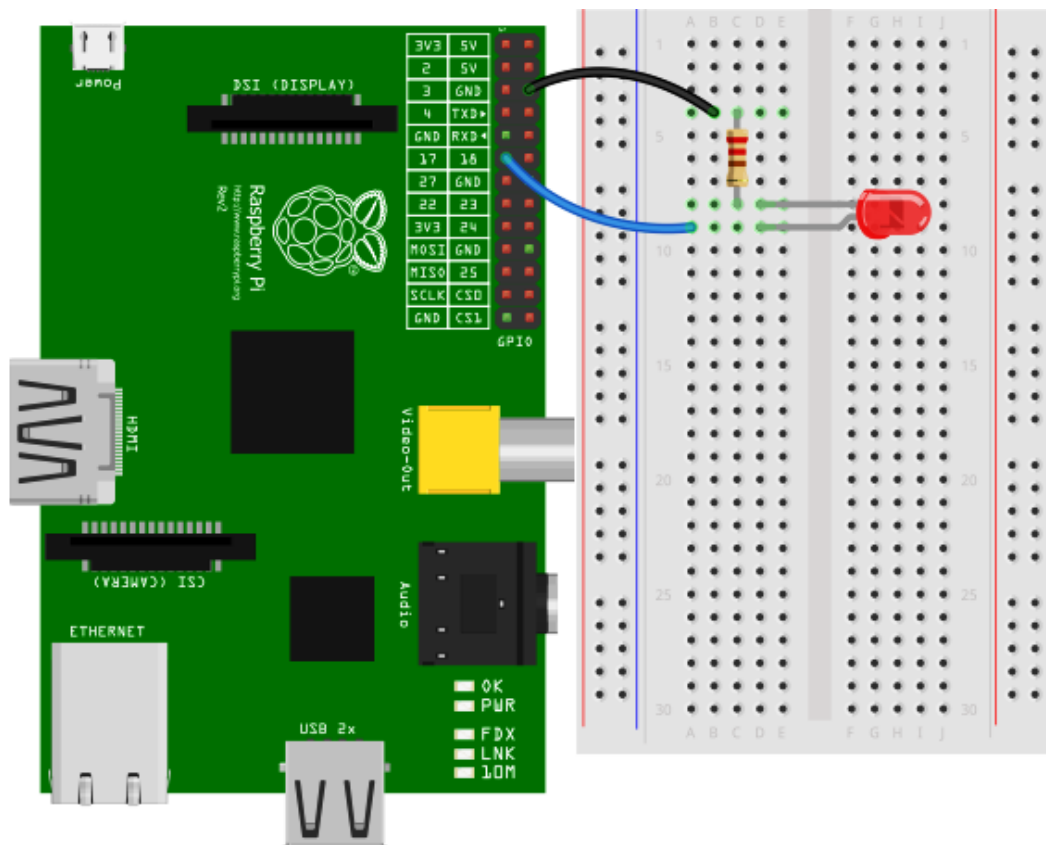




## المثال الأول: تشغيل وإطفاء ليد Blinking Led



### المكونات المطلوبة:

- ✓ لوحة التجارب Bread Board
- ✓ لوحة راسبيري باي
- ✓ دايود ضوئي Led
- ✓ مقاومة ٣٠٠ أوم
- ✓ أسلاك توصيل

### الهدف من المثال

تشغيل الدايود الضوئي و اطفاءه (عمل  
فلاش Flash) إلى ما لا نهاية

### تجهيز أجزاء المشروع:

قم بوضع الدايود الضوئي على لوحة التجارب ووصل الطرف السالب مع المقاومة الـ ٣٠٠ أوم و الطرف الموجب مع المنفذ رقم ١١ على لوحة الراسبيري، ثم وصل طرف المقاومة الآخر بالطرف على السالب على لوحة الراسبيري، المرحلة التالية ستكون كتابة الكود البرمجي الذي سيتحكم في تشغيل و اغلاق الدايود الضوئي.



