Programar o robô por blocos

Alunos:

Abel Teixeira - 2180

Samuel Lourenço - 2180356

Docente: Luís Conde

Carlos Neves

Ano letivo: 2019/2020

**Índice**

[**Introdução** 3](#_Toc37304179)

[**Abrir o ArduBlock** 4](#_Toc37304180)

[**Guia básico do ArduBlock** 5](#_Toc37304181)

[**Nível Iniciante (Beginner level)** 6](#_Toc37304182)

[1. Bloco “Fashing Led” 6](#_Toc37304183)

[2. Bloco “Go in circles” 7](#_Toc37304184)

[3. Bloco “Walking Randomly” 7](#_Toc37304185)

[4. Bloco “Guide by Bluetooth” 8](#_Toc37304186)

[5. Bloco “Follow Line” 8](#_Toc37304187)

[**Nível Normal (Normal level)** 9](#_Toc37304188)

[1. Bloco “Define Speed” 9](#_Toc37304189)

[2. Bloco “Right” 9](#_Toc37304190)

[3. Bloco “Left” 9](#_Toc37304191)

[4. Bloco “Advace” 9](#_Toc37304192)

[5. Bloco “Back Off” 9](#_Toc37304193)

[6. Bloco “Dodge Obstacles” 10](#_Toc37304194)

[7. Bloco “END” 10](#_Toc37304195)

[8. Exemplo de um programa 10](#_Toc37304196)

[**Nível Medio (Middle level)** 11](#_Toc37304197)

[1.1. Bloco “ESP32 configuration” 11](#_Toc37304198)

[1.2. Blocos “Encoder configuratio” e o “Wheel controller” 11](#_Toc37304199)

[1.3. Bloco “Ultrasound Configuration” 11](#_Toc37304200)

[2. Blocos dos loop 11](#_Toc37304201)

[2.1. Bloco “Timing” 11](#_Toc37304202)

[2.2. Bloco “Divert Obestaculos” 12](#_Toc37304203)

[2.3. Blocos “Stop the 2 Wheels”, “Lesft Wheel Stop” e o “Right Wheel Stop” 12](#_Toc37304204)

[2.4. Blocos “Left Wheel Advances” e “Right Wheel Advances” 12](#_Toc37304205)

[2.5. Blocos “Left Wheel Recedes” e “Right Wheel Recedes” 12](#_Toc37304206)

[3. Exemplo de um programa 12](#_Toc37304207)

[**Nível Avançado (Adanced level)** 13](#_Toc37304208)

[1. Bloco “pinMode (pin , Mode);” 13](#_Toc37304209)

[2. Bloco “Serial.println” 13](#_Toc37304210)

[3. Bloco “delay (ms);” 13](#_Toc37304211)

[4. Bloco “digitalWrite (pin , value);” 14](#_Toc37304212)

[5. Bloco “digitalRead (pin);” 14](#_Toc37304213)

[6. Bloco “analogWrite (pin, value);” 14](#_Toc37304214)

[7. Bloco “analogRead (pin);” 15](#_Toc37304215)

[8. Funções também muto utilizadas na programação 15](#_Toc37304216)

# **Introdução**

Este projeto tens como objetivos de incentivar e ajudar qualquer utilizador que queira aprender a programar. Sendo que começa a programar no ArduBlock sempremente arrematando os blocos e contruído um programa. Sem ter que estar a escrever todas a linhas de programação que isso seria difícil para um utilizador que esteja a iniciar a sua aprendizagem.

Quando o utilizado acaba o seu programa no ArduBlock ele passa para o Arduíno IDE para a linguagem C++. Onde o utilizador pode ser o programa que fez por blocos e testá-lo enviando para o esp32 que está montado no carro. Mas também poderia enviar para um Arduíno qualquer dispositivo que leia linguagem C++.

Para o utilizador possa melhor a sua aprendizagem mais rápida, fizemos quatro níveis:

Nível Iniciante - O utilizados só arrenta um bloco para o loop;

Nível Normal - O utilizado já dá a intorções para o robô andar arrentando os blocos para o loop;

Nível Médio - O utilizado tem definir no setup os comandos que vai utilizar no loop arrentando os blocos para o locar exato;

Nível Avançado - O utilizado já vê os vários comandos escritos nos blocos só tem de definir se os pinos são entradas ou saídas no setup e no loop se eles estão ligados ou desligados. Arrastando os blocos e escrever os pinos e o modo deles.

Depois de passar os vários níveis o utilizados já está apto para programar no Arduíno IDE sem o ArduBlock, programando linha a linha.

# **Abrir o ArduBlock**

Se já tens o ArduBlock aberto pode saltar este passo e passar para o Nível Iniciante.

Caso ainda não tenha, comece por abrir o Arduíno IDE.

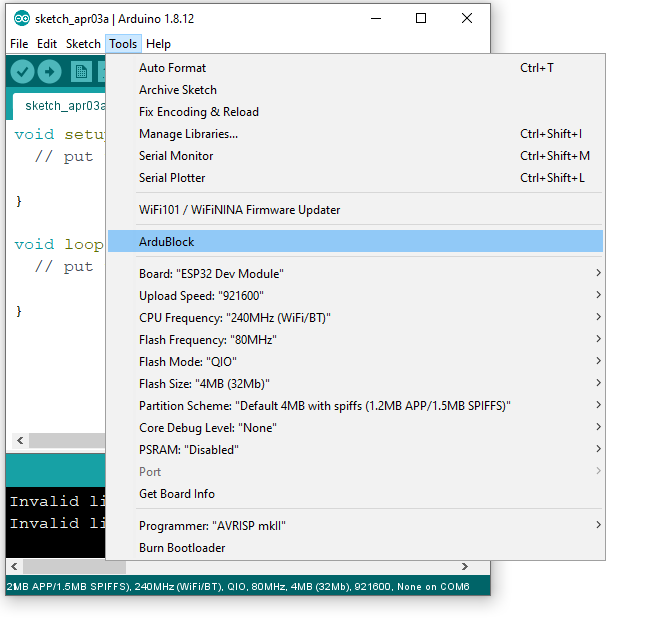


Figura 1- Arduino IDE

No Arduíno IDE clique “Tools” 🡺 “ArduBlock” e já tem ao ArduBlock aberto proto para contruir o programa.

