

```
1 import RPi.GPIO as GPIO # GPIOを利用する
2 import time # sleepを利用する
3
4 # ポート番号の定義
5 SWITCH = 18
6 LED = 21
7 led_value = GPIO.LOW
8
9 # GPIOの初期化 --- (*1)
10 GPIO.setmode(GPIO.BCM)
11 GPIO.setup(SWITCH, GPIO.IN, pull_up_down=GPIO.PUD_DOWN)
12 GPIO.setup(LED, GPIO.OUT)
13
14 # スイッチの切り替えイベントの定義 --- (*2)
15 def callback_change_switch(ch):
16     global led_value
17     print("callback", ch)
18     if ch != SWITCH: return
19     #違うボタンが押されたときに何もしないためのif文
20     if led_value == GPIO.LOW:
21         GPIO.output(LED, )#①この行のプログラムを完成させる
22         led_value = GPIO.HIGH
23     else:
24         GPIO.output(LED, )#②この行のプログラムを完成させる
25         led_value = GPIO.LOW
26
27 # イベントの設定 --- (*3)
28 GPIO.add_event_detect(
29     SWITCH, # ポート番号
30     GPIO.RISING, # イベントの種類
31     callback=callback_change_switch, # 関数の指定
32     bouncetime=200) # 連続イベントを制限
33
34 # LEDを消灯する
35 GPIO.output(LED, GPIO.LOW)
36
37 # プログラムが終わらないように待機 --- (*4)
38 try:
39     while True:
40         time.sleep(0.1)
41
42 except KeyboardInterrupt:
43     GPIO.cleanup()
44
```

```
1 import RPi.GPIO as GPIO # GPIOを利用する
2 import time # sleepを利用する
3
4 # ポート番号の定義
5 SWITCH = 18
6 LED = 21
7 led_value = GPIO.LOW
8
9 # GPIOの初期化 --- (*1)
10 GPIO.setmode(GPIO.BCM)
11 GPIO.setup(SWITCH, GPIO.IN, pull_up_down=GPIO.PUD_DOWN)
12 GPIO.setup(LED, GPIO.OUT)
13
14 # スイッチの切り替えイベントの定義 --- (*2)
15 def callback_change_switch(ch):
16     global led_value
17     print("callback", ch)
18     if ch != SWITCH: return
19     if led_value == GPIO.LOW:
20         led_value = GPIO.HIGH
21     else:
22         led_value = GPIO.LOW
23
24 # イベントの設定 --- (*3)
25 GPIO.add_event_detect(
26     SWITCH, # ポート番号
27     GPIO.RISING, # イベントの種類
28     callback=callback_change_switch, # 関数の指定
29     bouncetime=200) # 連続イベントを制限
30
31 # LEDを消灯する
32 GPIO.output(LED, GPIO.LOW)
33
34 # プログラムが終わらないように待機 --- (*4)
35 try:
36     while True:
37         if led_value == GPIO.HIGH:
38
39             else:
40
41             time.sleep(0.1)
42
43 except KeyboardInterrupt:
44     GPIO.cleanup()
45
46
```

```
1 import RPi.GPIO as GPIO # GPIOを利用する
2 import time # sleepを利用する
3
4 # ポート番号の定義
5 SWITCH = 18
6 LED = 21
7 led_value = GPIO.LOW
8 LED_BLINKING = 1
9 LED_LIGHTING = 2#モードの定義
10
11 # GPIOの初期化 --- (*1)
12 GPIO.setmode(GPIO.BCM)
13 GPIO.setup(SWITCH, GPIO.IN, pull_up_down=GPIO.PUD_DOWN)
14 GPIO.setup(LED, GPIO.OUT)
15
16
17 #コールバック関数
18 def callback_change_switch(ch):
19     global led_value
20     print("callback", ch)
21     if ch != SWITCH: return
22     if led_mode == LED_BLINKING:
23         led_mode = LED_LIGHTING
24     else:
25         led_mode = LED_BLINKING
26
27
28 # イベントの設定 --- (*3)
29 GPIO.add_event_detect(
30     SWITCH, # ポート番号
31     GPIO.RISING, # イベントの種類
32     callback=callback_change_switch, # 関数の指定
33     bouncetime=200) # 連続イベントを制限
34
35 # LEDを消灯する
36 GPIO.output(LED, GPIO.LOW)
37
38 # プログラムが終わらないように待機 --- (*4)
39 try:
40     while True:
41         if mode ==:
42             GPIO.output(LED, GPIO.HIGH)
43             time.sleep(0.5)
44             GPIO.output(LED, GPIO.LOW)
45             time.sleep(0.5)
46     else:
47         GPIO.output(LED, GPIO.HIGH)
48         time.sleep(0.5)
49
50 except KeyboardInterrupt:
51     GPIO.cleanup()
52
```

```
1 import RPi.GPIO as GPIO # GPIOを利用する
2 import time # sleepを利用する
3
4 # ポート番号の定義
5 SWITCH = 18
6 LED = 21
7 led_value = GPIO.LOW
8 LED_BLINKING = 1#モードの定義
9 LED_LIGHTING = 2#モードの定義
10
11 # GPIOの初期化 --- (*1)
12 GPIO.setmode(GPIO.BCM)
13 GPIO.setup(SWITCH, GPIO.IN, pull_up_down=GPIO.PUD_DOWN)
14 GPIO.setup(LED, GPIO.OUT)
15
16 # LEDを消灯する
17 GPIO.output(LED, GPIO.LOW)
18
19 # プログラムが終わらないように待機 --- (*4)
20 try:
21     while True:
22         if GPIO.input(SWITCH):#ポーリング
23             mode = LED_BLINKING#点滅
24         else:
25             mode = LED_LIGHTING#点灯
26
27         if mode == LED_BLINKING:
28             GPIO.output(LED, GPIO.HIGH)
29             time.sleep(0.5)
30             GPIO.output(LED, GPIO.LOW)
31             time.sleep(0.5)
32         else:
33             GPIO.output(LED, GPIO.HIGH)
34             time.sleep(0.5)
35
36 except KeyboardInterrupt:
37     GPIO.cleanup()
38
39
```