

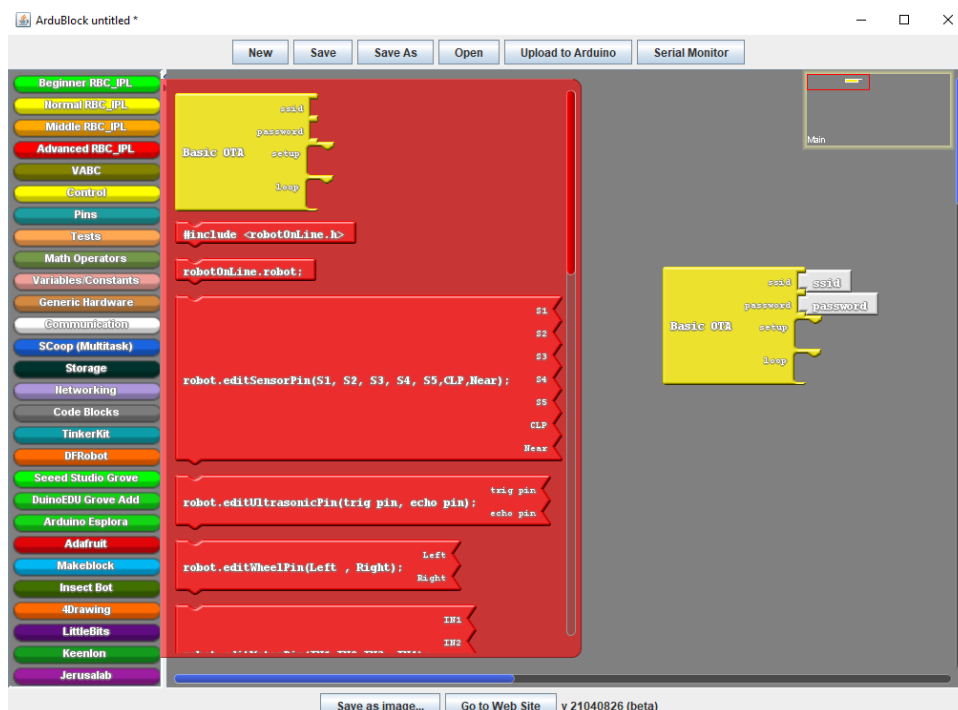


**INSTITUTO DE ENGENHARIA DE
SISTEMAS E COMPUTADORES
(INESC) DE COIMBRA**



Tutorial – Nível Avançado

Programar por blocos



Alunos:

Abel Teixeira - 2180522

Samuel Lourenço - 2180356

Docente: Luís Conde

Carlos Neves

Ano letivo: 2019/2020

Índice

Nível Avançado (Adanced)	4
1. Blocos do Setup	4
1.1. Bloco “#include < iModBot.h>”	4
1.2. Bloco “robot.ediSensorPin(S1, S2, S3, S4, S5, CLP, Near);”	4
1.3. Bloco “robot.editUltrasonicPin(trig pin, echo pin);”	4
1.4. Bloco “robot.editWheelPin(Left , Right);”	5
1.5. Bloco “robot.editMotorPin(INT1, INT2, INT3, INT4);”	5
1.6. Bloco “robot.begin();”	5
2. Blocos do loop	5
2.1. Blocos “robot.getLeftEncoderCount();” e “robot.getRightEncoderCount();”	5
2.2. Bloco “robot.clearEncoderCount();”	5
2.3. Bloco “robot.distance();”	6
2.4. Blocos “robot.forward();” e “robot.reverse();”	6
2.5. Bloco “robot.leftWheel();” e “robot.rightWheel();”	6
2.6. Blocos “robot.rotateLeft();” e “robot.rotateRight();”	6
2.9. Bloco “robot.setSpeeds(Fast, Average, Slow)”	7
2.10. Bloco “robot.steerLeft(byte);” e “robot.steerRight(byte);”	7
2.11. Blocos da condução autónoma	7
2.11.1. Bloco “robot.autoDrive(byte);”	7
2.11.2. Bloco “robot.readCLP();”	8
2.11.3. Bloco “robot.readNear();”	8
2.11.4. Bloco “robot.readS1();”	8
2.11.5. Bloco “robot.readS2();”	8
2.11.6. Bloco “robot.readS3();”	9
2.11.7. Bloco “robot.readS4();”	9
2.11.9. Bloco “robot.readS5();”	9
2.11.10. Bloco “robot.disableCLP();”	9
2.11.11. Bloco “robot.disableNear();”	9
2.11.12. Bloco “robot.disableUltrasonic();”	10
2.11.13. Bloco “robot.endAutoDrive();”	10
2.11.14. Bloco “robot.noLineDelay(uint);”	10
2.12. Blocos “robot.forwardRPS(float);”, “robot.reverseRPS(float);”, “robot.rotateRightRPS(float);” e “robot. rotateLeftRPS(float);”	10
2.13. Blocos “robot.forwardRPM(float);”, “robot.reverseRPM(float);”, “robot.rotateRightRPM(float);” e “robot. rotateLeftRPM(float);”	11
2.14. Blocos “robot.forwardMS(float);”, “robot.reverseMS(float);”, “robot.rotateRightMS(float);” e “robot. rotateLeftMS(float);”	11
2.15. Bloco “robot.enSpeedAdj();”,	11
2.16. Blocos “robot.getRightRPS();” e “robot. gerLeftRPS();”	12
2.17. Blocos “robot.getRightRPM();” e “robot. getLeftRPM();”	12
2.18. Blocos “robot.getRightMS();” e “robot. getLeftMS();”	12