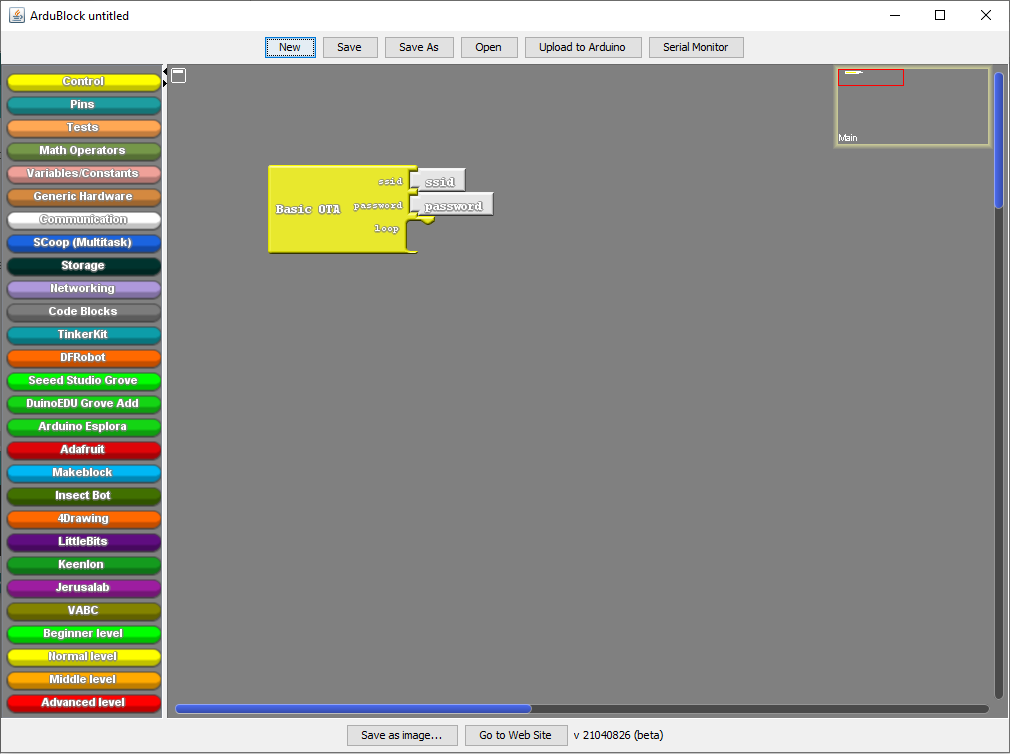


Figura 2- ArduBlock

# **Guia básico do ArduBlock**

Este guia é para encenar o funcionamento básico do ArduBlock para o utilizador. Saber como utilizar e o nome de cada ferramenta.

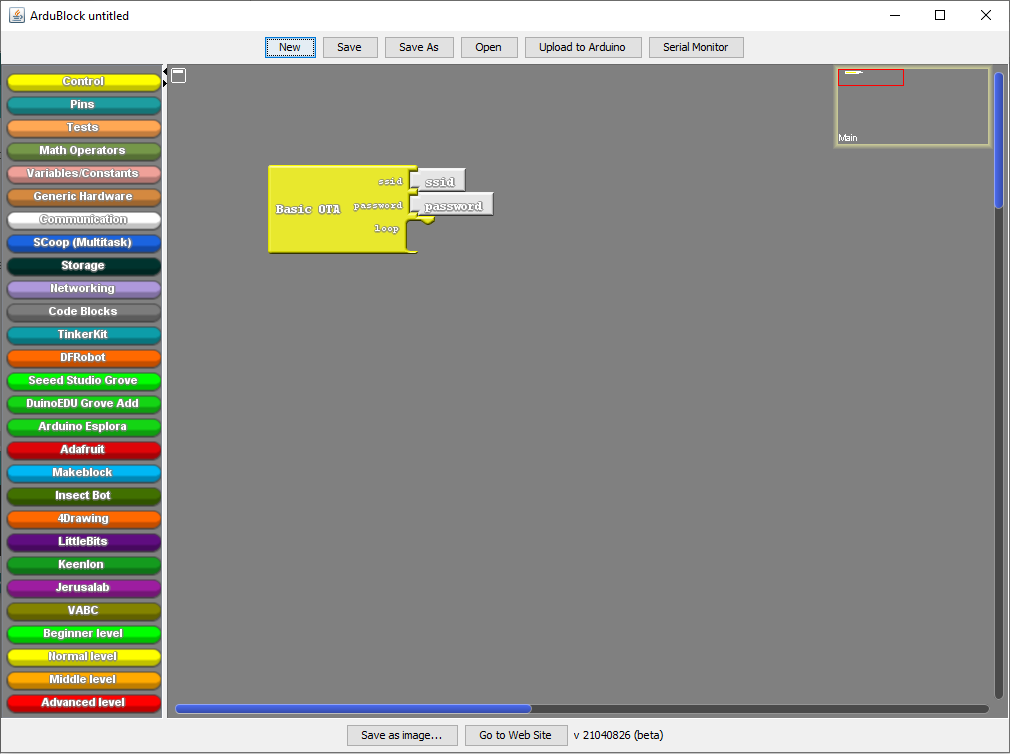
O botão “Upload to Arduíno” converte o programa de blocos para linguagem C++ no Arduíno IDE.