## الكود البرمجي

```
import time
import RPi.GPIO as GPIO

GPIO.setmode(GPIO.BOARD)
GPIO.setup(11, GPIO.OUT)

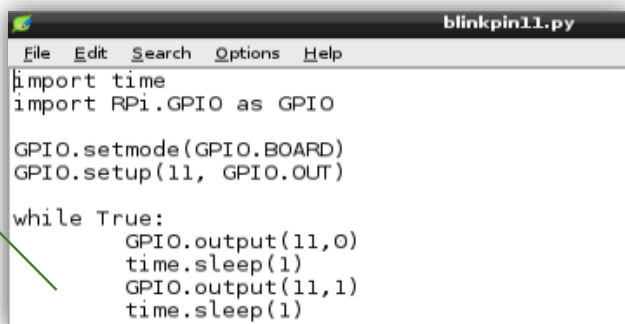
while True:
    GPIO.output(11,0)
    time.sleep(1)
    GPIO.output(11,1)
    time.sleep(1)
```

يمكنك كتابة كود البرنامج اما عن طريق محرر النصوص على الواجهة الرسومية Leafpad أو محرر النصوص الذي يعمل من سطر الأوامر nano، كلاهما يصلح لكتابة أي كود برمجي مع العلم ان محرر نانو يحتوي بعض المميزات الخاصة لكتابة الأكواد البرمجية مثل تلوين الكود (هذه الخاصية تسهل قراءة الكود).

## استخدام محرر النصوص LeafPad

افتح برنامج LeafPad من قائمة Accessories ، ثم اكتب النص و احفظ الملف باسم blinkpin11.py داخل المجلد /home/pi

لا تنسى ترك المسافة بعد  
while True عن طريق  
الضغط على زر Tab في  
الجانب الأيسر من لوحة  
المفاتيح



```
File Edit Search Options Help
import time
import RPi.GPIO as GPIO

GPIO.setmode(GPIO.BOARD)
GPIO.setup(11, GPIO.OUT)

while True:
    GPIO.output(11,0)
    time.sleep(1)
    GPIO.output(11,1)
    time.sleep(1)
```

## استخدام محرر النصوص Nano

يعتبر محرر النصوص نانو من أقوى محررات النصوص في بيئة سطر الأوامر داخل أنظمة لينكس لما له من قدرة على التعرف على العديد من لغات البرمجة و القدرة على البحث و التنسيق داخل الملفات لذلك سأستخدم هذا البرنامج دائما في كتابة النصوص البرمجية.



تشغيل نانو بسيط جدا فكل ما عليك فعله هو فتح برنامج سطر الأوامر و كتابة Your-File nano حيث تستبدل Your-File باسم الملف الذي تريد تحريره و اذا لم يكن هذا الملف موجود فسيقوم برنامج نانو بعمل ملف جديد وتسميته على هذا الأسم، في هذا المثال سأستخدم الأمر

```
nano blinkpin11.py
```

```
pi@raspberrypi ~ $ nano blinkpin11.py
```

سيقوم برنامج نانو بعمل ملف جاهز لاستقبال أوامر بلغة البايثون، والآن كل ما عليك فعله هو كتابة الأكواد البرمجية السابقة وسيظهر الكود المكتوب في محرر النصوص كالتالي:

```
GNU nano 2.2.6      File: blink11.py

import time
import RPi.GPIO as GPIO

GPIO.setmode(GPIO.BOARD)
GPIO.setup(11, GPIO.OUT)

while True:
    GPIO.output(11, 0)
    time.sleep(1)
    GPIO.output(11, 1)
    time.sleep(1)
```

[ Read 12 lines ]

^G Get Help ^O WriteOut ^R Read File ^Y Prev Page ^K Cut Text ^C Cur Pos  
^X Exit ^J Justify ^W Where Is ^V Next Page ^U UnCut Text ^T To Spell

لحفظ الملف أضغط Ctrl+x ثم ستظهر رسالة في الأسفل تسألك اذا ما كنت تريد حفظ البرنامج عندها اضغط زر y ثم اضغط Enter ليتم حفظ الملف.

```
Save modified buffer (ANSWERING "No" WILL DESTROY CHANGES) ?
Y Yes
N No      ^C Cancel
```

