

## Mãos na Massa: Criando o script para controlar os motores

### Criando a pasta dos scripts

O primeiro passo é criarmos uma pasta na *home* de nosso Raspberry Pi, que é onde ficará salvo os nossos scripts:

```
mkdir ~/pibot
cd ~/pibot
```

Dentro dessa pasta, crie o arquivo **controle.py** com um editor de texto de sua preferência, como o **nano** ou **vi**. Não esqueça que você também pode fazer todo esse processo através da interface gráfica do Raspberry Pi, conectando-se a ele utilizando o VNC, como vimos nos cursos anteriores!

### Importando as bibliotecas

Dentro do **controle.py**, vamos começar importando as bibliotecas que usaremos:

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

Como temos que instalar o módulo `getch`, que não vem por padrão no Raspberry Pi, vamos executar o seguinte comando no terminal:

```
sudo pip3 install getch
```

Com o módulo `getch` instalado, podemos continuar com a criação do **controle.py**.

### Configurando os GPIO

Agora, vamos configurar as portas do GPIO através da função `GPIO.setmode()`, para dizer que queremos utilizar como referência a pinagem física do GPIO:

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

GPIO.setmode(GPIO.BOARD)
```

Além disso, vamos desabilitar as mensagens de alerta com a função `GPIO.setwarnings()`, passando o valor `False` como parâmetro:

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

```
import getch

GPIO.setmode(GPIO.BOARD)
GPIO.setwarnings(False)
```

## Mapeando os pinos

Com essas configurações iniciais prontas, podemos iniciar o mapeamento dos pinos físicos. Vamos criar quatro variáveis e colocar em cada uma delas o valor do seu pino físico do GPIO:

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

GPIO.setmode(GPIO.BOARD)
GPIO.setwarnings(False)

F_DIREITA = 16
F_ESQUERDA = 11
T_DIREITA = 18
T_ESQUERDA = 13
```

Com os pinos configurados, podemos criar uma função de *setup*, que será executada no início de nosso código e configurará os pinos como saída, ou seja como **OUTPUT** :

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

GPIO.setmode(GPIO.BOARD)
GPIO.setwarnings(False)

F_DIREITA = 16
F_ESQUERDA = 11
T_DIREITA = 18
T_ESQUERDA = 13

def setup_motor():
    GPIO.setup(F_DIREITA, GPIO.OUT)
    GPIO.setup(F_ESQUERDA, GPIO.OUT)
    GPIO.setup(T_DIREITA, GPIO.OUT)
    GPIO.setup(T_ESQUERDA, GPIO.OUT)
```

## Colocando os motores para rodar

Com todo este *setup* inicial pronto, falta agora criarmos as **quatro funções** responsáveis por controlar o movimento do robô. As funções são `move_frente` , `move_tras` , `move_direita` e `move_esquerda` .

Vamos começar com a `move_frente` e `move_tras` :

```
def move_frente():
    GPIO.output(F_DIREITA, GPIO.HIGH)
    GPIO.output(F_ESQUERDA, GPIO.HIGH)
    time.sleep(0.4)
    GPIO.output(F_DIREITA, GPIO.LOW)
    GPIO.output(F_ESQUERDA, GPIO.LOW)

def move_tras():
    GPIO.output(T_DIREITA, GPIO.HIGH)
    GPIO.output(T_ESQUERDA, GPIO.HIGH)
    time.sleep(0.4)
    GPIO.output(T_DIREITA, GPIO.LOW)
    GPIO.output(T_ESQUERDA, GPIO.LOW)
```

Aqui estamos setando os motores para **HIGH**, assim ambos girarão para frente (F) ou para trás (T), por 0,4 segundos e em seguida voltam para **LOW**, parando de girar. Com essa lógica, a cada chamada de função os motores girarão um pouco e moverão o robô na direção desejada.

Vamos agora criar as funções `move_esquerda` e `move_direita`:

```
def move_direita():
    GPIO.output(F_ESQUERDA, GPIO.HIGH)
    time.sleep(0.2)
    GPIO.output(F_ESQUERDA, GPIO.LOW)

def move_esquerda():
    GPIO.output(F_DIREITA, GPIO.HIGH)
    time.sleep(0.2)
    GPIO.output(F_DIREITA, GPIO.LOW)
```

Crie também a função `move_parar`, que será usada em caso de emergência, ela para todos os motores, colocando todos eles em **LOW**:

```
def move_parar():
    GPIO.output(T_DIREITA, GPIO.LOW)
    GPIO.output(T_ESQUERDA, GPIO.LOW)
    GPIO.output(F_DIREITA, GPIO.LOW)
    GPIO.output(F_ESQUERDA, GPIO.LOW)
```

Repare que aqui acionamos os motores inversos da direção que queremos mexer, pois desejamos que o robô gire, e em seguida se mova.

## Capturando as teclas do usuário

Agora que já temos todas as funções necessárias para movimentar o carrinho, vamos começar a escutar os *inputs* do usuário que irão acionar essas funções. Vamos utilizar as teclas W, A, S e D para movimentar o carrinho para frente, esquerda, trás e direita, respectivamente.

Para capturar as teclas do usuário, vamos utilizar a função `getch` e criar a função `le_tecla()`:

```
def le_tecla():  
    while True:  
        tecla_comando = getch.getch()  
        if tecla_comando == 'w':  
            move_frente()  
        if tecla_comando == 'd':  
            move_direita()  
        if tecla_comando == 'a':  
            move_esquerda()  
        if tecla_comando == 's':  
            move_tras()  
        if tecla_comando == 'q':  
            move_parar()
```

Repare que nessa função estamos entrando em um loop infinito, pois queremos o script fique escutando as teclas do usuário até o mesmo ser cancelado.

Agora precisamos apenas chamar ambas funções ao final do script:

```
setup_motor()  
le_tecla()
```

## Executando o script

Com tudo pronto, basta executar o comando no terminal:

```
python3 ~/piobot/controle.py
```

Fique atento para pegar possíveis erros de sintaxe! Uma vez rodando, repare que o terminal irá travar. Isso é esperado, pois ele fica aguardando o envio de *inputs* dentro do laço infinito. Faça os testes e veja se os motores estão sendo ativados corretamente.

Você pode parar a execução apertando **CTRL + C**.

## Código completo

Você pode baixar o código completo do script **controle.py** [aqui \(https://s3.amazonaws.com/caelum-online-public/raspberry3/files/cap3/controle.py\)](https://s3.amazonaws.com/caelum-online-public/raspberry3/files/cap3/controle.py).