O botão “Open” abre o programa que guardaram.

Os botões “Save” e “Save as” guarda o vosso programa onde o colocarem.

No botão “New” abre um novo ArduBlock.

O botão “Serial Monitor” aber uma janela onde mostra valores o que nós enviarmos para lá.



O botão “Go to Web Site” reencaminha-te para o site do ArduBlock, este link: <http://blog.ardublock.com/> .

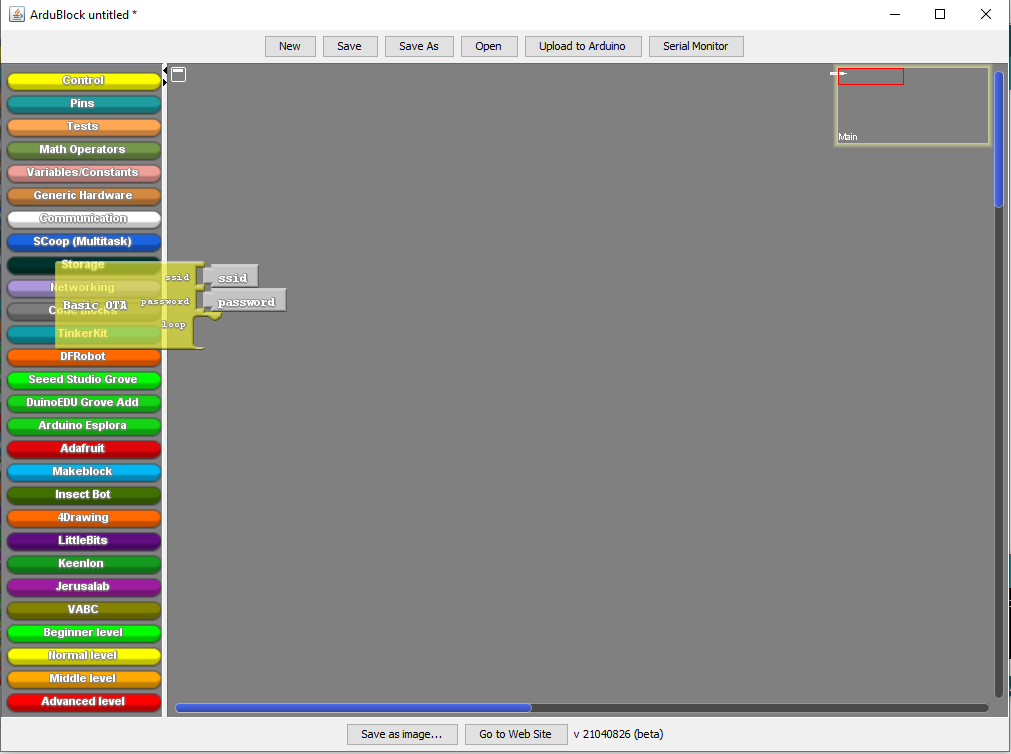
O botão “Save a image…” tira uma foto ao teu programa e guarda onde indicar.

Nesta coluna mostra as várias abas. Que ao clicar nelas mostra os blocos que tem dentro de cada uma.

Nós vamos começar por utilizar as 4 últimas abas.

Este retângulo mostra o vario espaço livre que tem para fazer o programa

Para eliminar algum bloco que não queiram basta arrentá-lo para cima da coluna da abas e largar e ele é eliminado do espeço de trabalho como pode ver na figura a baixo.



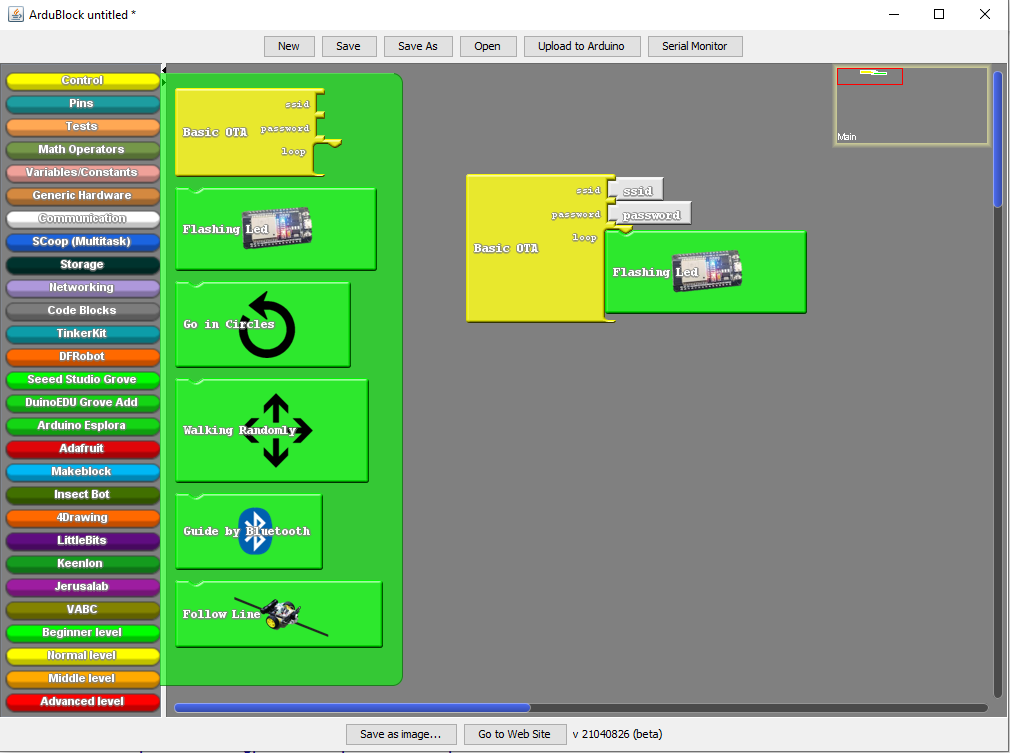
# **Nível Iniciante (Beginner level)**

Neste nível basto só colocar um bloco no loop do bloco “Basic OTA”.