2.19. Funções também muito utilizadas na programação	12
---	-----------

Nível Avançado (Adanced)

No nível avançado já é como se estivesse a programar. Os blocos vermelhos não necessitam de biblioteca porque são funções que só usam na linguagem C++ e os blocos roxos são da biblioteca robot. Esta biblioteca facilitará a programação do robô.

1. Blocos do Setup

1.1. Bloco “#include <iModBot.h>”

Este bloco chama a biblioteca iModBot.h que é necessário para que os outros blocos vermelhos funcionem e também define o nome “robot” para os vários blocos vermelhos que iniciam com o nome “robot.”



```
#include <iModBot.h>
```

1.2. Bloco “robot.editSensorPin(S1, S2, S3, S4, S5, CLP, Near);”

Este bloco define os pinos da placa IR 74HC14. Os pinos S1, S2, S3, S4 e S5 são os pinos dos sensores de infravermelhos, o pino CLP corresponde ao fim-de-curso da placa e o Near corresponde ao sensor infravermelho que está à frente da placa.



1.3. Bloco “robot.editUltrasonicPin(trig pin, echo pin);”

Este bloco define os pinos que estão ligados ao sensor ultrassom (HC-SR04).



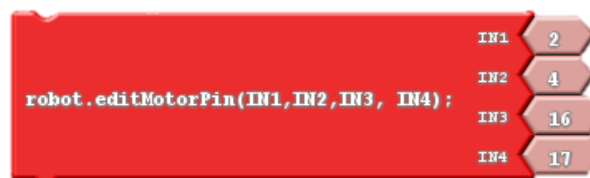
1.4. Bloco “robot.editWheelPin(Left , Right);”

Este bloco define os pinos que estão ligados aos encodares. Left ao encoder da roda esquerda e Right ao encoder da roda direita.



1.5. Bloco “robot.editMotorPin(INT1, INT2, INT3, INT4);”

Este bloco define os pinos que estão ligados no modulo (L293D) que controla os motores que estão acoplados às rodas do robô.



1.6. Bloco “robot.begin();”

Este bloco é necessário para iniciar a biblioteca e as configurações dos pinos.



2. Blocos do loop

2.1. Blocos “robot.getLeftEncoderCount();” e “robot.getRightEncoderCount();”

O bloco “robot.getLeftEncoderCount();” corresponde ao encoder do lado esquerdo e o “robot.getRightEncoderCount();” corresponde ao encoder do lado direito. Guardando o valor na variável com o “Nome” que colocar no bloco branco.



2.2. Bloco “robot.clearEncoderCount();”

Este bloco serve para colocar os valores guardados pelos encoders das duas rodas a zero.



2.3. Bloco “robot.distance();”

Este bloco verifica a distância e guarda-a na variável com o nome que colocar no bloco branco.



