



Cofinanciado pela  
União Europeia

Financiado pela União Europeia. Os pontos de vista e as opiniões expressas são as do(s) autor(es) e não refletem necessariamente a posição da União Europeia ou da Agência de Execução Europeia da Educação e da Cultura (EACEA). Nem a União Europeia nem a EACEA podem ser tidos como responsáveis por essas opiniões.

# O projeto do carro robótico DIY



Introduzir as 5 Grandes Ideias da Inteligência Artificial  
utilizando a Internet das Coisas no ensino STEM

T2.4 Conceção de projetos IoT e desenvolvimento de recursos

06.10.2023 | EDUMOTIVA  
NÚMERO DO PROJECTO: 2022-1-FR01-KA220-SCH-000085611

# Projetos IoT AI4STEM

## Projeto: O projeto do carro robótico DIY

### Copyright

© Direitos de autor do Consórcio AI4STEM  
2022-1-FR01-KA220-SCH-000085611  
Todos os direitos reservados.



Projetos IoT AI4STEM Projeto: O projeto do carro robótico DIY © 2023 pelo [Consórcio AI4STEM](#)  
está licenciado sob [Atribuição-NãoComercial-CompartilhaIgual 4.0 Internacional](#)

## Índice

1. introdução ao projeto .....	4
1.1 O âmbito do projeto .....	4
1.2 Os grupos-alvo .....	5
1.3 Objetivo do presente documento.....	5
2. Glossário da unidade.....	5
3. Apresentamos o "Carro robótico DIY que pode ser controlado e navegado através de comandos de voz"	5
3.1 Descrição .....	5
3.2 Objetivos e resultados da aprendizagem.....	7
3.3 Duração prevista da unidade .....	7
3.4 Atividade 1 - Introduzir a grande ideia da perceção através da IoT: .....	8
3.4.1 Descrição .....	8
3.4.2 Hardware .....	8
3.4.3 Configuração .....	8
3.4.4 Experiência 1 .....	16
3.5 Atividade 2: Introduzir a ideia de Representação e Raciocínio .....	18
3.5.1 Descrição .....	18
3.5.2 Criar uma árvore de decisão .....	18
3.5.3 Conceção e programação da aplicação.....	19
3.5.4 Experiência 2 .....	28
3.6 Atividade 3: Introduzir a ideia de aprendizagem através do treino de um modelo de reconhecimento de comandos de voz .....	30
3.6.1 Descrição .....	30
3.6.2 Utilizar o classificador de áudio pessoal para treinar um modelo.....	30
3.6.3 Experiência 3 .....	35
3.7 Atividade 4: Introduzir a ideia de Interação Natural através da integração de um modelo treinado numa aplicação de IA .....	36
3.7.1 Descrição .....	36
3.7.2 Integrar o modelo treinado na aplicação de IA .....	37
3.7.3 Experiência 4 .....	48
3.8 Atividade 5: Introduzir a ideia de impacto social.....	49
3.8.1 Descrição .....	49
3.9 Material e recursos .....	50



3.10 O hardware do carro robótico .....	51
---	----



## 1. introdução ao projeto

O presente projeto centra-se na criação de um carro robótico DIY que pode ser controlado por comandos de voz, ao mesmo tempo que é capaz de recolher uma série de dados (como a temperatura, o nível de luz, a distância, a aceleração, etc.) que podem levar a determinadas decisões para otimizar o seu desempenho. Esta intervenção de aprendizagem baseada em projetos ajudará tanto os professores como os alunos a serem introduzidos nos domínios da IA e da IoT, através da lente das 5 Grandes Ideias (nomeadamente Perceção, Representação e Raciocínio, Aprendizagem, Interação Natural e Impacto Social), bem como à luz da Robótica, através da implementação de uma série de práticas e atividades práticas e baseadas em computador. No que diz respeito à IA, serão introduzidos ao Reconhecimento de Fala e à forma como este serviço pode ser utilizado para navegar num artefacto robótico. Para o efeito, serão introduzidos nos processos de conceção e programação de uma aplicação baseada em IA utilizando o software MIT App Inventor, bem como no treino de um modelo para classificar a informação recebida utilizando o ambiente Personal Audio Classifier. No que diz respeito à Robótica, aprenderão a construir um carro robótico utilizando o microcontrolador BBC micro:bit e vários componentes electrónicos compatíveis, bem como a programar este artefacto robótico utilizando o software baseado em blocos Makecode. O projeto será dividido em 5 atividades. Cada uma destas atividades gira em torno de uma das 5 Grandes Ideias. Durante estas atividades, os alunos serão convidados a concentrarem-se em diferentes partes do processo de implementação e a envolverem-se em diferentes aspetos da IA e da IoT. <sup>st</sup>As atividades incluirão orientações para os professores e várias tarefas sugeridas para os alunos, a fim de garantir uma introdução e implementação harmoniosas de todos os conceitos acima referidos e dos seus aspetos inerentes, conduzindo, em última análise, à aquisição e desenvolvimento de várias competências do século XXI, como a criatividade, o pensamento crítico, a resolução de problemas e a colaboração.

### 1.1 O âmbito do projeto

Através deste projeto, os alunos serão introduzidos no domínio da IA e das 5 Grandes Ideias, bem como no domínio da IoT, à luz da robótica. Em particular, através das 5 atividades e das tarefas associadas, que giram em torno da construção e da programação do carro robótico, bem como da conceção e da programação da aplicação de IA, os alunos compreenderão melhor as 5 Grandes Ideias e aprofundarão alguns dos mecanismos fundamentais da IA e da IoT. Mais especificamente, na atividade 1<sup>st</sup>, os alunos serão apresentados à IoT, aprendendo a programar o seu carro robótico para recolher e monitorizar uma série de dados ambientais através de um serviço de plataforma de análise da IoT (ou seja, ThingSpeak). Através das tarefas aí contidas, os alunos serão introduzidos à ideia de Perceção. Na atividade 2<sup>nd</sup>, os alunos serão introduzidos à ideia de representação e raciocínio, aprendendo como o carro robótico pode "pensar" e como os dados podem ser representados de várias formas. Para o efeito, aprenderão a criar árvores de decisão e fluxogramas e utilizarão esta informação para criar uma aplicação que permitirá navegar no carro robótico utilizando comandos de voz. Assim, vão aprender a utilizar o serviço de IA de reconhecimento de voz e a avaliar os resultados da sua implementação. Na atividade 3<sup>rd</sup>, os alunos serão introduzidos à ideia de aprendizagem e, utilizando uma ferramenta de aprendizagem automática (ML) (o classificador de áudio pessoal), familiarizar-se-ão com os métodos que o carro robótico pode aprender a partir de dados e através de ML. Na Atividade 4<sup>th</sup>, serão introduzidos à ideia de Interação Natural, integrando o modelo treinado (produzido na Atividade 3<sup>rd</sup>) na aplicação criada e avaliando os resultados à luz de uma implementação física, tomando assim consciência das limitações dos sistemas de IA em termos de interação natural. Por fim, na Atividade 5<sup>th</sup> será introduzida a ideia de Impacto Societal, refletindo sobre toda a sua experiência e tomando consciência das vantagens, desvantagens e riscos

subjacentes à utilização da IA e da IoT no nosso quotidiano. <sup>st</sup>O objetivo final deste projeto é aumentar a confiança dos alunos em todos os conceitos e aspetos acima mencionados e criar experiências de aprendizagem significativas que os ajudem a desenvolver uma série de competências do século XXI, como a criatividade, o pensamento crítico, a resolução de problemas e a colaboração.

## 1.2 Os grupos-alvo

O projeto destina-se a estudantes entre os 12 e os 16 anos de idade. Os alunos devem ter alguma experiência com ambientes de programação baseados em blocos.

## 1.3 Objetivo do presente documento

O objetivo deste documento é fornecer aos professores algumas ideias concretas e atividades de aprendizagem sobre a forma como os conceitos de IA e IoT podem ser introduzidos e ensinados de forma significativa aos alunos, através da lente da robótica e de uma série de tarefas práticas.

## 2. Glossário da unidade

Palavra	Definição
<b>ThingSpeak</b>	Uma plataforma de serviços IoT para monitorizar os dados recolhidos pelo módulo WiFi ESP8266
<b>Inventor de aplicações do MIT</b>	Software para criar aplicações
<b>Reconhecedor de fala</b>	Um serviço de IA no MIT App Inventor que reconhece a fala e a transforma num texto
<b>Aplicação React ou classificador de áudio pessoal</b>	Uma ferramenta de classificação de áudio de aprendizagem automática (ML) e de formação de modelos compatível com o MIT App Inventor

## 3. Apresentamos o "Carro robótico DIY que pode ser controlado e navegado através de comandos de voz"

### 3.1 Descrição

Este projeto apresentará aos alunos as 5 Grandes Ideias da IA e da IoT no ensino STEM através da criação de um carro robótico DIY (*Figura 1*), programado para ser navegado através de comandos de voz. Especificamente, os alunos serão encorajados a conceber e montar o carro robótico utilizando vários componentes eletrónicos e materiais simples, e a programá-lo utilizando ambientes de programação baseados em blocos. Aprenderão a recolher dados, em tempo real, e a refletir sobre os resultados obtidos e a forma como podem ser utilizados para tomar decisões específicas sobre o desempenho do carro

robótico. Além disso, serão incentivados a conceber e programar uma aplicação que utilize o serviço de IA de reconhecimento de voz para permitir a navegação verbal do carro robótico. Além disso, aprenderão a criar um modelo treinado para otimizar o desempenho da aplicação.

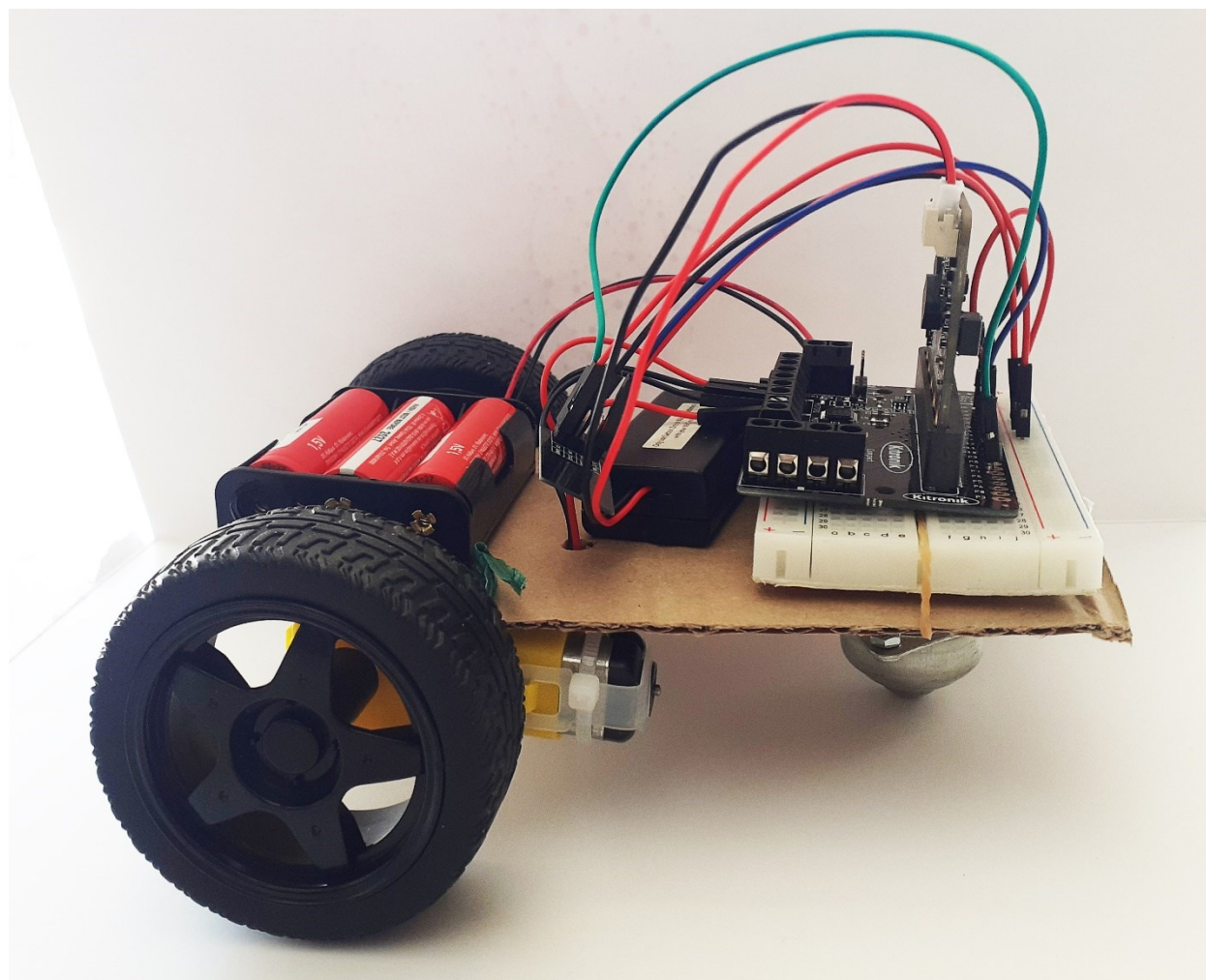


Figura 1: O carro robótico de bricolage

Para prosseguir com as atividades contidas neste documento, deve primeiro seguir as instruções incluídas no ficheiro "[T2.4\\_Criar\\_o\\_carro\\_robótico.pdf](#)", enquanto pode incentivar os seus alunos a realizar algumas atividades de programação de aquecimento, semelhantes às contidas no ficheiro "[T2.4\\_Atividades\\_de\\_programação\\_de\\_aquecimento\\_para\\_o\\_carro\\_robótico.pdf](#)". Recomenda-se vivamente que divida os seus alunos em equipas e que os incentive a discutir os diferentes aspetos de cada atividade. Para facilitar este processo, neste documento encontrará ligações para material que o pode ajudar a introduzir cada passo do projeto sem problemas, bem como algumas sugestões de perguntas que o podem ajudar a iniciar um diálogo com os seus alunos sobre diferentes partes e aspetos do projeto.