Caso não esteja ligado ao ESP32 com o cabo micro USB e queira ligar do wifi. Tem de clicar no bloco branco “ssid” e colocar o nome da internet e no bloco “password” colocar a passe da internet. Fica a comunicar com o ESP32 sem fios, sem esquecer da alimentação dele.

1. Bloco “Fashing Led”

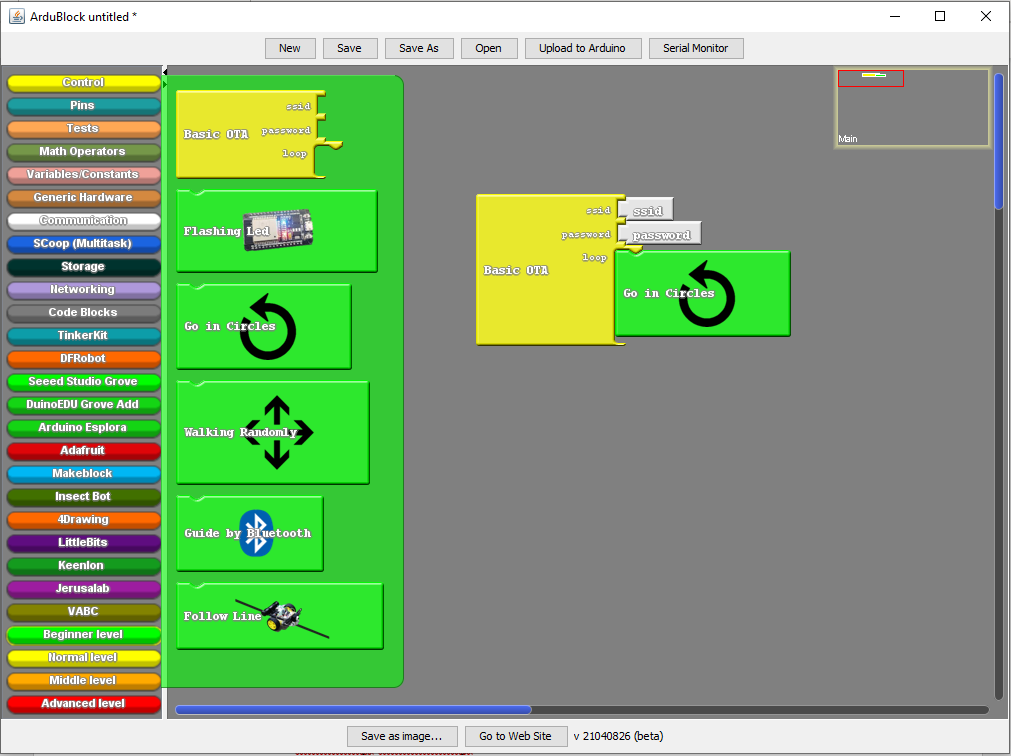
Neste primeiro bloco verde vai fazer piscar o led azul que está ligado no pino 2. O objetivo é fazer piscar o led é para saber se o ESP32 está a comunicar por wifi. Para isso tem de abrir a aba verde com o nome “Beginner level” e arrastar o bloco “Fashing Led” para dentro do loop do bloco “Basic OTA”. Como podem ver na figura abaixo.



Não se esqueçam de colocar o “ssid” e a “password”. Depois clique no botão “Upload to Arduíno” que vai converter o programa que fizemos para a linguagem C++ no Arduíno IDE e envia para o ESP32. Que vais fazer piscar o led.

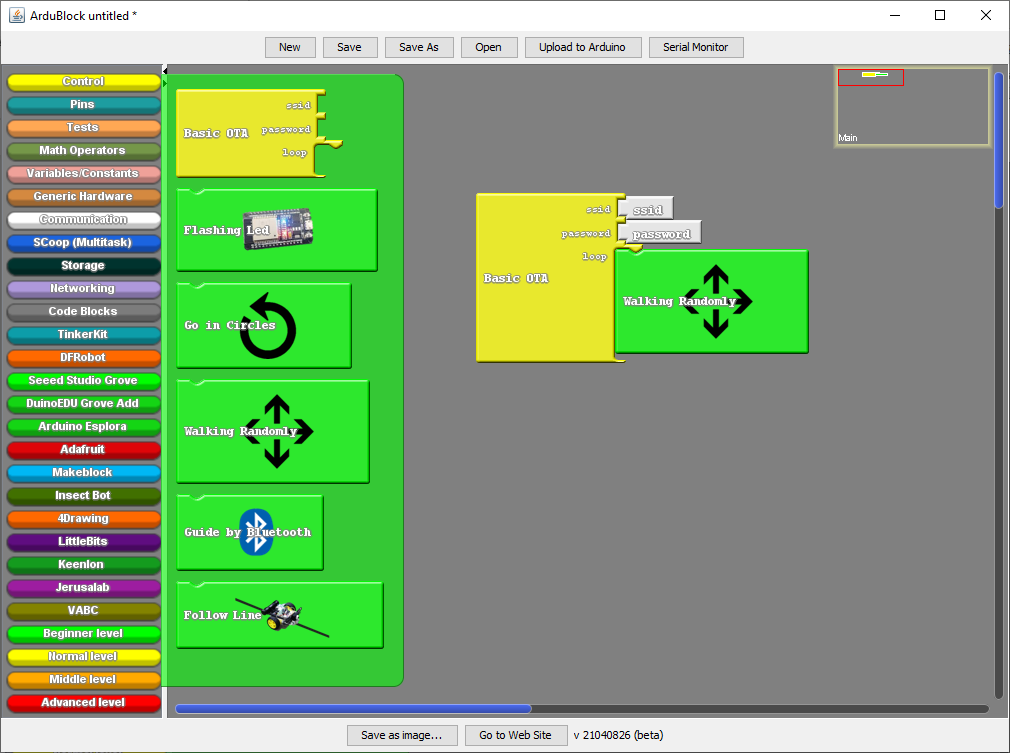
1. Bloco “Go in circles”