2.4. Blocos “robot.forward();” e “robot.reverse();”

O bloco “robot.forward();” faz com que o robô ande para frente e o bloco “robot.reverse();” faz com que o robô ande para trás. Há frente de cada bloco tem de escolher a velocidade que vai andar de 0 a 255.



2.5. Bloco “robot.leftWheel();” e “robot.rightWheel();”

Estes blocos controlam as rodas. O bloco “robot.leftWheel();” controla a roda esquerda e o bloco “robot.rightWheel();” controla a roda direita. À frente de cada bloco coloca-se a velocidade de -255 a 255, sendo que se for um número negativo a roda respetiva recua e se for positiva ela avança.



2.6. Blocos “robot.rotateLeft();” e “robot.rotateRight();”

O bloco “robot.rotateLeft();” faz rodar para a esquerda com a velocidade determinada pelo Duty-Cycle (Modelação de Largura de Pulso) inserido. O bloco “robot.rotateRight();” faz rodar para a direita com a velocidade determinada pelo Duty-Cycle inserido.



2.7. Blocos “robot.turnLeft(degrees);” e “robot.turnRight(degrees);”

O bloco “robot.turnLeft();” faz rodar o robô para a esquerda consoante os graus inseridos. O bloco “robot.turnRight();” faz rodar o robô para a direita consoante os graus inseridos.



2.8. Bloco “robot.stopMotors();”

Este bloco faz com que o robô pare ambos os motores.



2.9. Bloco “robot.setSpeeds(Fast, Average, Slow)”

Este bloco serve para definir a velocidade máxima (255), média e mínima(1) que o robô vai andar.



2.10. Bloco “robot.steerLeft(byte);” e “robot.steerRight(byte);”

O bloco “robot.steerLeft();” faz com que o robô vira para a esquerda e o bloco “robot.steerRight();” faz com que o robô vira para a direita. À frente de cada bloco tem de escolher a velocidade que vai andar de 0 a 255.



2.11. Blocos da condução autónoma

2.11.1. Bloco “robot.autoDrive(byte);”

Este bloco serve para iniciar a condução automática, sem ele os blocos deste documento que começam por 2.11.x não funcionam. Ele pode enviar e receber valores.

Os valores devolvidos para a variável “Nome” têm os seguintes significados:

- 0 - Nada a reportar;
- 1 - Múltiplas linhas encontradas;
- 2 - Obstáculo encontrado;
- 3 - Não foram encontradas linhas.

Os valores que colocar à frente do “byte_sent” que podem ser enviados são:

- 1 - Rodar para a direita;
- 2 - Rodar para a esquerda;

- 3 - Seguir em frente;
- 4 - Retroceder.



2.11.2. Bloco “robot.readCLP();”

Este bloco faz a leitura do fim de curso CLP da placa IR 74HC14, devolvendo 0 ou 1 para a variável “Nome”.



2.11.3. Bloco “robot.readNear();”

Este bloco faz a leitura do infravermelho que se localiza à frente do robô da placa IR 74HC14, devolvendo 0 ou 1 para a variável “Nome”.



2.11.4. Bloco “robot.readS1();”

Este bloco faz a leitura do sensor mais à direita da placa IR 74HC14, devolvendo 0 ou 1 para a variável “Nome”.



2.11.5. Bloco “robot.readS2();”

Este bloco faz a leitura do sensor à direita da placa IR 74HC14, devolvendo 0 ou 1 para a variável “Nome”.



2.11.6. Bloco “robot.readS3();”

Este bloco faz a leitura do sensor a meio da placa IR 74HC14, devolvendo 0 ou 1 para a variável “Nome”.



2.11.7. Bloco “robot.readS4();”

Este bloco faz a leitura do sensor à esquerda da placa IR 74HC14. Devolvendo 0 ou 1 para a variável “Nome”.



2.11.9. Bloco “robot.readS5();”

Este bloco faz a leitura do sensor mais à esquerda da placa IR 74HC14. Devolvendo 0 ou 1 para a variável “Nome”.



2.11.10. Bloco “robot.disableCLP();”

Este bloco desabilita a leitura do sensor CLP da placa IR 74HC14 caso esteja a usar a função de condução automática proporcionada pela biblioteca.