## 3.2 Objetivos e resultados da aprendizagem

Após a conclusão bem-sucedida desta unidade, os alunos devem ser capazes de

- Identificar a presença e a utilização da robótica de IA na vida quotidiana
- Discutir e compreender o papel da IA na robótica
- Explicar e debater os diferentes aspetos da integração da IA num projeto robótico através do reconhecimento da fala e de comandos de voz
- Compreender como funcionam os serviços de IA, como o reconhecimento de voz
- Compreender como as árvores de decisão ou os fluxogramas podem mostrar possíveis caminhos lógicos e também levar a decisões sobre operadores que podem ser utilizados numa fase posterior
- Debater o papel da aprendizagem automática (ML) na robótica de IA e o modo como a ML pode ser utilizada para treinar um artefacto robótico a perceber o seu ambiente
- Explicar os conceitos básicos da classificação áudio
- Identificar e discutir vantagens e riscos da aplicação de comandos de voz na condução
- Identificar e discutir as vantagens e os riscos da classificação áudio
- Explicar as construções/conceitos básicos de programação relacionados com a implementação de métodos de conversão de voz em texto
- Compreender os principais conceitos subjacentes à IoT e as implicações da monitorização dos dados
- Refletir sobre o impacto das decisões baseadas em dados na vida quotidiana
- Compreender que serviços como o reconhecimento de voz são suscetíveis de erro

Os alunos também aprenderão a:

- Construir um artefacto robótico e criar circuitos no âmbito de uma construção robótica
- Monitorizar dados utilizando um serviço de plataforma analítica IoT
- Criar árvores de decisão e fluxogramas para representar um tipo de informação
- Utilizar comandos de programação associados a métodos de IA para abordar um comportamento específico de um artefacto robótico
- Programar um robô para receber instruções através de comandos de voz
- Exprimir as suas ideias através da programação
- Utilizar a ferramenta ML Personal Audio classifier para classificar diferentes sons
- Avaliar os resultados produzidos por uma ferramenta de ML
- Melhorar um modelo treinado com base na avaliação
- Investigar fatores nos dados de formação que possam conduzir a enviesamentos

## 3.3 Duração prevista da unidade

Trata-se de um projeto bastante extenso, que necessita de várias horas para abordar corretamente todos os aspetos incluídos. A duração que se segue é indicativa e pode variar consoante a idade e o nível dos seus alunos.

**Atividade 1:** 4 - 6 horas

**Atividade 2:** 8 - 15 horas

**Atividade 3:** 2 - 4 horas

**Atividade 4:** 2 - 4 horas

**Atividade 5:** 1 - 2 horas



## 3.4 Atividade 1 - Introduzir a grande ideia da percepção através da IoT:

### 3.4.1 Descrição

Esta atividade é uma intervenção de aprendizagem de aquecimento para apresentar aos alunos a ideia de percepção à luz da IoT e da robótica. Os alunos vão explorar a forma como o seu carro robótico pode sentir e perceber o seu ambiente, recolhendo e armazenando dados, que podem mais tarde ser utilizados para tomar algumas decisões sobre o desempenho do carro. Em particular, vão explorar a forma como os sensores incorporados na placa BBC micro:bit, ou os sensores que podem ser ligados à placa (por exemplo, um sensor ultrassónico) podem recolher dados e ser monitorizados utilizando um serviço de plataforma analítica IoT. O serviço utilizado para as necessidades desta atividade é o ThingSpeak, uma plataforma que permite a agregação, visualização e análise de fluxos de dados em direto na nuvem.

**Sugestão:** Compreender um ambiente deste tipo (ou seja, ThingSpeak) pode ser um pouco difícil para os alunos. Por isso, dependendo do nível deles, pode decidir introduzir esta atividade após o final das atividades 2 a 4, de modo a perceberem melhor como a IoT pode ajudar a otimizar o carro robótico. Se os seus alunos tiverem menos de 12 anos, pode saltar esta atividade.

### 3.4.2 Hardware

Para além do BBC micro:bit, o hardware necessário para esta atividade é o módulo Wi-Fi ESP8266 e, opcionalmente, alguns outros sensores, como um sensor ultrassónico (de preferência o HC-SR04).

### 3.4.3 Configuração

Antes de começar a ligar os cabos e a codificar, é necessário criar um canal para monitorizar os dados recebidos. Para tal, aceda ao sítio Web do ThingSpeak (<https://thingspeak.com/>) e inicie sessão na sua conta, introduzindo o seu e-mail (1.), ou crie uma conta clicando na opção "Create One" (2.) (Figura 2).

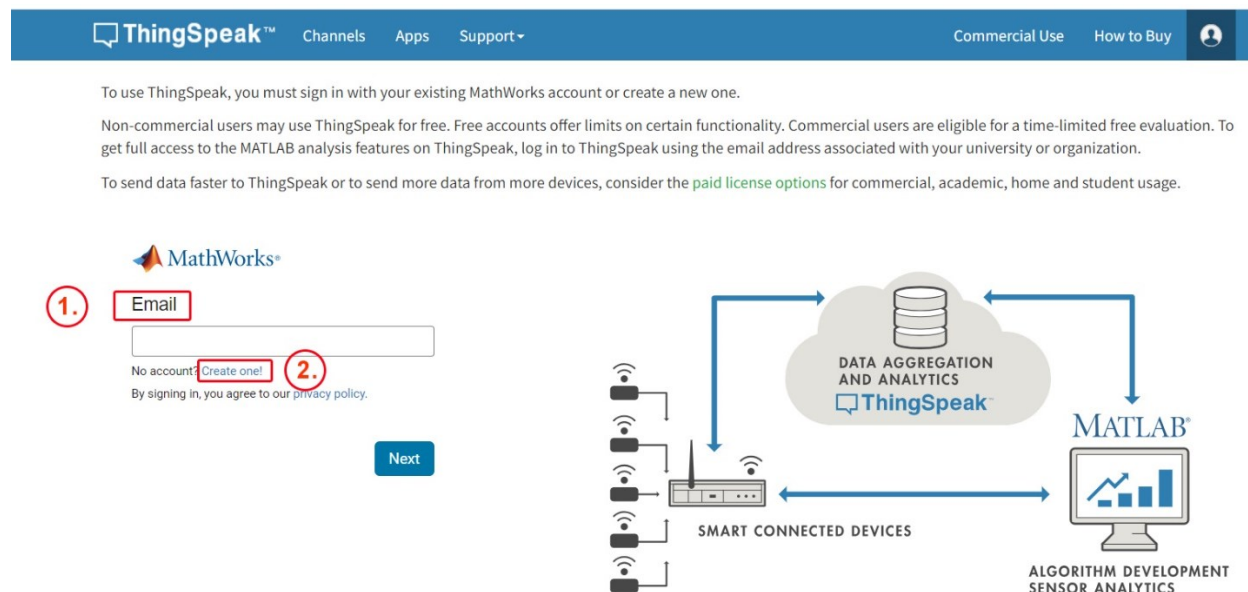


Figura 2: Menu para iniciar sessão ou criar uma nova conta no ThingSpeak

Em seguida, aceda ao menu Channels (Canais) e clique no botão "New Channel" (Novo canal) em "My Channels" (Os meus canais) para criar um novo canal para registar e apresentar os dados dos seus sensores (Figura 3).

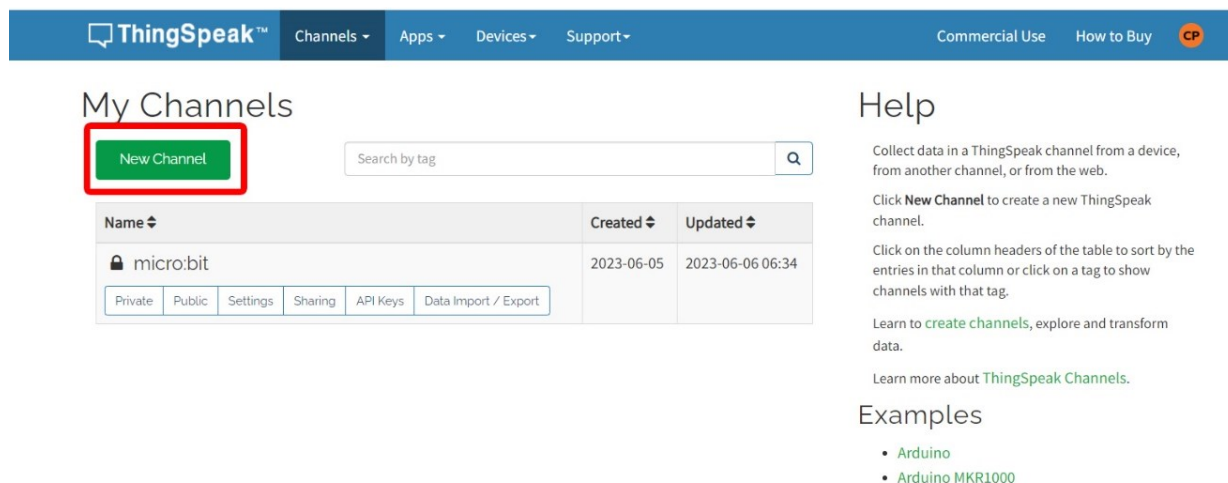


Figura 3: O botão New Channel (Novo canal), situado no menu Channels (Canais)

O passo seguinte consiste em criar um novo canal, inserindo o nome (1.) do canal (por exemplo, microbit) e os parâmetros que pretende monitorizar nos campos fornecidos (2.) (Figura 4). Por exemplo, na Figura 3, o utilizador inseriu Temperatura no Campo 1 e Luz no Campo 2. É possível ativar mais campos clicando na caixa de verificação (3.), ao lado de um campo. Opcionalmente, pode escrever uma descrição para o ajudar a lembrar-se dos parâmetros que o seu canal está a monitorizar.

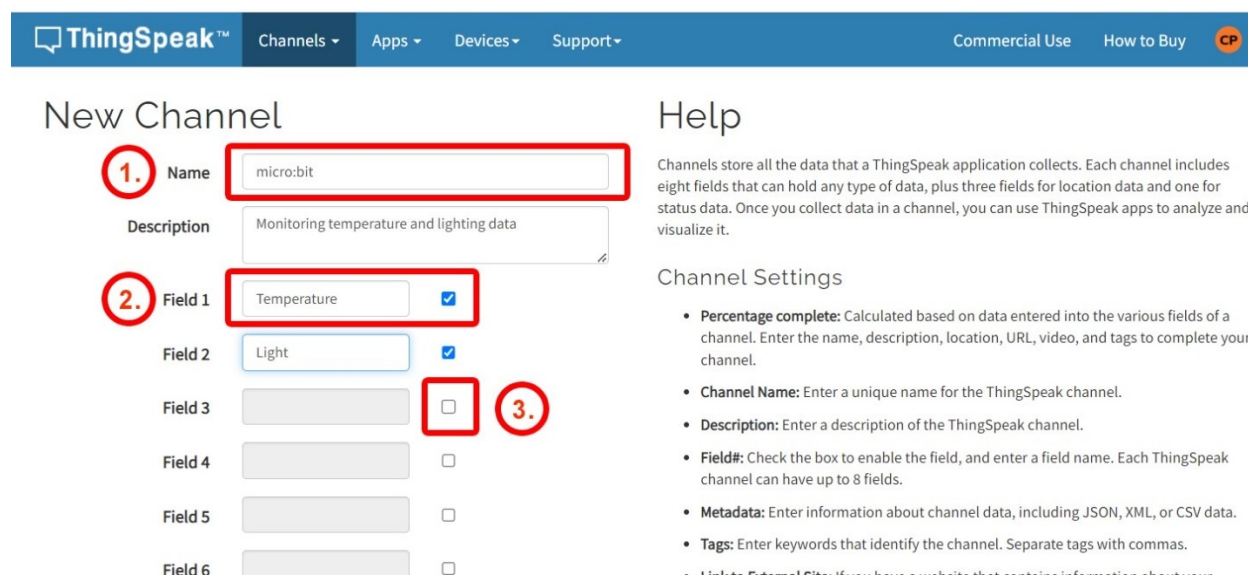
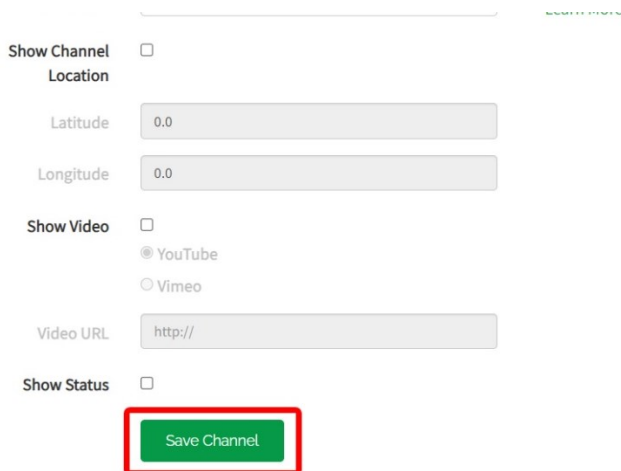


Figura 4: Criar um novo canal

Depois de ter declarado todos os parâmetros que pretende monitorizar, desloque-se para baixo na página Novo canal e clique no botão Guardar canal (Figura 5), para guardar todas as informações inseridas.

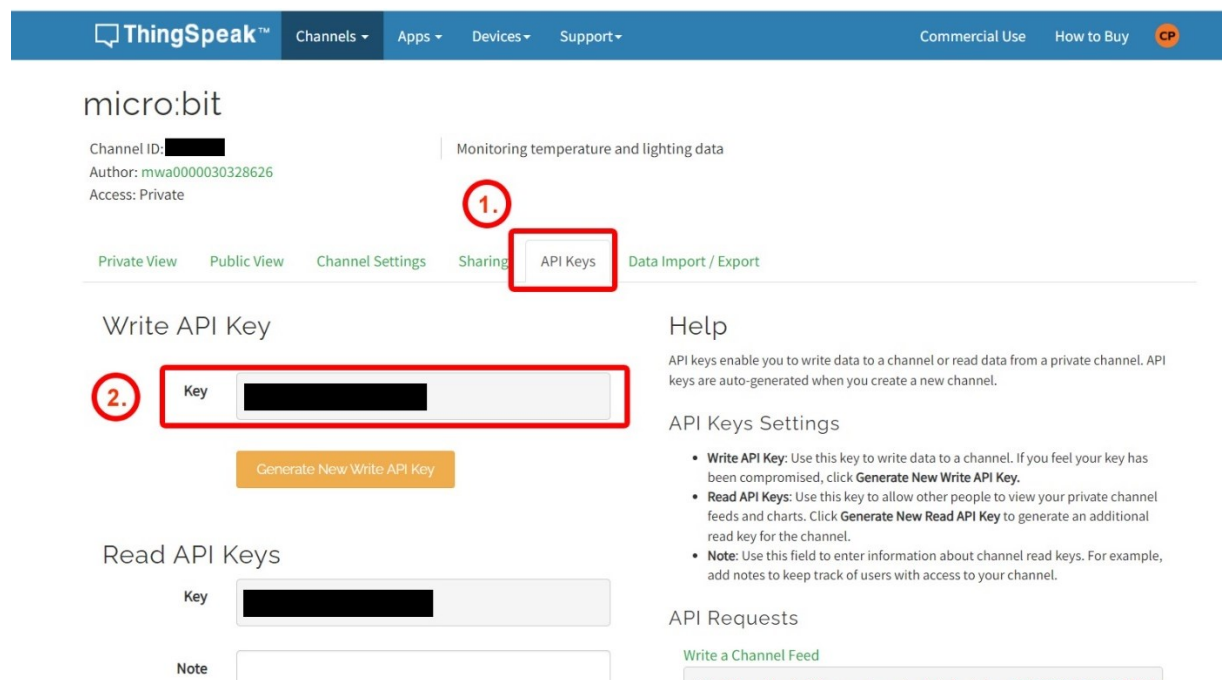


Form fields and buttons:

- Show Channel Location: Latitude (0.0), Longitude (0.0)
- Show Video: YouTube (selected), Vimeo
- Video URL: http://
- Show Status: ☐
- Save Channel** (highlighted)

Figura 5: Percorrer a página Novo canal para clicar no botão Guardar canal

Depois de premir Save Channel (Guardar canal), será automaticamente redireccionado para o seu New Channel (Novo canal). Clique no menu Chaves de API (1.) e escreva a chave (2.) que aparece no campo Escrever chave de API (Figura 6).