No segundo bloco o robô vai andar a roda. O objetivo é saber se as rodas estão bem ligadas. Para isso basta só colocar o bloco “Go in circles” no loop retirando o loco anterior.



1. Bloco “Walking Randomly”

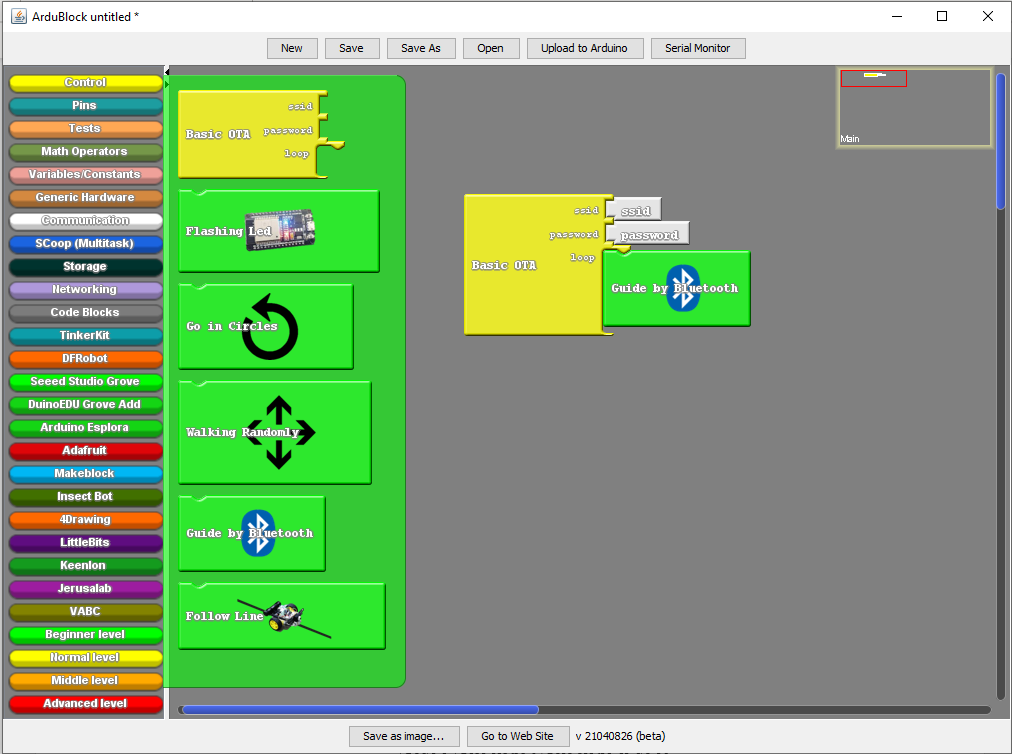
No terceiro bloco o robô vai andar aleatoriamente. O objetivo é saber se tá tudo bem ligado e as ordens do ESP32 (avançar, recuar, esquerda, direita e parar) estão a ser recebidas. Para isso basta só colocar o bloco “Walking Randomly” no loop retirando o loco anterior.



1. Bloco “Guide by Bluetooth”

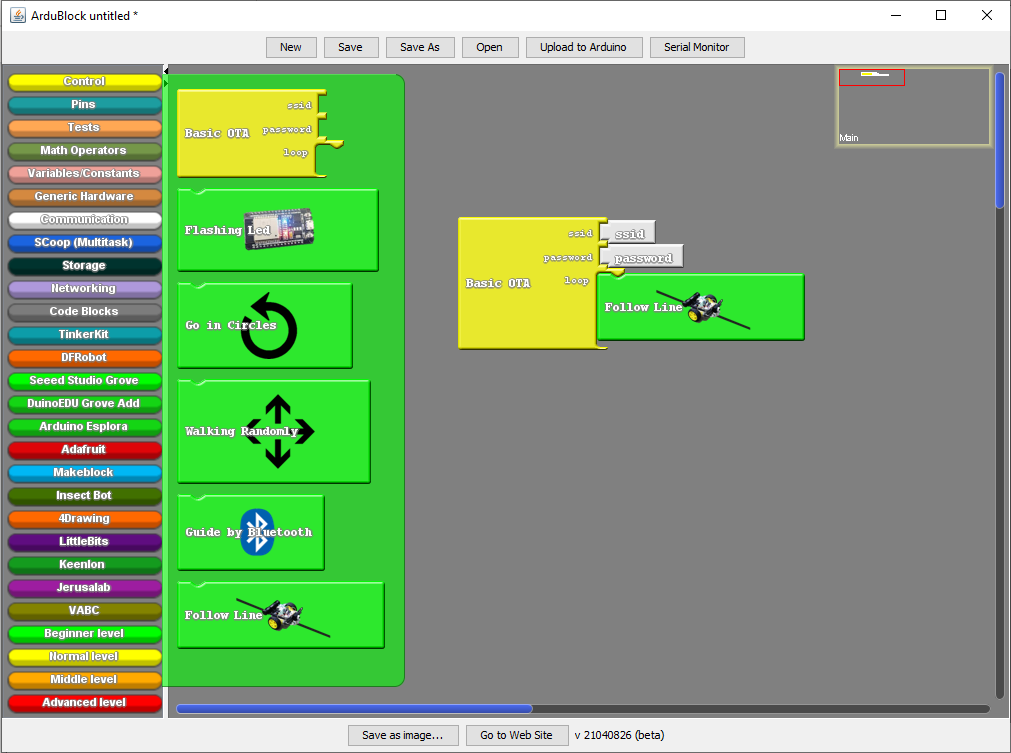
No quarto bloco o robô vai ser guiado por um telemóvel através do Bluetooth.

