = True
    sleep(1)

GPIO.cleanup()
```

Funciones

```
from time import sleep
from RPi import GPIO


def blink(veces, tiempo):
    cambios = 2 * veces
    encendido = False
    GPIO.output(4, 0)

    for i in range(cambios):
        if encendido:
            GPIO.output(4, 0)
            encendido = False
        else:
            GPIO.output(4, 1)
            encendido = True
        sleep(tiempo / cambios)

GPIO.setmode(GPIO.BCM)
GPIO.setup(4, GPIO.OUT)

blink(10, 5)
blink(20, 5)
blink(40, 5)

GPIO.cleanup()
```

Try

```
from time import sleep
from RPi import GPIO

def blink(veces, tiempo):
    encendido = False
    GPIO.output(4, 0)

    for i in range(2*veces):
        if encendido:
            GPIO.output(4, 0)
            encendido = False
        else:
            GPIO.output(4, 1)
            encendido = True
        sleep(tiempo)

try:
    GPIO.setmode(GPIO.BCM)
    GPIO.setup(4, GPIO.OUT)

    blink(100, 0.05)
    blink(50, 0.1)
    blink(5, 1)
    blink(10, 0.5)

except KeyboardInterrupt:
    exit()

finally:
    GPIO.cleanup()
```

Objetos

- Son agrupaciones de variables y funciones.
- Se tratan de parecer a un objeto real.
- En Python TODO es un objeto.

```
'Hola'.upper() → 'HOLA'
'Hola'.lower() → 'hola'
```

Objetos (ejemplo)

```
class Perro(object):
    hambre = 2

    def comer(self):
        if self.hambre > 0:
            print('Nom nom nom nom!')
            self.hambre = self.hambre - 1
        else:
            print('No gracias, ya estoy satisfecho!')

    def caminar(self):
        print(':D')
        self.hambre = self.hambre + 1


toby = Perro()
pluto = Perro()

toby.caminar()
pluto.comer()
pluto.comer()
pluto.comer()
toby.comer()
```

Tornado

```
from tornado.web import RequestHandler, Application
from tornado.ioloop import IOLoop


class Handler(RequestHandler):
    def get(self):
        self.write('Hola Mundo HTML!')

try:
    Application([('/$', Handler)]).listen(50000)
    IOLoop.instance().start()

except KeyboardInterrupt:
    exit()
```

Tornado (output)

```
from tornado.web import RequestHandler, Application
from tornado.ioloop import IOLoop
from RPi import GPIO


class Handler(RequestHandler):
    def get(self):
        self.write(
            "<form action='/' method='post'>"
            "    <input type='submit' name='encender' value='Encender'>"
            "    <input type='submit' name='apagar' value='Apagar'>"
            "</form>"
        )

    def post(self):
        encender = self.get_argument('encender', '')
        apagar = self.get_argument('apagar', '')

        if encender and not apagar:
            GPIO.output(4, 1)

        if not encender and apagar:
            GPIO.output(4, 0)

        self.get()

try:
    GPIO.setmode(GPIO.BCM)
    GPIO.setup(4, GPIO.OUT)

    Application([('/$', Handler)]).listen(50000)
    IOLoop.instance().start()

except KeyboardInterrupt:
    exit()

finally:
    GPIO.cleanup()
```

Tornado (input)

```
from tornado.web import RequestHandler, Application
from tornado.ioloop import IOLoop, PeriodicCallback
from RPi import GPIO

last_test = 0


def test_night():
    global last_test

    night = GPIO.input(22)

    if night and not last_test:
        GPIO.output(4, 1)

    if not night and last_test:
        GPIO.output(4, 0)

    last_test = night


class Handler(RequestHandler):
    def get(self):
        self.write(
            "<form action='/' method='post'>"
            "    <input type='submit' name='encender' value='Encender'>"
            "    <input type='submit' name='apagar' value='Apagar'>"
            "</form>"
        )

    def post(self):
        encender = self.get_argument('encender', '')
        apagar = self.get_argument('apagar', '')

        if encender and not apagar:
            GPIO.output(4, 1)

        if not encender and apagar:
            GPIO.output(4, 0)

        self.get()
```

```
try:
    GPIO.setmode(GPIO.BCM)
    GPIO.setup(4, GPIO.OUT)
    GPIO.setup(22, GPIO.IN)

    Application([('/$', Handler)]).listen(50000)
    PeriodicCallback(test_night, 100).start()
    IOLoop.instance().start()

except KeyboardInterrupt:
    exit()

finally:
    GPIO.cleanup()
```