Page structure and elements:

- Header: ThingSpeak™, Channels, Apps, Devices, Support, Commercial Use, How to Buy, CP
- Channel Info: micro:bit, Channel ID: [redacted], Author: mwa000030328626, Access: Private
- Monitoring: temperature and lighting data
- Navigation: Private View, Public View, Channel Settings, **API Keys (1.)**, Data Import / Export
- Write API Key: **Key (2.)** [redacted], Generate New Write API Key
- Read API Keys: Key [redacted], Note [redacted]
- Help: API keys enable you to write data to a channel or read data from a private channel. API keys are auto-generated when you create a new channel.
- API Keys Settings:
  - Write API Key:** Use this key to write data to a channel. If you feel your key has been compromised, click **Generate New Write API Key**.
  - Read API Keys:** Use this key to allow other people to view your private channel feeds and charts. Click **Generate New Read API Key** to generate an additional read key for the channel.
  - Note:** Use this field to enter information about channel read keys. For example, add notes to keep track of users with access to your channel.
- API Requests: Write a Channel Feed

Figura 6: Encontrar a chave da API

Esta chave será então inserida num dos blocos de comando que utilizará no ambiente de programação Makecode.

Agora que registou o seu canal no site ThingSpeak, pode prosseguir com a fase de cablagem e programação do módulo Wi-Fi ESP8266.

### 3.4.3.1 Cablagem

A Figura 7 apresenta a forma de ligar o módulo Wi-Fi ESP8266 ao microcontrolador BBC micro:bit. O módulo ESP8266 tem 8 pinos diferentes. É necessário ligar 5 deles à placa micro:bit. Em particular, é necessário ligar os pinos 3V3 e EN (1) à alimentação (3V), e o pino GND (4) ao pino GND do micro:bit. Finalmente, ligar o pino TX (2) e o pino RX (3) aos pinos P1 e P0, respetivamente. Quando o circuito estiver completo, ligue o microbit ao seu computador utilizando um cabo USB.

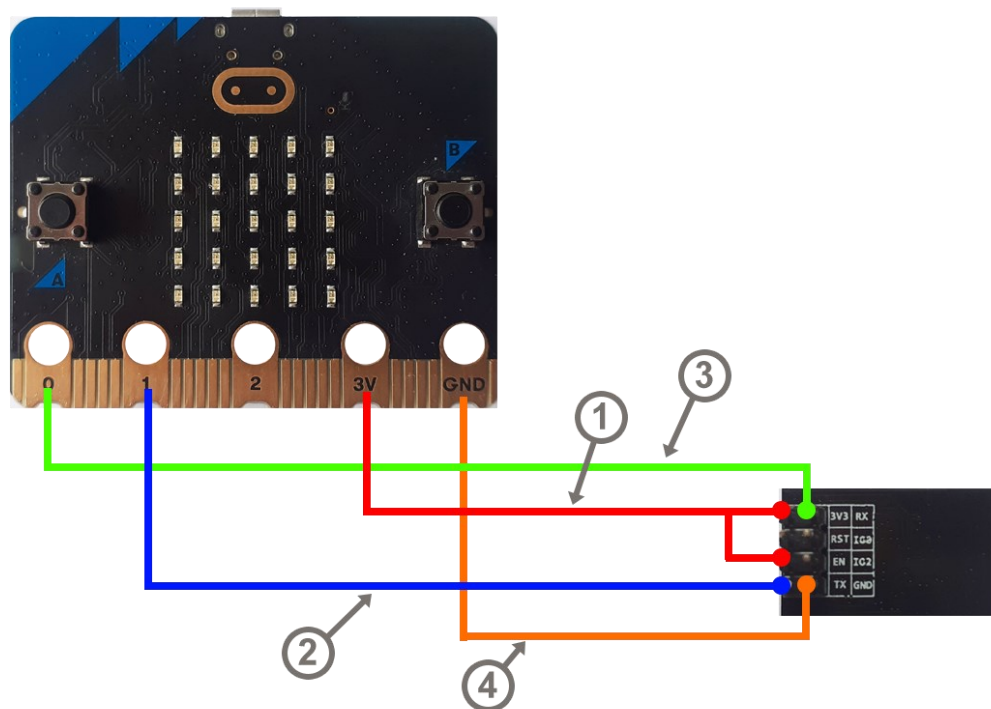


Figura 7: Ligação do módulo WiFi ESP8266 ao microcontrolador BBC micro:bit

### 3.4.3.2 Código

O passo seguinte é abrir o ambiente de programação baseado em blocos do Microsoft Makecode (<https://makecode.microbit.org/#>) e criar um Novo Projeto. Na página inicial do Makecode, clique no separador New Project (1.) e, no menu instantâneo, introduza um nome para o seu projeto (por exemplo, ThingSpeak\_WarmUp). Em seguida, clique no botão Criar (2.) (Figura 8).

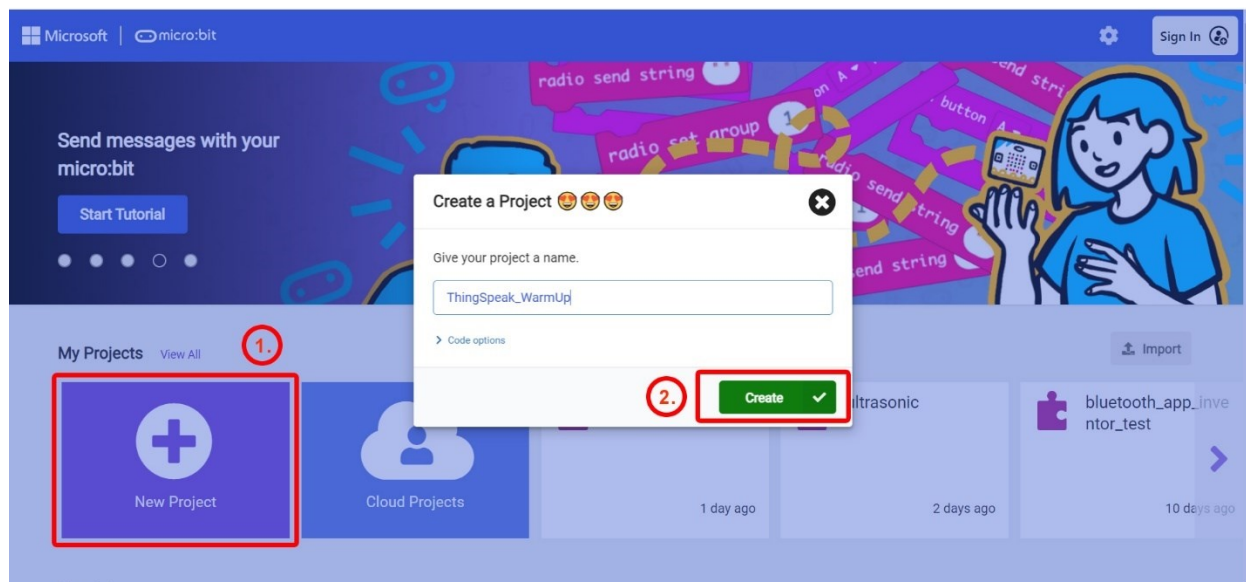


Figura 8: Criar um novo projeto no ambiente de programação MakeCode

Será automaticamente redirecionado para a área onde pode montar o seu código. Para incluir os blocos de programação do módulo Wi-Fi ESP8266, clique no menu +Extensões (Figura 9).

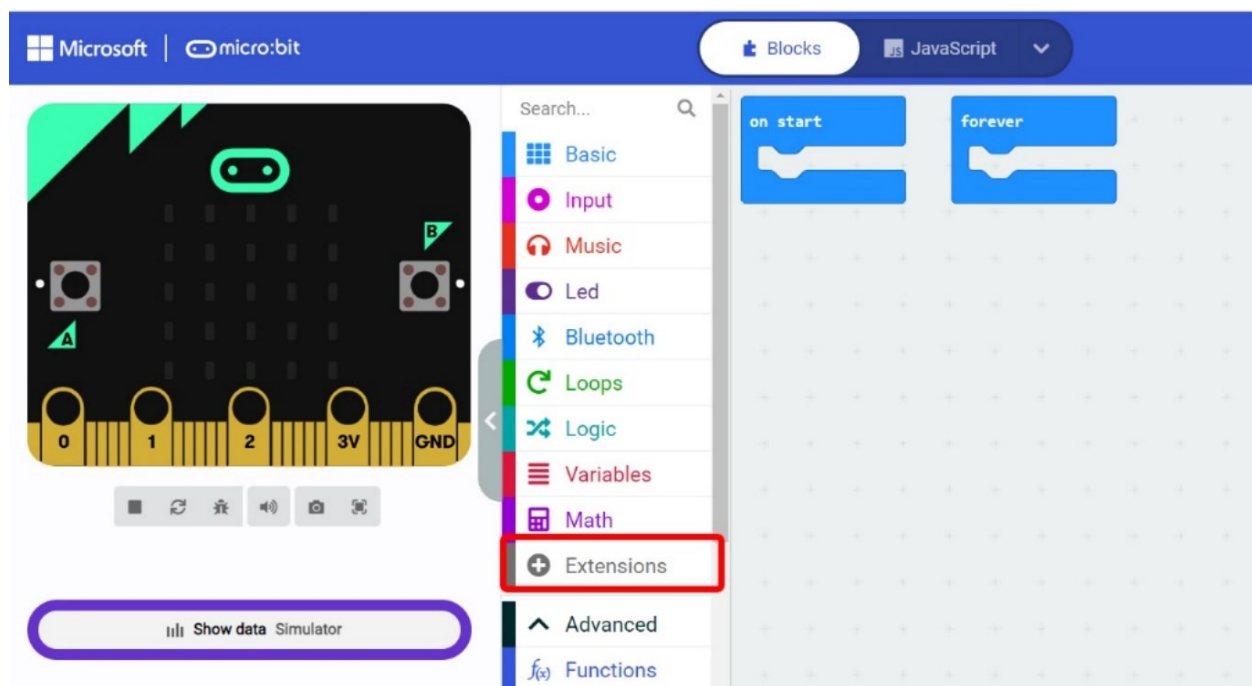


Figura 9: O menu +Extensões

Escreva "ThingSpeak" no campo "Search or enter project URL" (1.) e selecione a extensão correspondente (2.) (Figura 10).



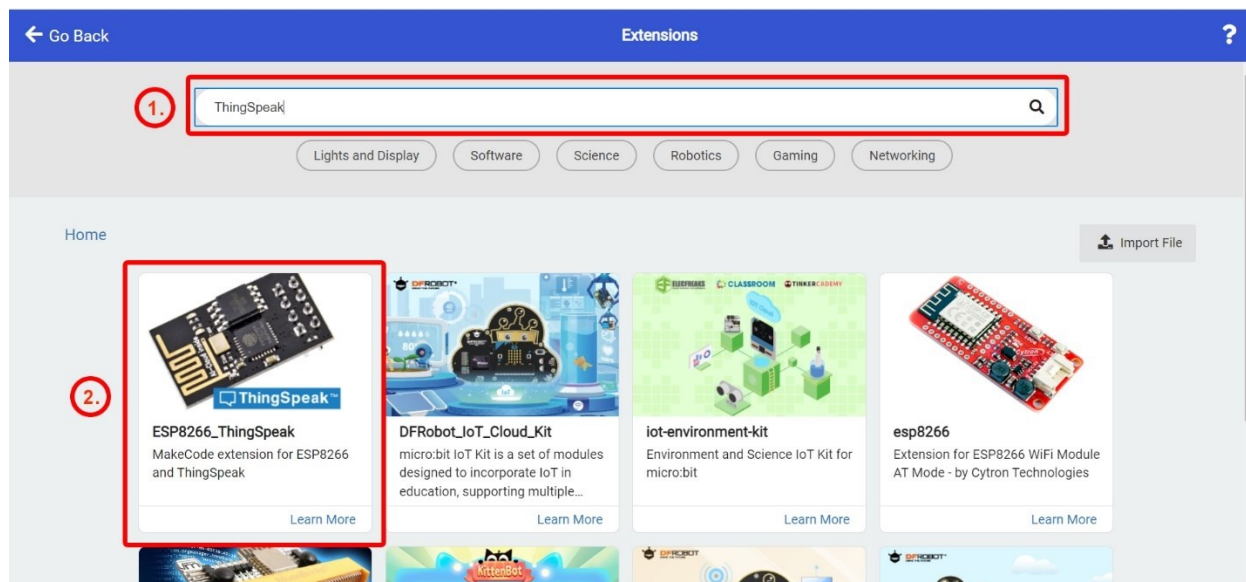


Figura 10: Encontrar e selecionar a extensão ESP8266\_ThingSpeak

Aparecerá um novo menu de blocos (ou seja, ESP8266 ThingSpeak) no menu de comandos (Figura 11).