Bloco ainda não acabado.



1. Bloco “Follow Line”

No quinto bloco o robô vai seguir a linha preta basta só colocar o bloco no loop e colocar o robô em cima da linha e ele vai seguir.



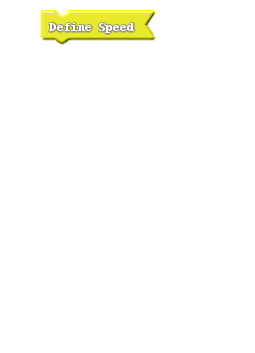
# **Nível Normal (Normal level)**

Neste nível ao contrário do anterior, o utilizado escolhe

1. Bloco “Define Speed”

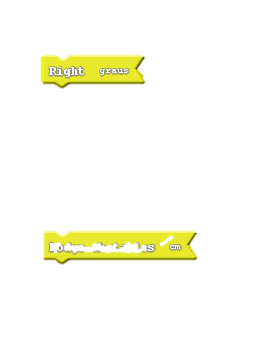
Com este bloco define a velocidade que o robô vai andar. Escrevendo do bloco a frente 1, 2 ou 3. Sendo que o número 3 corresponda ao 255 que é igual a 5V, a tenção máxima do ESP32

Nota: Este bloco tem de ser o primeiro.



1. Bloco “Right”

Ao colocar o bloco “Right” vai fazer com que o robô vire para a direita os graus que colocar no bloco.



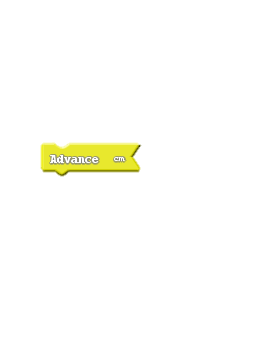
1. Bloco “Left”

Ao colocar o bloco “Left” vai fazer com que o robô vire para a esquerda os graus que colocar no bloco.



1. Bloco “Advace”

Ao colocar o bloco “Advace” vai fazer com que o robô ande para frente os centímetros que colocar no bloco.



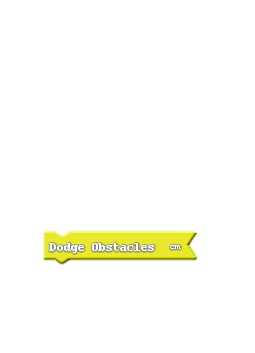
1. Bloco “Back Off”

Ao colocar o bloco “Back Off” vai fazer com que o robô recue os centímetros que colocar no bloco.



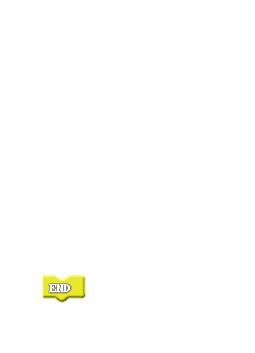
1. Bloco “Dodge Obstacles”

Ao colocar o bloco “Dodge Obstacles” vai fazer com que o robô ande para frente até que esteja a uma distância de um obstáculo. A distância é determinada pelo número que colocar no bloco a frente do “cm”.



1. Bloco “END”

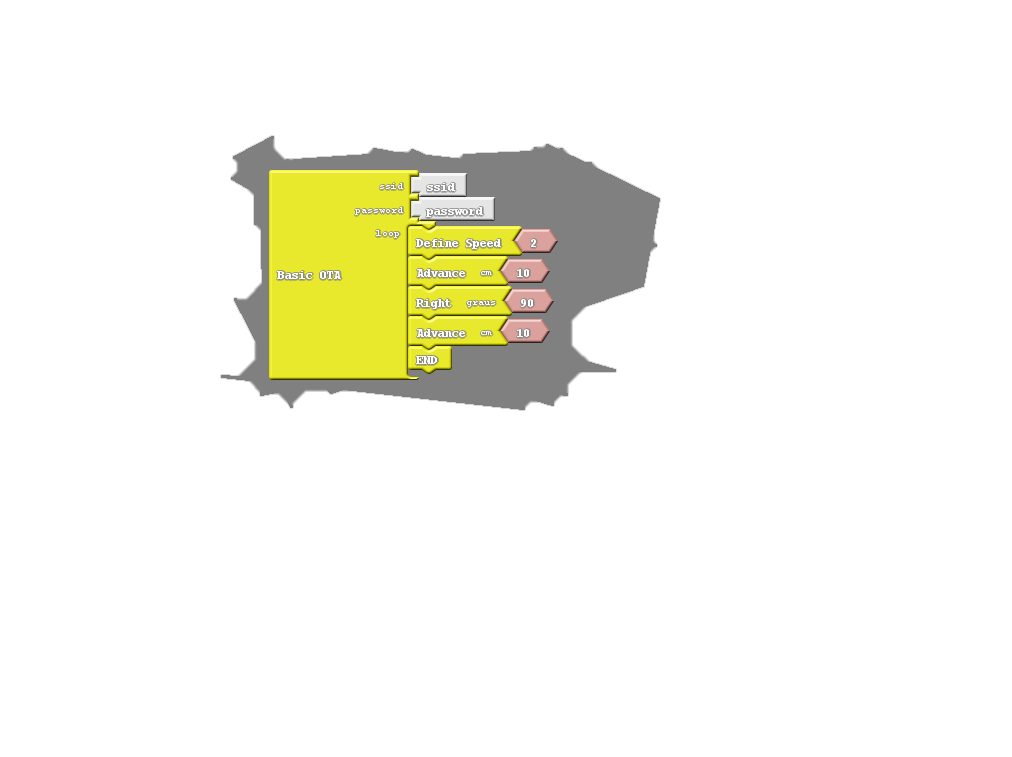
Este bloco “END” coloca-se no finar de todos os blocos para que o robô não repta o percurso novamente.