## تشغيل البرنامج

قم بتنفيذ الأمر التالي في سطر الأوامر:

```
sudo python blinkpin11.py
```

ولاحظ ما يحدث للدايود الضوئي (يضيئ لمدة ثانية و يطفئ لمدة ثانية).

```
pi@raspberrypi ~ $ sudo python blinkpin11.py
```

لاغلاق البرنامج أضغط على زر Ctrl + C (أغلب برامج لينكس التي تعمل من سطر الأوامر يمكن اغلاقها



بهذه الطريقة)، ثم قم بتشغيل البرنامج مرة ثانية ولاحظ الرسالة الجديدة التي ستظهر على الشاشة، في المرة الأولى التي شغلنا بها البرنامج سيعمل دون أن يظهر شيء على الشاشة و سيبدأ الـ dauid الضوئي Led بالانارة و الانطفاء كل ثانية لكن عند تشغيل البرنامج للمرة الثانية ستظهر رسالة تخبرك بأن "المخرج الذي تريد استخدام الآن قد يكون مُستخدمًا بالفعل"

```
pi@raspberrypi ~ $ sudo nano blink11.py
pi@raspberrypi ~ $ sudo python blink11.py
blink11.py:10: RuntimeWarning: This channel is already in use, continuing anyway
. Use GPIO.setwarnings(False) to disable warnings.
```

هذه الرسالة تظهر عند تشغيل برنامج تلو الآخر على نفس المنفذ (نفس Pin)، يمكنك تجاهل هذه الرسالة وإذا أحببت إخفائها اكتب `GPIO.setwarnings(False)` في ملف برنامج التحكم كالتالي:

```
import time
import RPi.GPIO as GPIO

GPIO.setwarnings(False)
GPIO.setmode(GPIO.BOARD)
GPIO.setup(11, GPIO.OUT)

while True:
    GPIO.output(11, 0)
```

## شرح الكود

أمر استدعاء المكتبات: يمثل هذا الجزء بداية أي برنامج في معظم لغات البرمجة وهو إضافة المكتبة البرمجية `time` المسؤولة عن قياس الزمن و أو تنفيذ أمر معين لفترة محددة من الزمن، المكتبة الثانية هي `RPi.GPIO` وهي مكتبة التحكم في الـ `GPIO` الخاصة بالـ راسبيري باي.

```
import time
import RPi.GPIO as GPIO
```

أمر التفعيل: هذا الأمر يستخدم في تفعيل جميع منافذ الـ `GPIO`

```
GPIO.setmode(GPIO.BOARD)
```

و يجعلها جاهزة لاستقبال أوامر التحكم، كما يرتب المنافذ بناء على مكانها على لوحة الـ راسبيري باي.

```
GPIO.setup(11, GPIO.OUT)
```

اعداد وظيفة الـ `Pin`: هذا الأمر يحدد وظيفة أي `Pin` على

الـ `GPIO` و يحدد هل ستعمل كمخرج `OUT` أم كمدخل `IN` ويمثل رقم ١١ رقم الـ `Pin` التي نريد التحكم بها، مع ملاحظة انه في حالة استخدام اكثر من `Pin` يجب كتابة كيفية تشغيلها `IN` أو `OUT` في بداية البرنامج، على سبيل المثال نريد تشغيل الـ `pin` رقم ١١ كمخرج و الـ `Pin` رقم ٢٣ كمدخل اذا سكتب:

```
GPIO.setup(11, GPIO.OUT)
GPIO.setup(23, GPIO.IN)
```



تكرار الأوامر إلى ما لا نهاية: تستخدم دوال التكرار loops في تنفيذ مجموعة من الأوامر لعدد معين من المرات أو إلا ما لا

while True:

نهاية وعند كتابة الأمر while True: فهذا يعني أن جميع الأوامر التي تكتب بعدها ستنفذ إلا ما لا نهاية أو حتى يتم اغلاق البرنامج أو اغلاق الراسبيري باي نفسها.