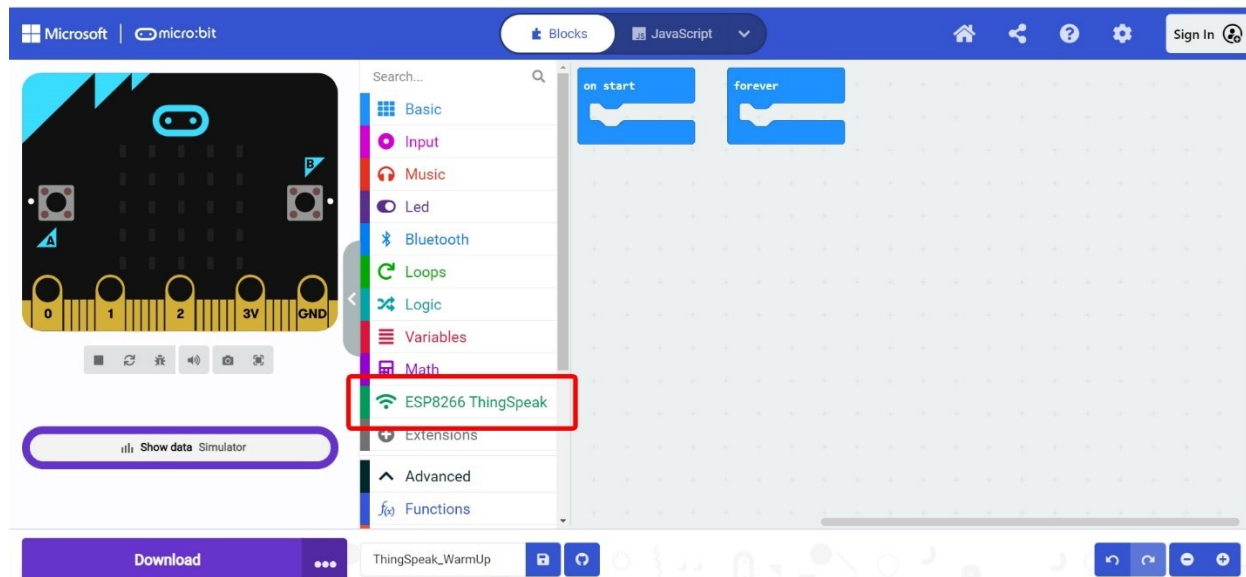


Figura 11: O novo menu com um bloco de comandos para programar o módulo WiFi ESP8266

Em primeiro lugar, é necessário inicializar o módulo Wi-Fi, declarando os pinos a que está ligado e inserindo informações do router Wi-Fi a que se vai ligar. Para isso, clique no menu ThingSpeak do ESP8266 e selecione o comando **initialize ESP8266** block no menu flutuante (Figura 12).

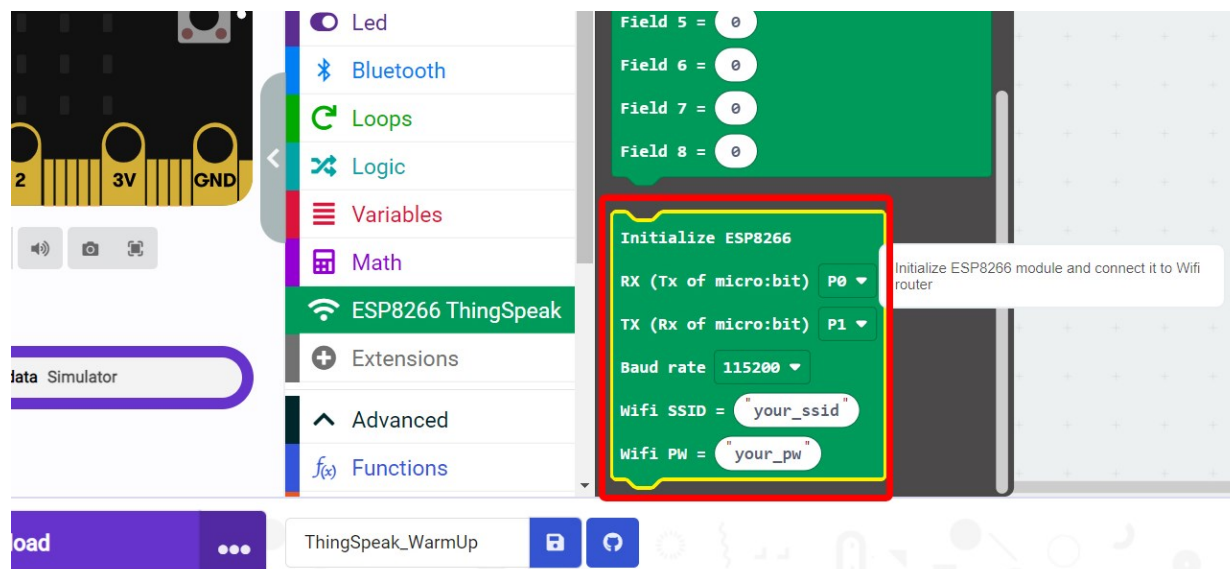


Figura 12: Encontrar e selecionar o comando do bloco Initialize ESP8266

Arraste e encaixe este comando no bloco "on start". Nos campos RX e TX (1.), indique as portas às quais ligou os pinos correspondentes do módulo (nomeadamente P0 e P1) (Figura 13). Em seguida, nos campos wifi SSID e wifiPW (2.), introduza o nome do seu router Wi-Fi e a respetiva palavra-passe. Isto permite ligar o módulo Wi-Fi ao seu router Wi-Fi.

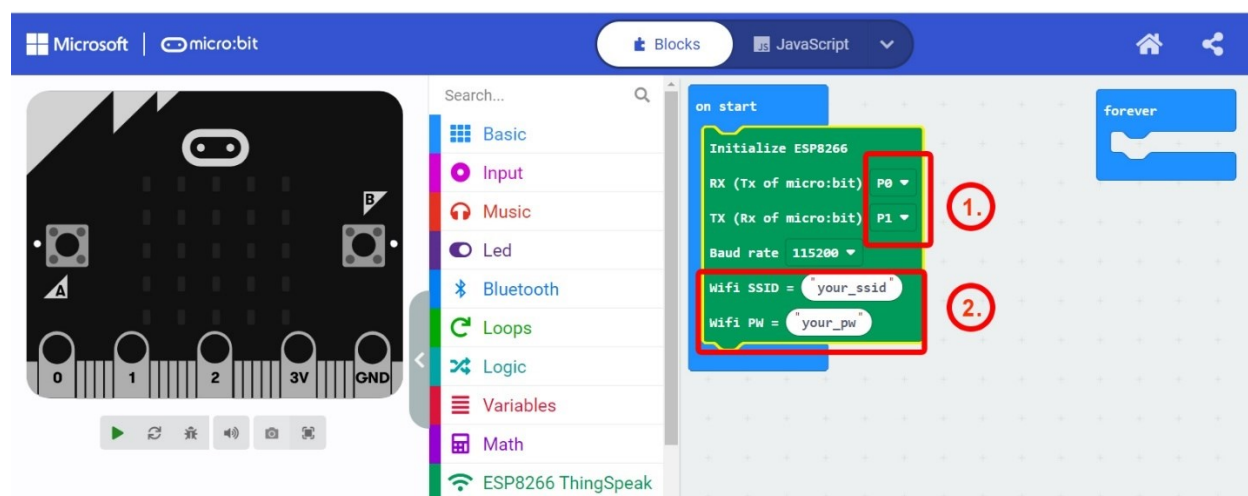


Figura 13: Declarar os pinos aos quais o ESP8266 está ligado e inserir informações sobre a ligação wifi

O próximo passo é permitir que o módulo Wi-Fi carregue os dados que recebe do microbit para a plataforma ThingSpeak. Para isso, arraste e encaixe o comando do bloco "upload data to ThingSpeak" do menu ESP8266 TingSpeak, para o bloco "forever". No campo Write API key (1.) (Figura 14), escreva a chave Write API que recebeu depois de criar o seu canal na plataforma ThingSpeak (Figura 6). Em seguida, para declarar os dados que serão monitorizados pela plataforma ThingSpeak (ou seja, a temperatura e a luminosidade), vá ao menu "Input" e arraste e largue os comandos de entrada "temperature" (3.) e "light level" (2.) para o campo 1 e o campo 2, respetivamente (Figura 14).



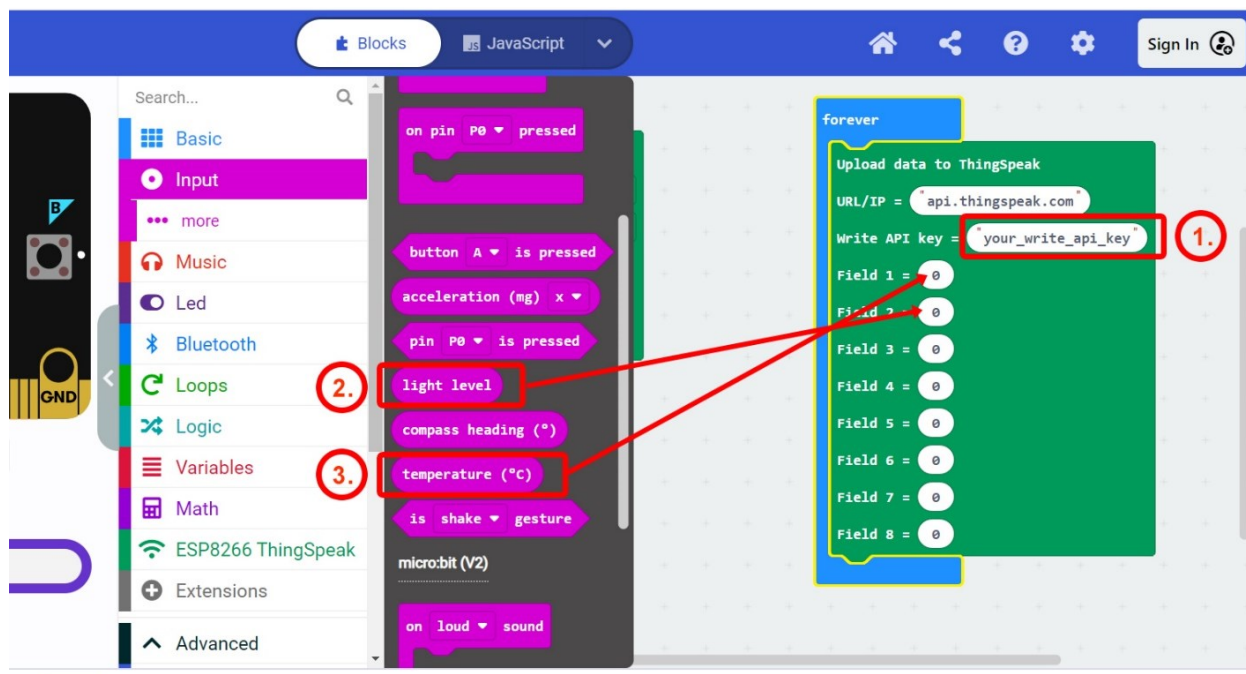


Figura 14: Inserir a chave API e adicionar os parâmetros que serão monitorizados

Depois disso, pode descarregar o script para o seu micro:bit clicando no botão de descarregamento.

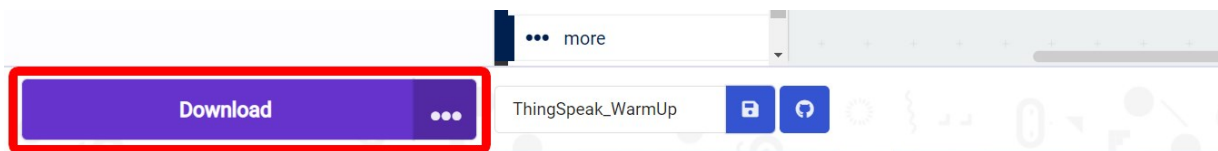


Figura 15: O botão Descarregar

**Nota:** Tenha em atenção que poderá ser necessário adicionar um comando de pausa de 15 segundos dentro do comando loop forever, para permitir uma transferência suave de dados para a plataforma ThingSpeak.

Opcionalmente, pode adicionar o script mostrado na Figura 16 ao seu código para mostrar alguns dos dados recebidos no ecrã LED do seu microbit. Estes comandos (i.e., "serial on data received..." "serial read string") estão localizados no menu de comandos **Serial**.

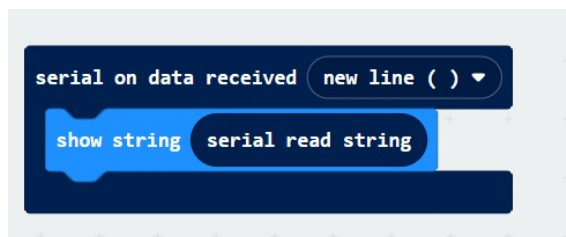


Figura 16: Visualização dos dados recebidos no ecrã LED do microbit

Depois de descarregar o código, aceda à plataforma ThingSpeak e utilize o separador Private View para ver como os dados recebidos pelo microbit se alteram ao longo do tempo (Figura 17).

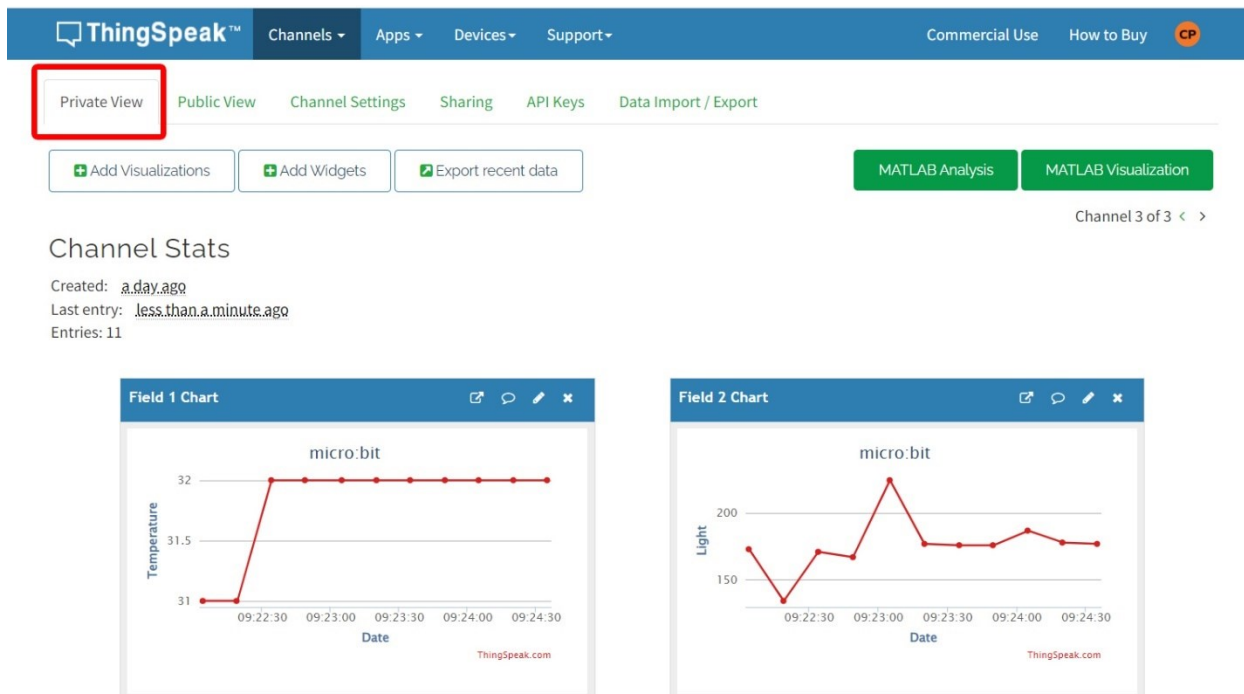


Figura 17: Verificação da evolução dos dados recebidos ao longo do tempo

### 3.4.4 Experiência 1

Repita o processo acima referido com os seus alunos. Antes disso, crie algumas contas na plataforma ThingSpeak e partilhe-as com os seus alunos. Divida-os em equipas de 2 ou 3 elementos e dê a cada equipa uma placa micro:bit e um módulo Wi-Fi ESP8266. Incentive os seus alunos a olhar para o módulo Wi-Fi para explorar os pinos incorporados e, em seguida, peça a cada equipa para ligar o módulo à placa micro:bit. Para facilitar este processo, pode dar-lhes o mapa do circuito contido no ficheiro "Circuit\_card\_Activity1.pdf".