1. Exemplo de um programa

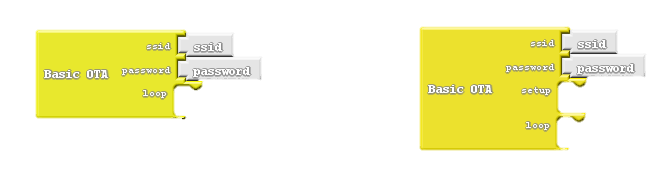
Na imagem em baixo é um exemplo da montagem de um programa.

Comecei com o bloco “Define Speed” e defini a velocidade 2, no segundo bloco digo para o robô avançar 10cm para frente. O terceiro bloco digo para virar à direita 90graus. No quarto bloco digo para o robô avançar 10cm para frente e o quinto bloco “END” digo que é o final do programa.



# **Nível Medio (Middle level)**

No nível medio vai altera o bloco “Basic OTA” só com loop para um bloco com o mesmo nome que esta na aba “Middle level” para se começam a habituar ao modo de programação em C++.

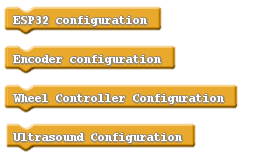


1. Blocos dos Setup

Os quatro blocos a baixo são colocados no setup para que os blocos do loop funcionem.

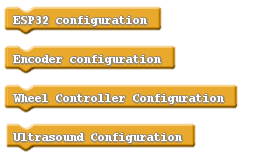
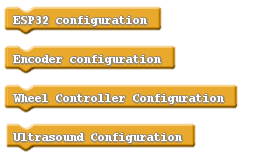
* 1. Bloco “ESP32 configuration”

Este bloco tem de estar presente em todos os programas feitos no nível medio (Middle level) para configurar a biblioteca.



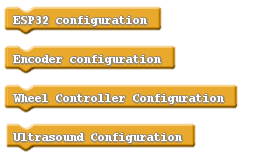
* 1. Blocos “Encoder configuratio” e o “Wheel controller”

Estes dois blocos servem para que os blocos do loop que controlam a rodas funcionem.



* 1. Bloco “Ultrasound Configuration”

Este bloco serve para que os blocos do loop do sonar funcionem.

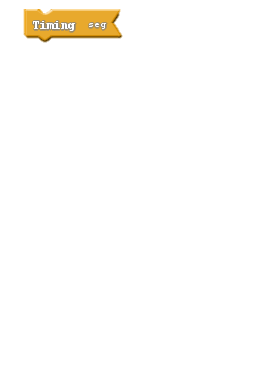


1. Blocos dos loop

Como os blocos no steup, vamos passar para o loop aqui conforme montar os blocos vai desenvolvendo o programa que o robô vais desempenhar.

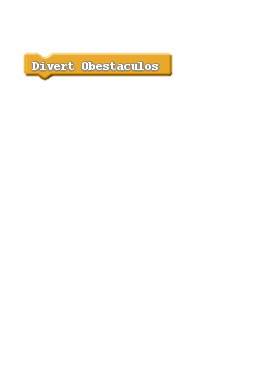
* 1. Bloco “Timing”

Ao colocar o bloco “Timing” vai temporizar os segundos que colocar no bloco a frente antes de passar para o próximo bloco.



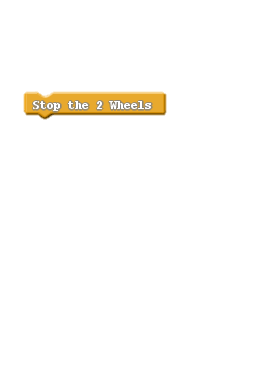
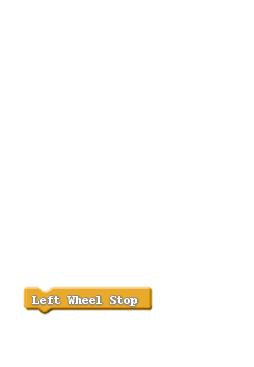
* 1. Bloco “Divert Obestaculos”

Bloco ainda não acabado.



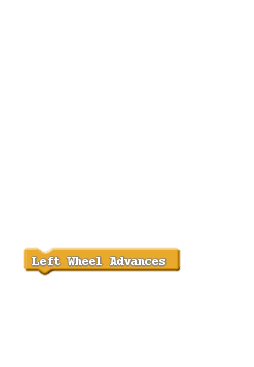
* 1. Blocos “Stop the 2 Wheels”, “Lesft Wheel Stop” e o “Right Wheel Stop”

Os três blocos fazem com que as rodas parem. O bloco “Stop the 2 Wheels” faz parar as duas rodas ao mesmo tempo. O bloco “Lesft Wheels Stop” faz parar só a roda esquerda e o bloco “Right Wheels Stop” faz parar só a roda direita.