2.11.11. Bloco “robot.disableNear();”

Este bloco desabilita a leitura do sensor Near da placa IR 74HC14 caso esteja a usar a função de condução automática proporcionada pela biblioteca.



2.11.12. Bloco “robot.disableUltrasonic();”

Este bloco desabilita a leitura do sensor ultrassons HC-SR04 caso esteja a usar a função de condução automática proporcionada pela biblioteca.

A red Scratch block with the text "robot.disableUltrasonic();" in white.

2.11.13. Bloco “robot.endAutoDrive();”

Este bloco destinado à funcionalidade da condução autónoma e necessita de ser chamada uma vez para desassociar as interrupções previamente configuradas.

A red Scratch block with the text "robot.endAutoDrive();" in white.

2.11.14. Bloco “robot.noLineDelay(uint);”

Este bloco serve para especificar o tempo (em milissegundos) que o robô aguarda para parar os motores após ter identificado que nenhum dos sensores detetou uma linha.

A red Scratch block with the text "robot.noLineDelay(uint); unit 0" in white.

2.12. Blocos “robot.forwardRPS(float);”, “robot.reverseRPS(float);”, “robot.rotateRightRPS(float);” e “robot.rotateLeftRPS(float);”

Estes blocos vão colocar o robô à velocidade de rotações por segundo que coloca à frente do bloco pretendido. O bloco “robot.forwardRPS(float);” faz com que as duas rodas andem para a frente à velocidade que colocar e o bloco “robot.reverseRPS(float);” faz a mesma coisa só que anda para trás. O bloco “robot.rotateRightRPS(float);” escolhe a velocidade da roda direita e o bloco “robot.rotateLeftRPS(float);” da roda esquerda.

A red Scratch block with the text "robot.forwardRPS(float); float 0" in white.A red Scratch block with the text "robot.reverseRPS(float); float 0" in white.A red Scratch block with the text "robot.rotateRightRPS(float); float 0" in white.A red Scratch block with the text "robot.rotateLeftRPS(float); float 0" in white.

2.13. Blocos “robot.forwardRPM(float);”, “robot.reverseRPM(float);”, “robot.rotateRightRPM(float);” e “robot.rotateLeftRPM(float);”

Estes blocos vão colocar o robô à velocidade de rotações por minuto que coloca à frente do bloco pretendido. O bloco “robot.forwardRPM(float);” faz com que as duas rodas ande para a frente à velocidade que colocar e o bloco “robot.reverseRPM(float);” faz a mesma coisa só que anda para trás. O bloco “robot.rotateRightRPM(float);” escolhe a velocidade da roda direita e o bloco “robot.rotateLeftRPM(float);” da roda esquerda.

2.14. Blocos “robot.forwardMS(float);”, “robot.reverseMS(float);”, “robot.rotateRightMS(float);” e “robot.rotateLeftMS(float);”

Estes blocos vão colocar o robô à velocidade de metros por segundo que coloca à frente do bloco pretendido. O bloco “robot.forwardMS(float);” faz com que as duas rodas andem para a frente e à velocidade que colocar e o bloco “robot.reverseMS(float);” faz a mesma coisa só que anda para trás. O bloco “robot.rotateRightMS(float);” escolhe a velocidade da roda direita e o bloco “robot.rotateLeftMS(float);” da roda esquerda.

2.15. Bloco “robot.enSpeedAdj();”,

Esse bloco serve para ativar uma funcionalidade na biblioteca, para calcular a velocidade nas diferentes medidas RPS, RPM e MS. Sem este bloco os blocos que terminam em RPS, RPM e MS não funcionam.

2.16. Blocos “robot.getRightRPS();” e “robot.getLeftRPS();”

Estes blocos guardam o valor de rotações por segundo na variável “Nome” que pode ser alterado. O bloco “robot.getRightRPS();” guarda o valor da roda direita e o “robot.getLeftRPS();” da roda esquerda.



2.17. Blocos “robot.getRightRPM();” e “robot. getLeftRPM();”

Estes blocos guardam o valor de rotações por segundo na variável “Nome” que pode ser alterado. O bloco “robot.getRightRPM();” guarda o valor da roda direita e o “robot. getLeftRPM();” da roda esquerda.