Passa por todas as equipas para te certificares de que todas criaram o circuito com sucesso. Em seguida, prossiga com o processo de programação. Dê a cada equipa uma das contas que criou na plataforma ThingSpeak e peça-lhes para iniciarem sessão. Incentive-as a olhar para os diferentes separadores (ou seja, Vista Privada, Chaves API, etc.). Depois, dê a cada equipa uma cópia do ficheiro "Half\_baked\_Activity1.pdf" e peça-lhes que criem o guião para programar o módulo Wi-Fi para monitorizar a temperatura e o nível de luz registados pelo micro:bit. Peça-lhes que descarreguem o guião para o seu micro:bit e aconselhe-os a observar os dados que estão a ser monitorizados na plataforma ThingSpeak. Incentive-os a usar uma lanterna e/ou a aproximar o circuito de superfícies mais frias ou mais quentes para observar como os dados monitorizados mudam dinamicamente.

Em seguida, discuta com eles algumas das seguintes questões:

- Que informações estão a ser extraídas?
- Esta informação afeta o desempenho do carro robótico?

- Poderá esta informação conduzir a algumas decisões baseadas em dados para otimizar o desempenho do automóvel robótico?
- Pensa noutros parâmetros que possam ser registados. Que tipo de sensores é necessário ligar ao veículo robótico para registar estes parâmetros? Todos estes parâmetros podem conduzir a decisões significativas baseadas em dados para o desempenho do veículo robótico?
- Pense nos serviços de IA que poderiam beneficiar desses dados. Como podem estes serviços utilizar estes parâmetros para extrair a informação subjacente a um conjunto de dados, otimizando assim o desempenho do automóvel robótico?

Através desta atividade e do debate que se segue, os alunos compreenderão:

- Como os sensores podem ser utilizados não só para detetar um ambiente, mas também para tomar algumas decisões baseadas em dados.
- Que a perceção de um artefacto robótico se baseia em dados recolhidos
- A importância de monitorizar e avaliar os dados para tomar decisões ótimas para o desempenho de um veículo autónomo
- Os dados monitorizados podem ser utilizados pelos serviços de IA para otimizar o seu desempenho.

## 3.5 Atividade 2: Introduzir a ideia de Representação e Raciocínio

### 3.5.1 Descrição

Esta atividade tem como objetivo apresentar aos alunos a Grande Ideia do 2<sup>nd</sup>, nomeadamente a ideia de Representação e Raciocínio, e familiarizá-los com a forma como um artefacto robótico pode "aprender" com os dados. Durante esta atividade, os alunos devem ter em mente que a IA e os agentes inteligentes podem "pensar": **a) construindo** representações do mundo sob a forma de estruturas de dados, e **b)** utilizando algoritmos para os ajudar a dar sentido a estas estruturas de dados (raciocínio). Por conseguinte, os alunos aprenderão como as estruturas de dados podem ser representadas e como os algoritmos podem ser utilizados para extrair informações específicas das representações. Para o efeito, serão incentivados a criar algumas árvores de decisão. Através deste processo, compreenderão os critérios para seleccionar um algoritmo que conduza à melhor solução possível.

Em particular, os alunos aprenderão a programar uma aplicação que permite ao utilizador navegar no carro robótico utilizando comandos de voz. Para o efeito, utilizarão o serviço de reconhecimento de voz da IA e serão introduzidos no ambiente do App Inventor. Antes de desenvolverem a aplicação, vão criar uma árvore de decisão para representar a decisão que o carro robótico deve tomar quando é recebido um comando de voz. Em seguida, decidirão qual o caminho lógico a seguir ao programar a sua aplicação.

### 3.5.2 Criar uma árvore de decisão

Para começar a criar uma árvore de decisão, é necessário pensar na forma como o serviço de IA será implementado e nas possíveis ações que podem ser ativadas. O objetivo principal é criar uma aplicação com um botão que registre comandos de voz quando premido. Com base nisto, pode iniciar a sua árvore assumindo que o comando recebido contém a palavra "*avançar*". Se assim for, o carro robótico deve receber instruções para avançar. Se não contiver "para a frente", então existem pelo menos dois caminhos lógicos diferentes: **a)** O comando não é reconhecido de todo, pelo que o carro robótico não se moverá (ou continuará a executar o comando anterior), **b)** o comando recebido contém outra palavra-chave direcional, pelo que o carro robótico receberá instruções em conformidade. A Figura 18 mostra um exemplo indicativo de uma árvore de decisão deste tipo. A árvore não está completa. Podem ser acrescentados mais ramos com base nos diferentes comandos recebidos.

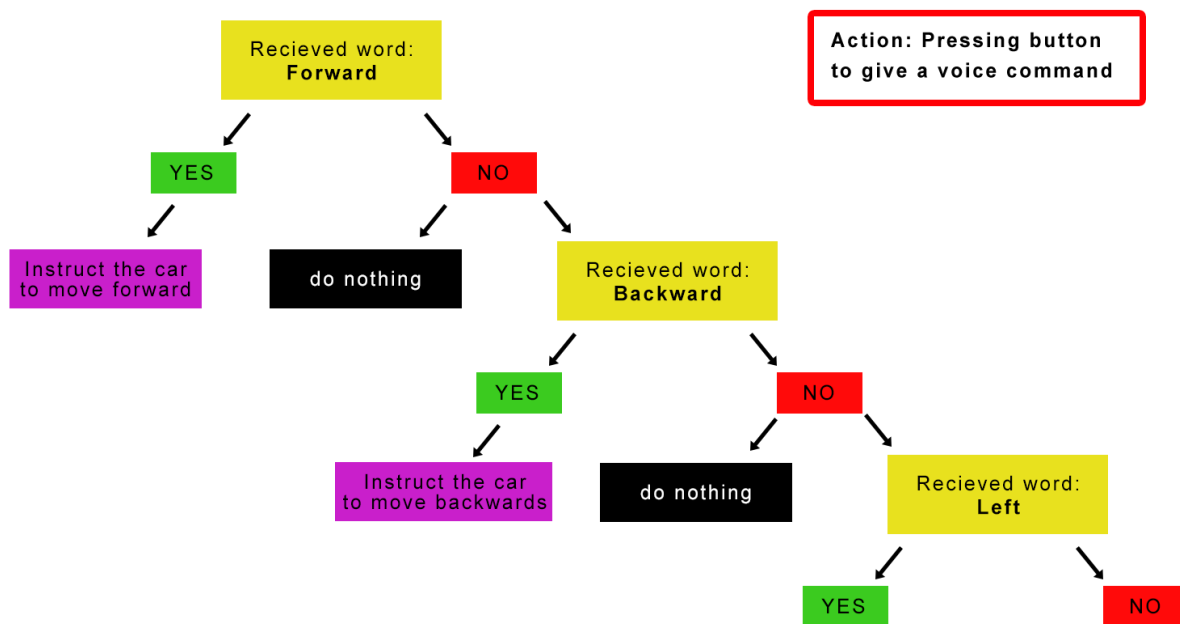


Figura 18: Uma árvore de decisão indicativa que descreve as ações possíveis que podem ser ativadas quando o botão "Premir para falar" é premido

Com base na árvore de decisão, os alunos podem decidir que caminhos lógicos devem ser utilizados na programação. A árvore de decisão apresentada na Figura 18 sugere que deve ser utilizada uma série de comandos "se...então" para programar a aplicação com êxito.

### 3.5.3 Conceção e programação da aplicação

Depois de analisar as possíveis ações que podem ser ativadas, o passo seguinte é criar a aplicação que utilizará o serviço de IA de reconhecimento de voz para permitir que o utilizador controle o carro robótico através de comandos de voz. Antes de começar a criar a aplicação, é altamente recomendável que realize com os seus alunos a atividade de aquecimento contida no ficheiro "2.4\_App\_Inventor\_Warm\_Up.pdf", e continue a trabalhar no mesmo projeto ".aia". Em alternativa, pode abrir o ficheiro de projeto "Remote\_control\_Microbit.aia" e trabalhar sobre ele. Nesse caso, recomenda-se vivamente que guarde o projeto com um novo nome.

**Nota:** Se utilizar o ficheiro .aia acima mencionado, em vez de realizar a atividade de aquecimento do App Inventor, não se esqueça de aconselhar os seus alunos a descarregar o guião descrito no ficheiro "T2.4\_Programação\_do\_carro\_robótico.pdf" para o carro robótico.

#### Processo de conceção

Em primeiro lugar, é necessário adicionar um novo botão que ativará o serviço de reconhecimento de fala da IA. Arraste e largue um item Button sob os botões de navegação (1), renomeie-o para btn\_speak (2) e altere o seu texto para "Press to speak" (3) (Figura 19).

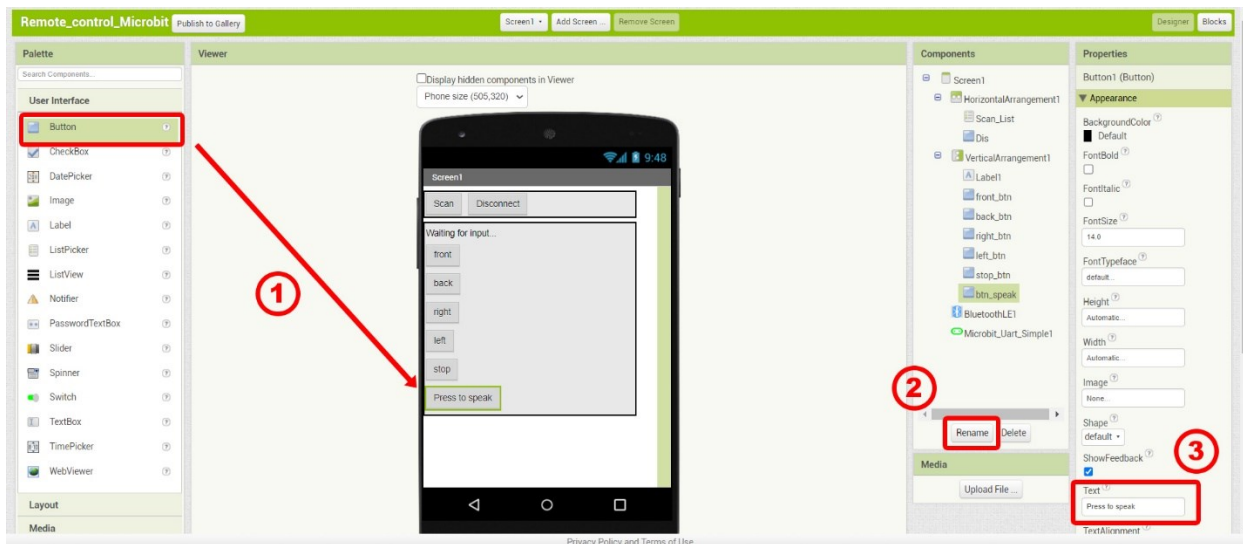


Figura 19: Adicionar um novo botão (1) e mudar o nome para btn\_speak (2), enquanto altera o texto para Premir para falar (3)

Quando o botão "**Premir para falar**" é premido, a aplicação começa a detetar a sua voz. Depois, por defeito, utiliza o serviço de IA de conversão de voz em texto para converter os comandos de voz recebidos em texto.

Para utilizar este serviço, é necessário adicionar o componente "SpeechRecognizer". Localize-o no separador "Media" e arraste-o e largue-o no ecrã (Figura 20).