while True:

command to do  
→ another thing to do  
another thing to do

لاحظ انه يجب ترك مسافة قبل كل سطر نريد إدخاله داخل عملية التكرار و ذلك عن طريق الضغط على زر Tab في الجانب الأيسر من لوحة المفاتيح

تشغيل و اغلاق المخارج: يستخدم الأمر GPIO.output(pin, status) في تشغيل او اغلاق أي منفذ GPIO حيث نستبدل pin

GPIO.output(11,0)

برقم المخرج المراد تشغيله أو اطفائه و نستبدل status بحالة التشغيل وهي اما = 1 و تعني تشغيل المنفذ (فرق الجهد = 3,3 فولت) و اما = 0 صفر و تعني اغلاق المنفذ (فرق الجهد = صفر).

Time.sleep(1)

التحكم في زمن التشغيل و الإغلاق: يستخدم الأمر

time.sleep(time) في تحديد زمن تنفيذ الأمر الذي يسبقه، فمثلاً إذا كان الأمر الذي يسبقه يشغل المنفذ رقم 11 و كتبنا time.sleep(5) فهذا يعني أن المنفذ رقم 11 سيظل يعمل لمدة 5 ثواني.

GPIO.output(11,0)

◆ أغلق المخرج رقم 11 (فرق الجهد = صفر)

time.sleep(1)

◆ انتظر لمدة ثانية

GPIO.output(11,1)

◆ شغل المخرج رقم 11 (فرق الجهد = 3,3 فولت)

time.sleep(1)

◆ انتظر لمدة ثانية

يمكن كتابة الأمر GPIO.output(pin,status) على صورة True أو False بحيث تمثل كلمة True تشغيل المخرج (بدلاً من 1) و تمثل كلمة False اغلاق المخرج (بدلاً من 0)، على سبيل المثال يمكننا تعديل البرنامج ليصبح كالتالي:

```
import time
import RPi.GPIO as GPIO
GPIO.setmode(GPIO.BOARD)
GPIO.setup(11, GPIO.OUT)
```

while True:

GPIO.output(11, False)

time.sleep(1)

GPIO.output(11, True)

time.sleep(1)





## تطوير المثال الأول

سنقوم بتطوير المثال الأول لكي يعرض رسالة على الشاشة تخبرنا بأن الليد يعمل الآن أو الليد مغلق، لعمل هذا التعديل سنضيف الأمر print مع الرسالة التي نريد عرضها ليصبح الكود كالتالي:

```
import time
import RPi.GPIO as GPIO
GPIO.setmode(GPIO.BOARD)
GPIO.setup(11, GPIO.OUT)

while True:
    GPIO.output(11, False)
    print "Led on Pin 11 is now OFF (output = zero volt)"
    time.sleep(1)

    GPIO.output(11, True)
    print "Led on Pin 11 is now ON (output = 3.3 volt)"
    time.sleep(1)
```

صورة الكود بعد التعديل على برنامج نانو

```
GNU nano 2.2.6      File: blink11.py      Modified
import time
import RPi.GPIO as GPIO

GPIO.setwarnings(False)
GPIO.setmode(GPIO.BOARD)
GPIO.setup(11, GPIO.OUT)

while True:
    GPIO.output(11, False)
    print "Led on Pin 11 is now OFF (output = zero volt)"
    time.sleep(1)

    print "Led on Pin 11 is now ON (output = 3.3 volt)"
    GPIO.output(11, True)
    time.sleep(1)
```

والآن أعد تشغيل البرنامج ولاحظ ما سيظهر على الشاشة بالتزامن مع تشغيل و اغلاق الدايد الضوئي.

```
pi@raspberrypi ~ $ sudo python blink11.py
Led on Pin 11 is now OFF (output = zero volt)
Led on Pin 11 is now ON (output = 3.3 volt)
Led on Pin 11 is now OFF (output = zero volt)
Led on Pin 11 is now ON (output = 3.3 volt)
```