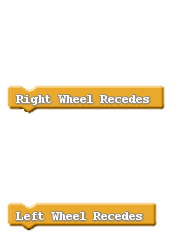
* 1. Blocos “Left Wheel Advances” e “Right Wheel Advances”

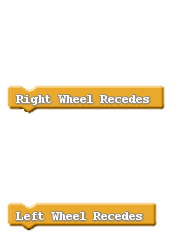
Os dois blocos fazem com que o robô possa andar para frente. O bloco “Left Wheel Advances” faz com que a roda ande para a frente e o bloco “Right Wheel Advances” faz o mesmo só que com a roda direita.



* 1. Blocos “Left Wheel Recedes” e “Right Wheel Recedes”

Os dois blocos fazem com que o robô recue. O bloco “Left Wheel Advances” faz com que a roda ande para a trás e o bloco “Right Wheel Advances” faz o mesmo só que com a roda direita.

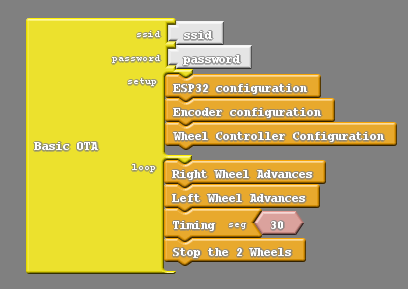




1. Exemplo de um programa

O seguinte exemplo faz com que o robô anda para a frente durante 30 segundos.

No setup coloquei os blocos necessários para controlar as rodas. No loop os dois primeiros blocos fazem com que a roda direita e esquerda ande para a frente, no terceiro bloco temporiza durante 30 segundos e o quarto bloco manda parar a rodas.



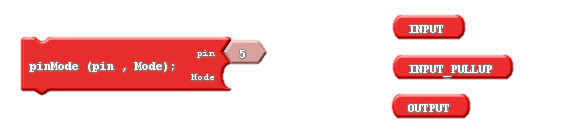
# **Nível Avançado (Adanced level)**

No nível avançado já é como se tivesse a programar os blocos tem o nome da funções que se usam na linguagem C++.

1. Bloco “pinMode (pin , Mode);”

Este bloco define cada pino se ele é entrada (INPUT) ou saída (OUTPUT). Alterando o número 5 para o pino que pretendem e arrentado o bloco mais indicado para o “Mode”.

Neste link tem uma descrição mais detalhada e um exemplo: <https://www.arduino.cc/reference/en/language/functions/digital-io/pinmode/>



1. Bloco “Serial.println”

Este bloco serve para escrever um valou ou um texto para o serial monitor despendo do que colocar no bloco val.

Neste link tem uma descrição mais detalhada e um exemplo: <https://www.arduino.cc/reference/pt/language/functions/communication/serial/println/>



1. Bloco “delay (ms);”

Este bloco temporiza quanto milissegundos colocar no bloco a frente.

Neste link tem uma descrição mais detalhada e um exemplo:

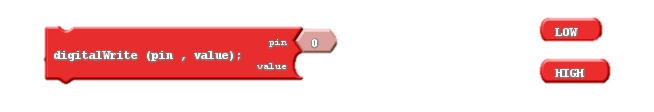
<https://www.arduino.cc/reference/en/language/functions/time/delay/>



1. Bloco “digitalWrite (pin , value);”

Este bloco define o pino se ele está ligado (HIGH) ou desligado (LOW). Alterando o número 0 para o pino que pretendem e arrentado o bloco mais indicado para o “value”.

Neste link tem uma descrição mais detalhada e um exemplo: <https://www.arduino.cc/reference/en/language/functions/digital-io/digitalwrite/>



1. Bloco “digitalRead (pin);”

Este bloco lé o estado do pino se está ligado (HIGH) ou desligado (LOW). Alterando o número 0 para o pino que pretendem.

Neste link tem uma descrição mais detalhada e um exemplo: <https://www.arduino.cc/reference/en/language/functions/digital-io/digitalread/>



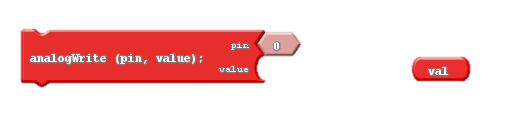
1. Bloco “analogWrite (pin, value);”

Este bloco define cada pino com a tenção que varia de 5 a 0 voltes sendo que 5V é 255 e 0V é 0. Alterando o número 0 para o pino que pretendem e arrentado o bloco “val” colocando um número de 0 a 255. Se pretender uma tenção mais especifica pode utilizar uma regra de três simples como mostra altera o “v” para a tenção que quere e o valor que der (x) coloca no bloco “val”.

255 – 5

x – v

Neste link tem uma descrição mais detalhada e um exemplo: <https://www.arduino.cc/reference/en/language/functions/analog-io/analogwrite/>



1. Bloco “analogRead (pin);”

Este bloco lé o valor do pino de 255 (5V) a 0 (0V). Alterando o número 0 para o pino que pretendem.

Neste link tem uma descrição mais detalhada e um exemplo: <https://www.arduino.cc/reference/en/language/functions/analog-io/analogread/>



1. Funções também muto utilizadas na programação

Pode investigar as varias abas e descobri 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.



https://www.arduino.cc/reference/en/language/structure/control-structure/dowhile/

<https://www.arduino.cc/reference/en/language/structure/control-structure/while/>

<https://www.arduino.cc/reference/en/language/structure/control-structure/else/>

<https://www.arduino.cc/reference/pt/language/structure/control-structure/if/>