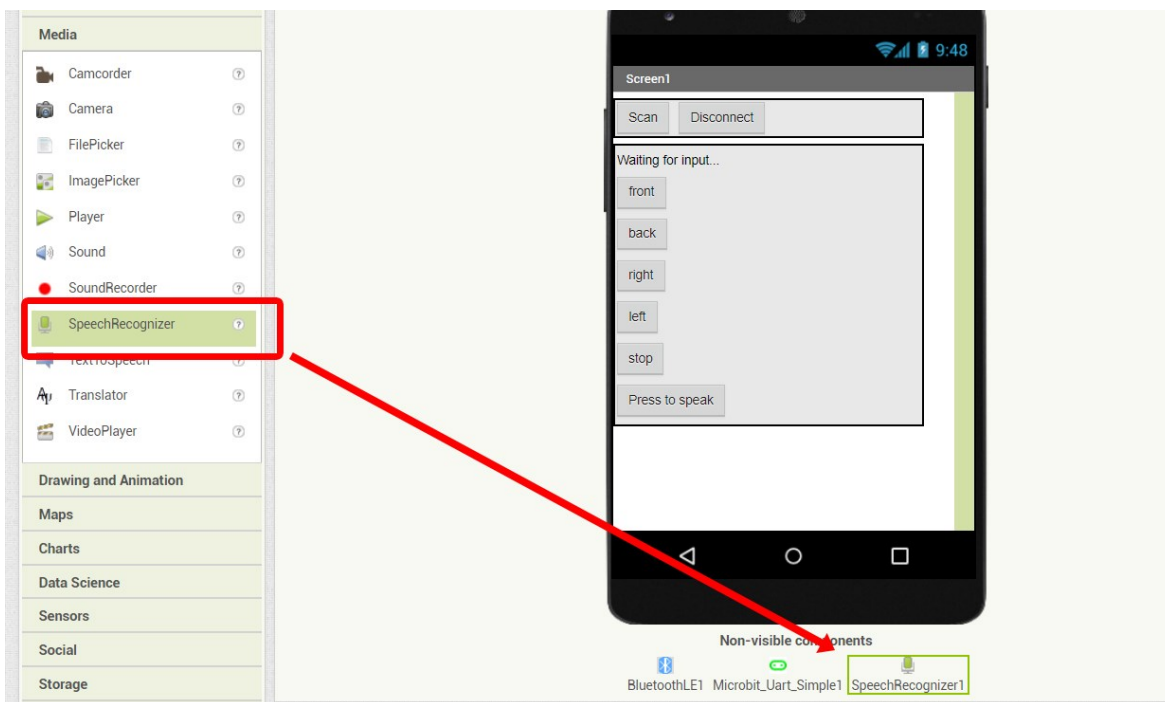


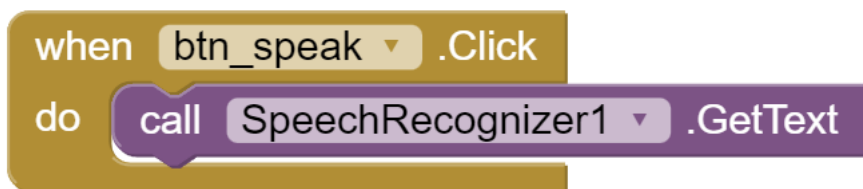
Figura 20: Arrastar e largar o componente SpeechRecognizer na área de desenho

**Nota:** Tal como os componentes BluetoothLE e Microbit\_Uart\_Simple, o SpeechRecognizer é um componente não visível e, por isso, não aparece no ecrã, mas na área abaixo da pré-visualização do ecrã.

### Processo de programação

Esta solução de programação dará instruções à sua aplicação para reconhecer um comando de voz, sempre que o botão "Press to Speak" for premido.

Vá ao menu Block e seleccione o componente "btn\_speak". No menu flutuante, seleccione o manipulador de eventos "**when btn\_speak.Click do**" e arraste-o para a área de montagem do script. Em seguida, clique no componente Reconhecedor de fala e, no menu flutuante, seleccione o comando "**call SpeechRecognizer1.GetText**" e coloque-o dentro do evento manipulador.



Este bloco programa o botão btn\_speak para "**obter o texto**" que o SpeechRecognizer detecta, sempre que o botão "Premir para falar" é premido. Simultaneamente, o serviço de IA de voz para texto é ativado e a voz do utilizador é convertida em texto, permitindo que o carro robótico execute o movimento instruído (ou seja, avançar se o comando de voz recebido for "frente", etc.)

**Nota:** o microfone do dispositivo inteligente do utilizador é utilizado para a captação e gravação de voz.

### Reagir a comandos de voz

Através do método de programação acima referido - e da implementação do serviço de IA de voz para texto - encontramos uma forma de armazenar um comando de voz como texto. Este texto é utilizado como entrada para comandar o carro robótico a efetuar diferentes movimentos. Para tal, os comandos de voz recebidos são filtrados e procurados por palavras-chave específicas que se aplicam ao nosso caso, nomeadamente frente, trás, etc. Desta forma, a aplicação pode corresponder a uma série de comandos de voz diferentes. O quadro seguinte explica sucintamente este conceito:

<b><i>Se a frase recebida pelo Reconhecedor de Voz contiver a palavra:</i></b>	<b><i>Depois, enviar a mensagem:</i></b>	<b><i>E, o automóvel vai:</i></b>
--	--	-----------------------------------

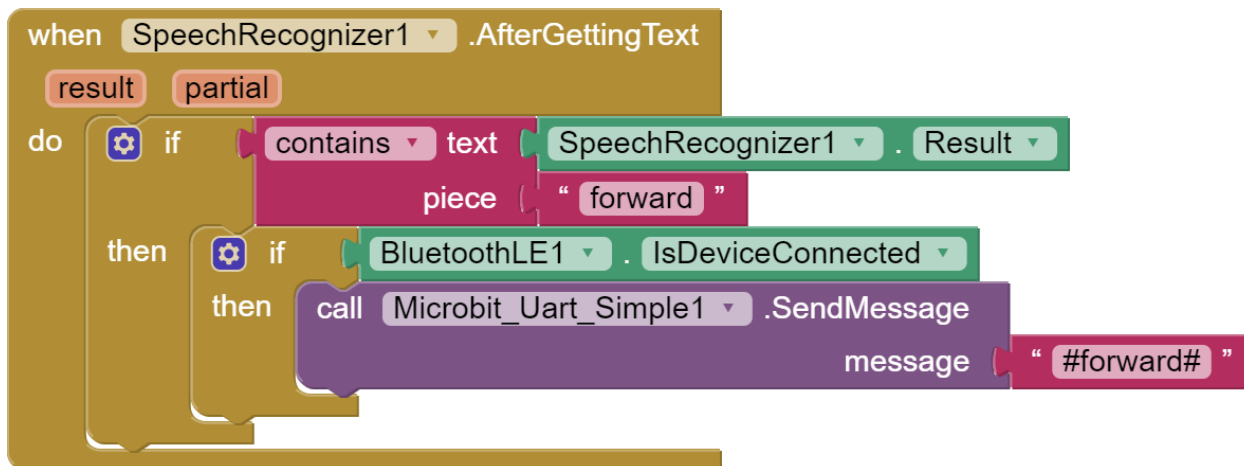


"avancar"	#forward#	avancar
"para trás"	#Para trás	andar para trás
"esquerda"	#esquerda	virar à esquerda
"direito"	#direita#	virar à direita
"parar"	#stop#	paragem

As frases semelhantes às seguintes terão o mesmo efeito no movimento do carro:

**Para a frente:** "Avança", "Avança, por favor", "Quero que avances", etc...

O seguinte guião dá instruções à aplicação para enviar a mensagem #front# ao carro robótico, quando um comando de voz gravado contém a palavra "forward" (avancar). Quando a mensagem #frente# é recebida pelo carro robótico (com base no código já carregado no carro robótico a partir do ambiente Makecode), o carro robótico avança.



A seguir, a funcionalidade de cada bloco de comandos é explicada com mais pormenor:

**When\_SpeechRecogniser.AfterGettingText:** Este bloco recebe o resultado do texto depois de o comando de voz gravado ter sido convertido em texto. Assim, quando o comando de voz do utilizador é convertido em texto pelo SpeechRecogniser, este bloco **executa** os seguintes blocos de comandos:

Se o resultado do `SpeechRecogniser.Result`, ou seja, o texto devolvido como resultado do processo do `SpeechRecogniser` `contains_text()`, for "forward" (ou seja, o texto recebido contém a palavra "forward")

Depois, se o `BluetoothLE1` estiver ligado ao dispositivo desejado, o `Microbit_Uart_Simple1.SendMessage` é chamado e envia a mensagem `#forward#` (texto "avançar") para o carro robótico. Como resultado, o carro robótico está a andar para a frente.

Para o ajustar à árvore de decisão produzida:

"Quando o `SpeechRecogniser` tiver convertido o comando de voz do utilizador em texto, verificar se o resultado do `SpeechRecogniser` (nomeadamente o texto) contém a palavra "frente". Em caso afirmativo, enviar a mensagem "frente" através do `Microbit_Uart_Simple`".

O resto dos comandos de voz podem ser reconhecidos da mesma forma. Por conseguinte, o guião acima deve ser expandido, acrescentando mais quatro condições "se... então", e as peças de texto e mensagens de texto correspondentes, que darão instruções ao nosso carro robótico para executar um movimento específico (ou seja, para trás e a mensagem `#para trás#` para andar para trás, para a direita e a mensagem `#para a direita#` para virar à direita, etc.).

O guião completo é apresentado na imagem seguinte (Figura 21):

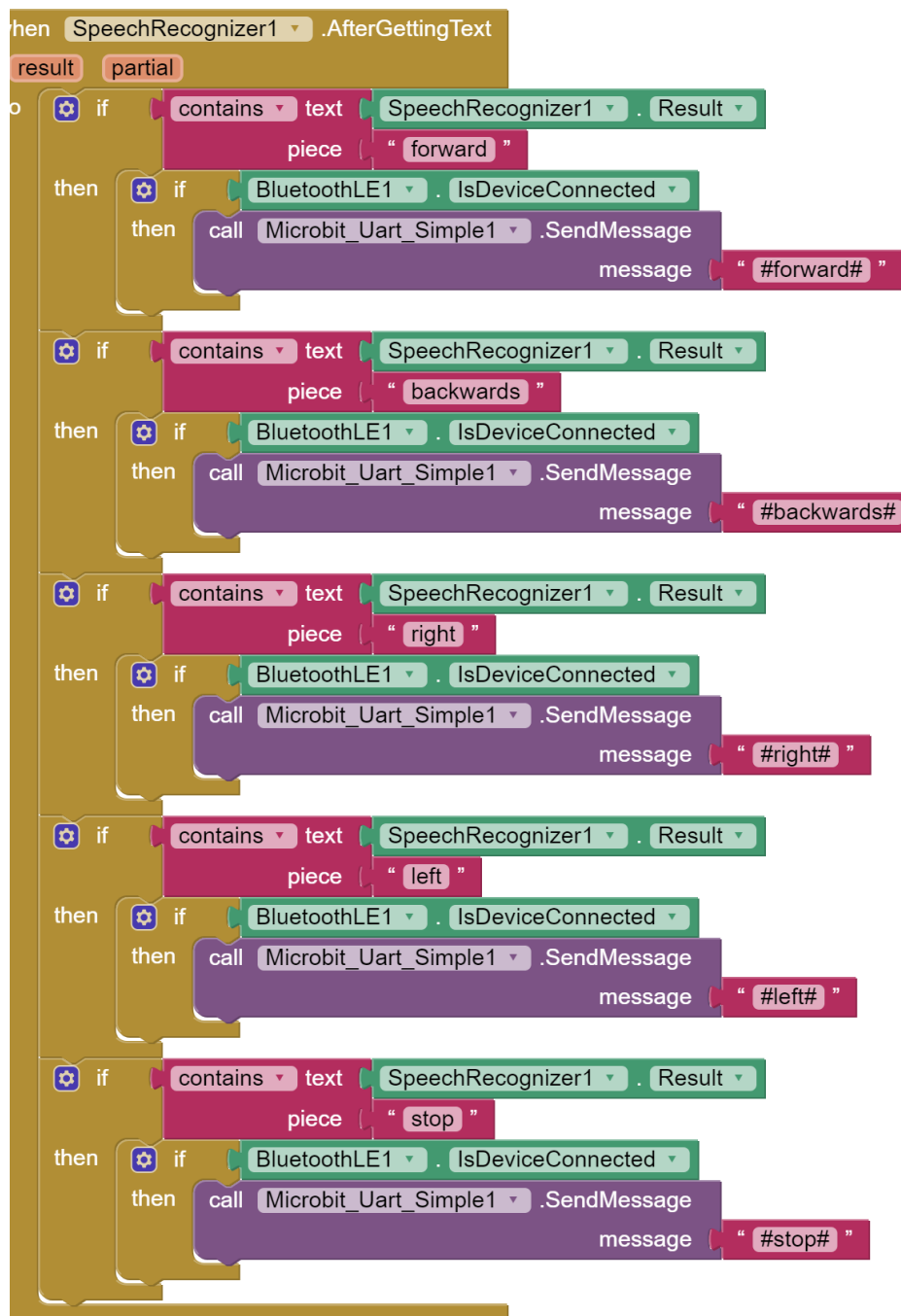


Figura 21: O guião completo para o componente SpeechRecognizer depois de obter o texto

### Limitações - rumo a uma solução de programação ótima

A solução de programação acima tem uma grande desvantagem: exige que o utilizador prima o botão "Prima para falar" sempre que quiser dar um comando de voz. Isto torna todo o processo menos intuitivo em termos de hábitos de condução e pode também reduzir o entusiasmo inicial dos seus alunos, uma vez que podem achar esta solução muito semelhante ao processo de controlo remoto (ou seja, carregar sempre num botão para dar um único comando de voz).

Para ultrapassar esta desvantagem, é também proposta uma solução de programação otimizada, em que o utilizador só tem de premir uma vez o botão "Press to speak" no início do processo e, em seguida, a aplicação tentará ouvir a voz do utilizador a cada 2 segundos.

**Nota:** Esta é uma solução um pouco mais avançada, por isso, dependendo do nível dos seus alunos, pode saltar esta parte e terminar esta atividade de aprendizagem aqui.

### Adicionar um sensor de relógio

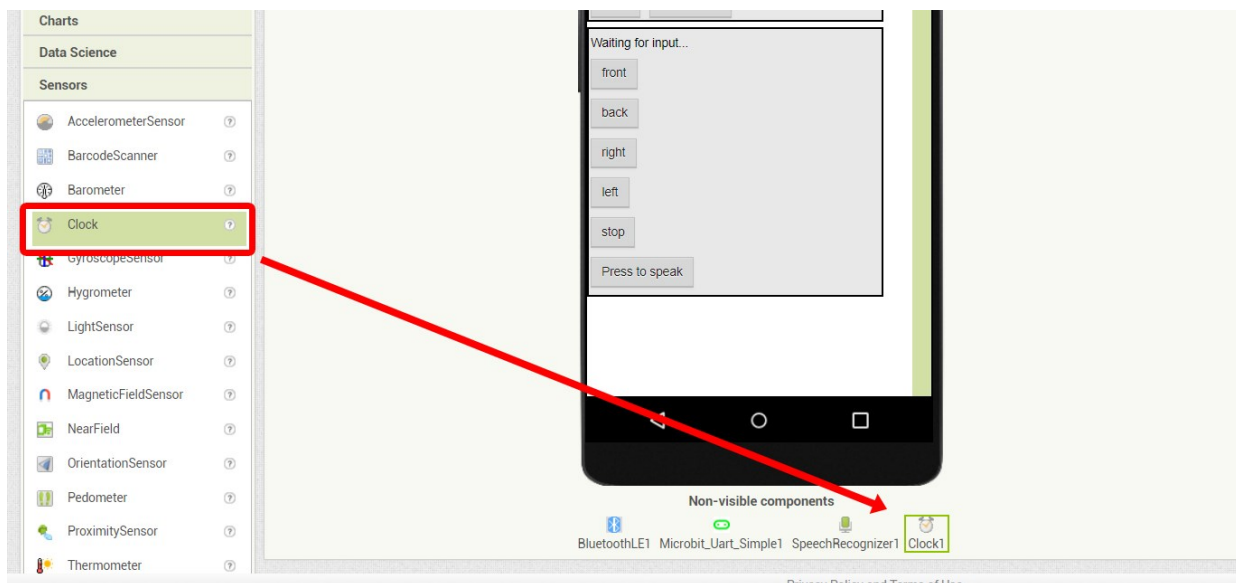
Nesta solução, adicionaremos um sensor de relógio que definirá um temporizador de contagem decrescente para a aplicação. Quando o botão "Premir para falar" é premido, o temporizador é disparado/ignorado e a contagem decrescente começa durante um intervalo de tempo especificado. Durante este intervalo de tempo, a aplicação fica à escuta de um comando de voz recebido. Se a aplicação reconhecer um comando de voz válido, dará instruções ao veículo robótico para efetuar o movimento correspondente. Caso contrário, continuará a procurar até ouvir um comando válido. Quando a contagem decrescente termina, o temporizador é automaticamente reiniciado e começa uma nova contagem decrescente - para o mesmo intervalo de tempo.