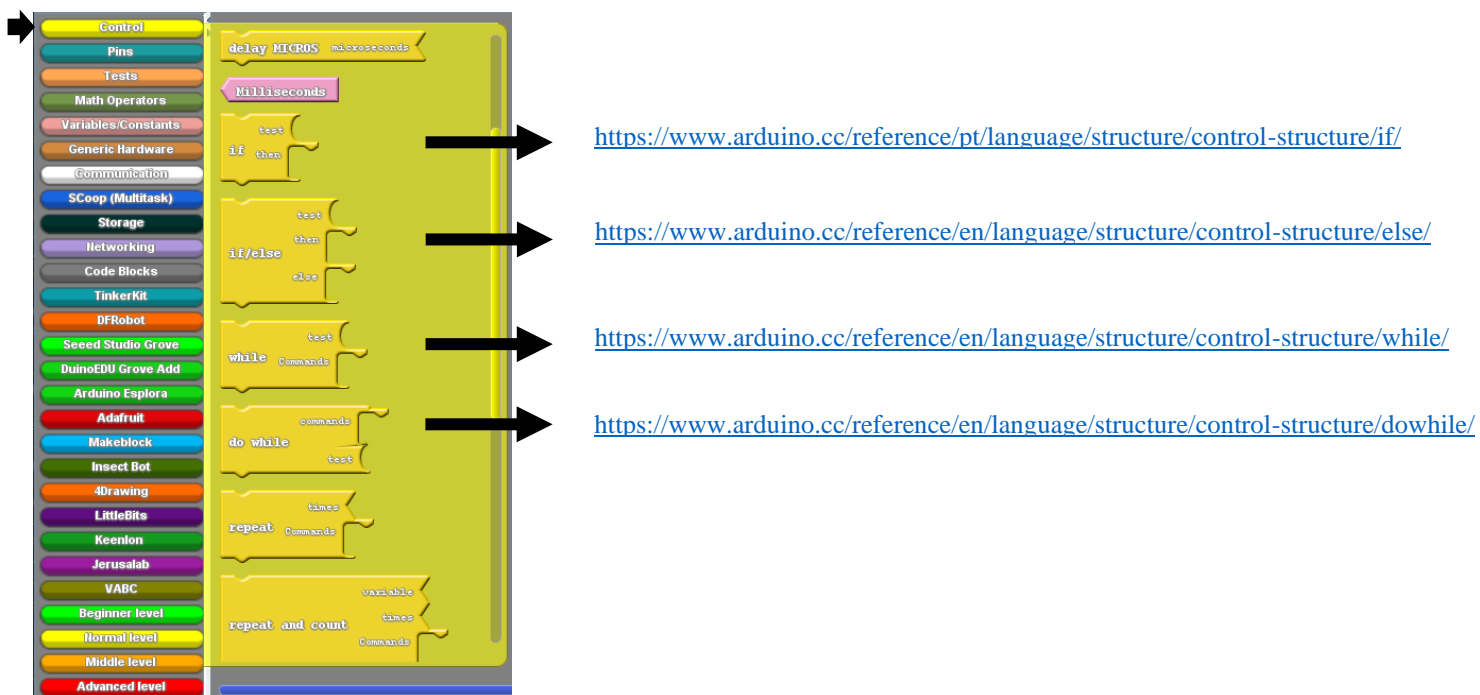
2.18. Blocos “robot.getRightMS();” e “robot. getLeftMS();”

Estes blocos guardam o valor de rotações por segundo na variável “Nome” que pode ser alterado. O bloco “robot.getRightMS();” guarda o valor da roda direita e o “robot.getLeftMS();” da roda esquerda.



2.19. Funções também muito utilizadas na programação

Pode investigar as várias abas e descobrir novas funções que possa adaptar no seu programa. Quando se sentir já preparado passa para o Arduino IDE sem o Ardublok e comece a programar linha a linha.



3. Exemplo de um programa

3.1. Robô vai andar em frente

No programa em baixo no Setup:

Bloco “#include <iModBot.h>” chama a biblioteca;

Bloco “robot.editMotorPin(INT1, INT2, INT3, INT4);” define os pinos que vão controlar as rodas;

Bloco “robot.begin();” configura a biblioteca e os pinos;

No loop:

Bloco “robot.forward();” faz com que o robô ande para a frente.