Assim, o utilizador só precisa de premir uma vez o botão "Premir para falar" para iniciar todo o processo, tendo em conta que a aplicação só pode receber um comando de voz válido entre intervalos de tempo.

Eis como esta solução pode ser implementada.

Em primeiro lugar, é necessário adicionar o sensor "Clock" à aplicação. No menu do designer, vá ao separador "Sensors" (Sensores) e arraste e largue o sensor "Clock" (Relógio) no ecrã (*Figura 22*).

**Nota:** O Relógio é também um item não visível, pelo que não aparece na pré-visualização do ecrã



*Figura 22: Adicionar o item Relógio à aplicação*

Selecione o componente Relógio, na lista de componentes, e, no menu Propriedades, defina os seguintes parâmetros: a) assinala a caixa sob `TimerAlwaysFires` para permitir que o Temporizador do Relógio seja ativado sempre que a contagem decrescente terminar, b) no campo `TimerInterval` defina a duração do

temporizador, inserindo manualmente um valor numérico em milissegundos (ou seja, 2000 no exemplo) (Figura 23).

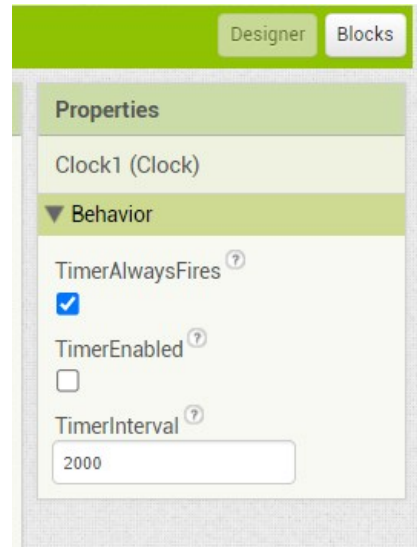


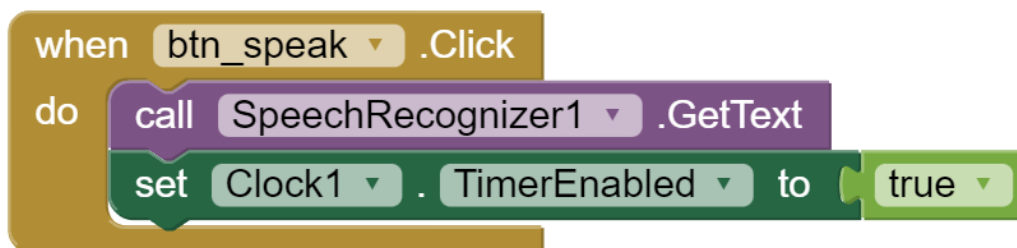
Figura 23: Definição das propriedades do componente relógio

**Nota 1:** a duração de um intervalo é medida em milissegundos. 1 segundo é igual a 1000 milissegundos. Por conseguinte, 2000 ms são iguais a 2 segundos.

**Nota 2:** a caixa sob `TimerEnabled` não está selecionada, porque não queremos que o temporizador seja ativado quando a nossa aplicação é inicializada. Queremos que ele seja ativado quando o botão "Pressione para falar" for pressionado.

O próximo passo é programar este novo componente.

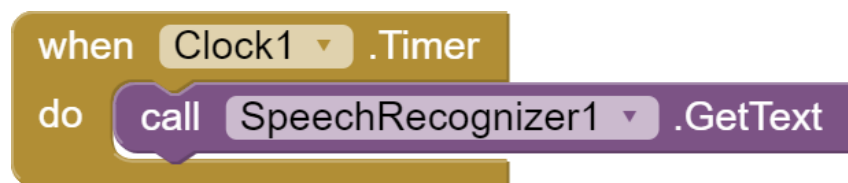
No menu Blocos, adicione o comando "`set Clock1.TimerEnabled to`" dentro do manipulador de eventos "`When btn_speak.Click`" e sob o comando "`call SpeechRecognizer1.GetText`". Em seguida, coloque uma condição "`true`" no lado direito do comando. Através deste comando, o temporizador do relógio é ativado após a ativação inicial do `SpeechRecognizer`.



Agora que disparámos/ignorámos o temporizador pela primeira vez (depois de premir o botão "Falar"), o passo seguinte é reativar o Reconhecimento de voz sempre que a contagem decrescente termina. A duração do temporizador é renovada automaticamente através deste processo.

Para tal, selecione o componente Relógio e, no menu flutuante, selecione o manipulador de eventos **"When Clock1.Timer"**. Em seguida, coloque o comando **"call SpeechRecognizer1.GetText"** dentro do manipulador.

Através deste script, damos instruções à nossa aplicação para **"chamar o SpeechRecogniser para obter o texto"** do que ouve, sempre que o Temporizador do Relógio dispara.



**Explicação de todo o código:** Quando o utilizador clica no botão "falar" pela primeira vez, o SpeechRecogniser é chamado e, em seguida, o temporizador do relógio é ativado. Quando o temporizador é ativado, dispara de 2 em 2 segundos. Sempre que é ativado, o SpeechRecogniser é chamado, aguardando a receção de um input audível/verbal (no nosso caso, um comando de voz).

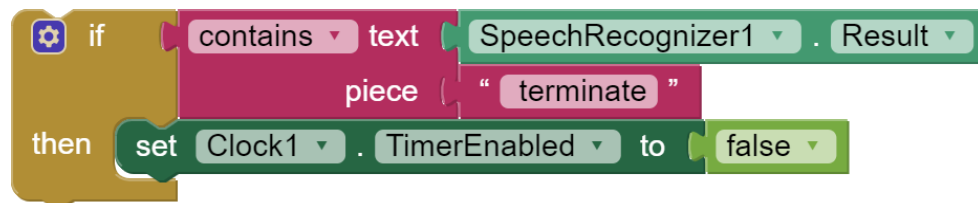
Assim, criou uma aplicação que permite ao utilizador clicar no botão "Premir para falar" apenas uma vez, no início, e depois chama automaticamente o Reconhecimento de voz de 2 em 2 segundos para verificar se há novos comandos de voz.

Adicionar um comando extra para terminar a aplicação

Ao adicionar o componente Relógio, criou uma aplicação mais intuitiva. No entanto, a aplicação vai funcionar para sempre. Isto deve-se ao facto de o Temporizador do Relógio disparar para sempre, e a cada 2 segundos, criando um ciclo infinito, no qual o Reconhecimento de voz está constantemente à procura de comandos de voz válidos.

Para resolver este problema, é necessário adicionar alguns blocos de código extra na aplicação. Especificamente, é necessário adicionar ao script da Figura 20, o seguinte bloco de comandos.

Em particular, acrescente mais uma condição **"if...then"** no final do guião. No interior da parte **if**, coloque um bloco de comandos que permitirá ao **SpeechRecogniser** identificar a **parte do texto "terminar"**. Na parte **"then"**, **coloque** um bloco de comandos que desativará o sensor de relógio, nomeadamente os blocos **"set Clock1.TimerEnabled to false"**.



Através destes blocos de código adicionais, o temporizador do relógio será desativado quando o utilizador disser a palavra terminar. Assim, o ciclo será terminado e o Reconhecedor de Voz deixará de procurar comandos de voz válidos.

**Sugestão:** Dependendo do nível dos alunos, pode incentivá-los a encontrar a sua própria solução para o problema do ciclo infinito. Desta forma, eles podem perceber melhor como funciona o sensor do Relógio e sentir-se mais confiantes em relação a todo o processo.

### 3.5.4 Experiência 2

Esta atividade de 2<sup>nd</sup> é bastante extensa, pelo que é necessário prever um par de horas para lidar com todos os conceitos e os seus aspetos inerentes.

Para introduzir a segunda ideia, para além de incentivar os alunos a conceber árvores de decisão ou fluxogramas para representar possíveis ações ou comportamentos que o carro robótico pode adotar, pode também iniciar um diálogo colocando algumas das seguintes **questões**:

- Que tipo de representações precisa de criar para ilustrar melhor a informação codificada?
- Que caminhos ou operadores lógicos é necessário adotar para ajudar um agente inteligente a tomar a melhor ação possível?
- Qual é a melhor forma de utilizar os resultados da perceção da IA? (por exemplo, com base na reação do carro robótico, pense em casos em que essa tecnologia seria uma mais-valia)

Através desta atividade e do debate que se segue, os alunos serão capazes de compreender:

- Como as árvores de decisão e os fluxogramas podem revelar possíveis caminhos lógicos, conduzindo também a decisões sobre os operadores a utilizar
- Compreender como um agente inteligente pode adotar o melhor comportamento ou desempenho possível
- Compreender como os agentes inteligentes podem percecionar o seu ambiente e agir com base na informação recebida

Para a conceção e criação da aplicação, pode utilizar o ficheiro "Students\_Worksheet\_for\_Activity\_2.pdf" e orientar discretamente os seus alunos ao longo do processo, dando-lhes dicas sempre que necessário. Pode também incentivá-los a experimentar diferentes comandos de voz (e, por conseguinte, **peças de texto**) e a observar os resultados. Para tal, pode aconselhá-los a fazer uma tabela - como a que se segue - na qual registam quais os comandos de voz que são bem sucedidos e quais os que não são para mover o carro robótico numa determinada direção.

Comando de voz	Sucesso	Falha
Avançar	√	
Ir para a frente		√
....		



Para a solução otimizada (com a adição do item Relógio), pode encorajar os seus alunos a definir diferentes durações para o Intervalo do Temporizador e registar as suas observações sobre a funcionalidade da aplicação. Pode também sugerir-lhes que criem uma tabela (como a que se segue) para organizar as suas observações. Além disso, pode incentivar cada equipa a utilizar um intervalo de tempo diferente e a comparar os resultados em demonstrações paralelas.

TemporizadorIntervalo	Observação
500ms	A aplicação está a funcionar muito rapidamente
2000ms	A aplicação está a funcionar corretamente
....	....

## 3.6 Atividade 3: Introduzir a ideia de aprendizagem através do treino de um modelo de reconhecimento de comandos de voz

### 3.6.1 Descrição

Nesta atividade, os alunos serão introduzidos na grande ideia do 3<sup>rd</sup>, nomeadamente a aprendizagem, treinando um modelo para reconhecer comandos de voz específicos. Através desta atividade, compreenderão o papel da Aprendizagem Automática (AM) e dos algoritmos de aprendizagem automática para ajudar os computadores a aprender. Em particular, aprenderão a utilizar a ferramenta ML Personal Audio Classifier para treinar um modelo que classifica comandos de voz de acordo com critérios específicos.

### 3.6.2 Utilizar o classificador de áudio pessoal para treinar um modelo

Na atividade anterior, aprendeu a utilizar o serviço de reconhecimento de voz para gravar comandos de voz e transformá-los em texto, de modo a navegar no carro robótico. Nesta atividade, vai aprender a treinar um modelo para perceber uma série de comandos de voz e classificá-los de acordo com critérios específicos. Para o efeito, utilizará o ambiente de treino do Personal Audio Classifier (PAC) (<https://c1.appinventor.mit.edu/>).

A Figura 24 apresenta o ambiente de treinamento do PAC. Como se pode ver, o modelo está vazio. Não existem categorias, nem etiquetas, nem sons classificados. Por isso, é necessário criar uma série de categorias. Cada uma destas categorias conterá várias amostras de áudio gravadas (por exemplo, diferentes comandos que podem ser identificados como frente, trás, etc.), que serão representadas por uma etiqueta comum (por exemplo, frente, trás, etc.).

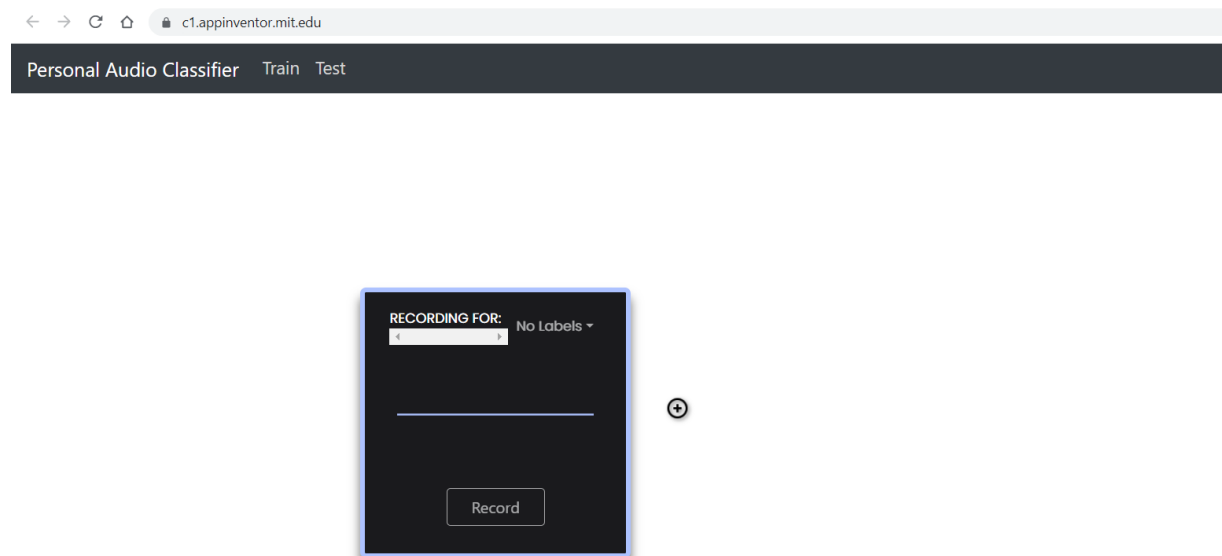


Figura 24: Ambiente de formação do Personal Audio Classifier (PAC)

**Nota:** Certifique-se de que o seu dispositivo (PC, computador portátil, etc.) está equipado com um microfone/altifalante. Caso contrário, não será possível gravar sons.

O primeiro passo para criar uma categoria de sons comuns é criar uma nova etiqueta. Clique no botão +, e no menu flutuante "Criar uma nova etiqueta" que aparece, escreva o nome da categoria que quer criar (i.e., "frente" no exemplo mostrado na *Figura 25a*). Prima a tecla "Enter" e aparecerá uma nova janela com o título da etiqueta que criou (*Figura 25b*). Esta é a primeira categoria de sons que vai preencher com amostras de áudio. Para isso, prima o botão Record (*Figura 25b*) (1) para começar a gravar o áudio recebido pelo seu microfone (para mais informações, consulte a secção "**Conselhos para gravar**"). Certifique-se de que está a gravar para a categoria correta, verificando o nome da etiqueta que aparece ao lado da secção "Recording for" (*Figura 25b*) (2).

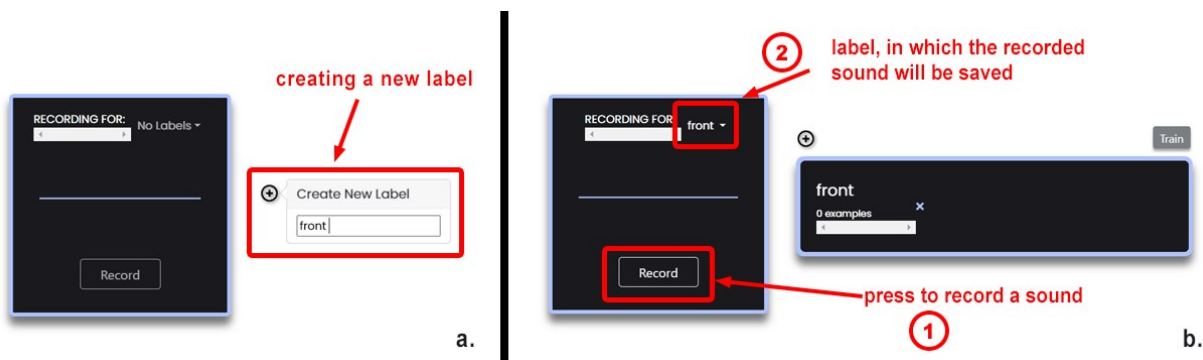


Figura 25a. Criação de uma nova etiqueta; b. Início da gravação de amostras de áudio para preencher a categoria "frente"

Repita o mesmo processo para criar todas as diferentes categorias de comandos de voz que pretende que a aplicação reconheça e que o automóvel robótico execute (por exemplo, frente, trás, direita, etc.).

**Nota:** É importante recordar o nome exato de cada etiqueta/categoria, para que a aplicação possa recuperar essas categorias com sucesso. Incentive os seus alunos a criar uma tabela na qual anotarão o nome de cada etiqueta e uma breve descrição dos sons nela armazenados.

Rótulo/Categoria	Resultado do comando de voz
frente	Mover o carro robótico para a frente
palmas	Mover o carro robótico para trás
esquerda	Fazer o carro robótico virar à esquerda
...	...

### Sugestões no registo (*Figura 26*):

Quando prime o botão Gravar, o processo de gravação é ativado durante cerca de 1 segundo. Durante este tempo, o microfone grava todos os sons que possam ser ouvidos. Devido a este limite de tempo, recomenda-se que grave comandos de voz curtos (por exemplo, "frente" em vez de "para a frente").

Pode verificar o estado da gravação observando a linha azul **(1)** acima do botão Gravar. Se a linha reta se transformar numa forma de onda, o microfone está a funcionar corretamente. Cada som gravado é guardado como um novo ficheiro de áudio dentro da categoria selecionada ("frente" no exemplo). Se o seu microfone tiver gravado um som com sucesso, este será convertido num ficheiro de áudio e aparecerá um ícone de espectrograma colorido **(2)**. Se o seu microfone não captou qualquer som, aparece uma caixa branca, indicando que o ficheiro de áudio guardado está vazio **(3)**. Se quiser remover um ficheiro de áudio que já foi guardado, passe o cursor sobre o ficheiro de áudio correspondente. Uma marca X aparecerá no canto superior direito. Clique nele e o ficheiro audio será apagado.

Para criar um modelo treinado bastante fiável, é necessário incluir pelo menos 5 a 6 exemplos/amostras de ficheiros áudio em cada categoria. O número de ficheiros de áudio já gravados é indicado no campo por baixo do título principal de cada categoria **(4)**. Se quiser apagar uma categoria inteira, prima o botão X **(5)** junto ao campo "exemplos".

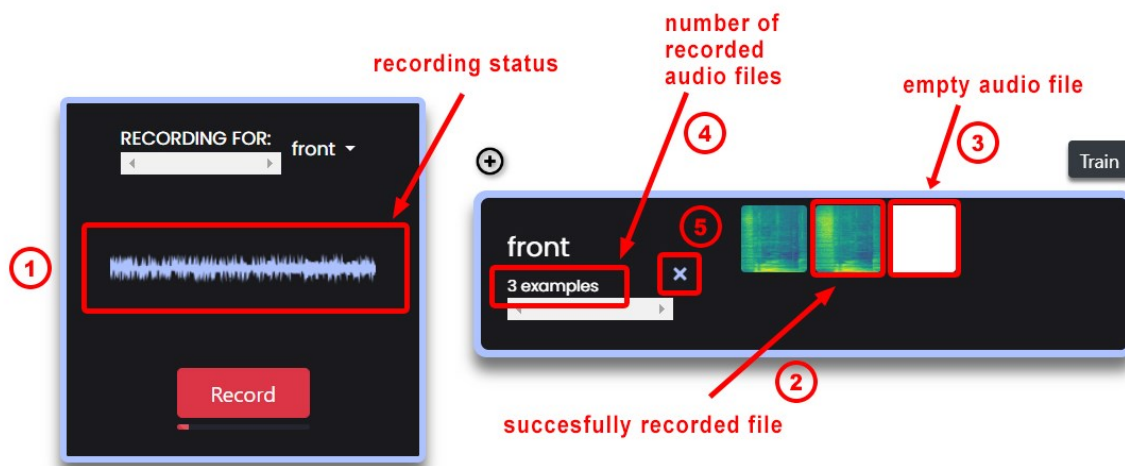


Figura 26: Informações sobre o processo de gravação e os ficheiros áudio gravados

### Treinar, testar e exportar o modelo

Uma vez criadas todas as etiquetas/categorias, com pelo menos 5 a 6 sons gravados em cada categoria, pode proceder ao treino do modelo (Figura 27). Para o fazer, clique no botão Treinar **(1)**. Aparecerá uma nova janela que lhe permitirá ajustar alguns parâmetros. Mantenha os parâmetros predefinidos e clique no botão Treinar modelo **(2)** para iniciar o processo de treino.

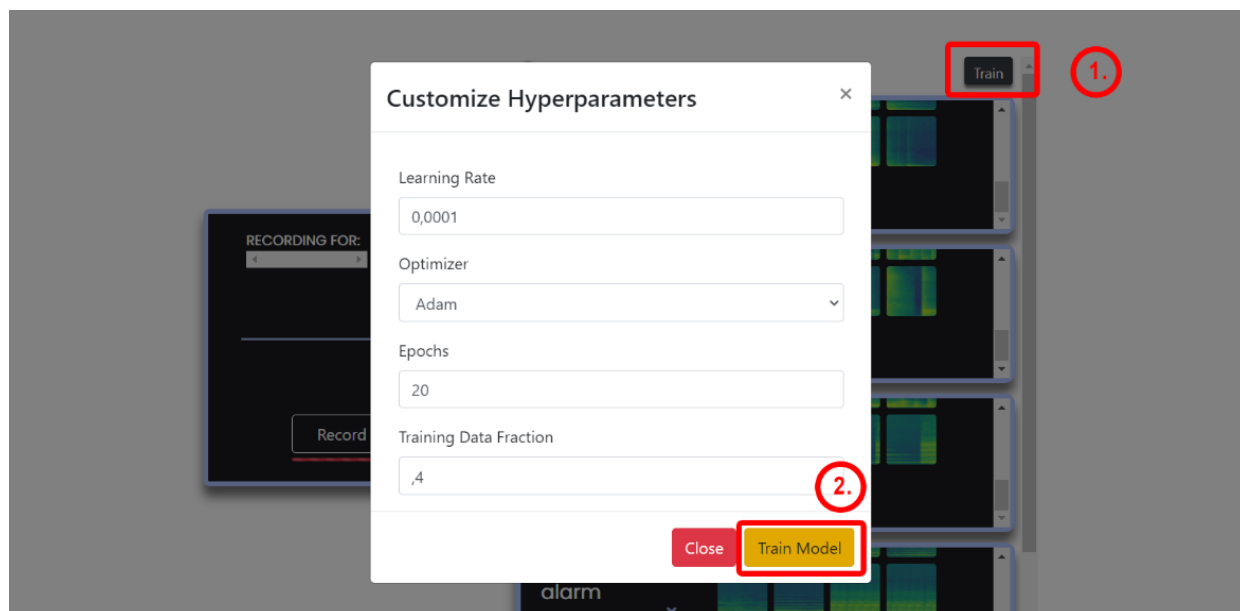


Figura 27: Treinar o modelo

Aguarde alguns minutos até o treino estar concluído. Antes de exportar o modelo treinado, aconselha-se vivamente a testar o modelo treinado criado, clicando no botão Gravar (2), gravando um som (por exemplo, prima gravar enquanto diz "voltar") e ver se o som gravado é reconhecido e classificado na categoria correspondente. Para isso, basta verificar quais as categorias que ficam verdes na janela Classificação (4). O espectrograma do som gravado (3) também aparece, confirmando que o som de entrada foi gravado com sucesso.

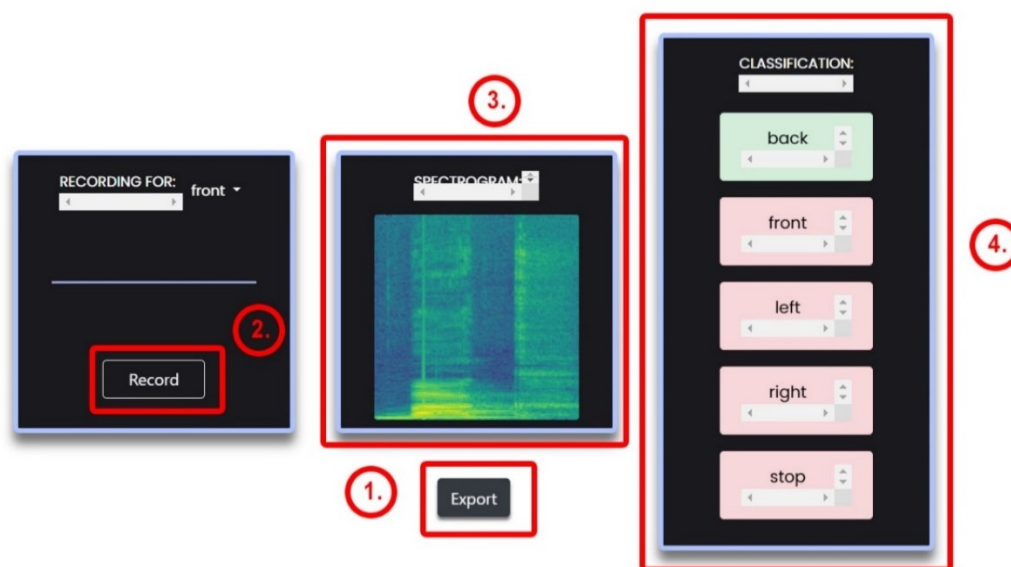


Figura 28: Testar e exportar o modelo treinado

Pode também deslocar-se para baixo na página para ver o nível de confiança da previsão. Por exemplo, na Figura 29, é informado de que o nível de confiança da previsão "voltar" é de 38%.



Figura 29: Nível de confiança da previsão para o registo de entrada

Incentive os seus alunos a efetuar vários testes com o modelo treinado. Se forem encontrados demasiados erros, aconselhe-os a clicar no separador "treinar" (Figura 30) para fazer alterações aos dados registados.

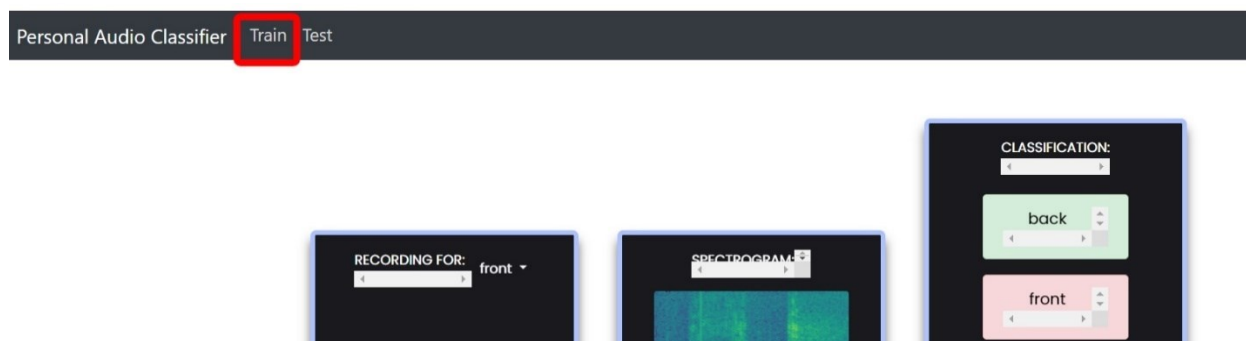


Figura 30: Clique no separador "Comboio" para regressar ao modo de gravação

Se o modelo treinado estiver a funcionar corretamente, clique no botão Exportar **(1)** (Figura 28) para guardar o modelo treinado localmente no seu computador, como um ficheiro .mdl.

**Nota:** o ficheiro é automaticamente designado por *model.mdl* quando é descarregado, mas pode renomear manualmente este ficheiro (após o processo de descarregamento) para algo significativo (por exemplo, *voice\_commands\_model.mdl*, etc.).

**Nota importante:** infelizmente, a versão atual do ambiente do Personal Audio Classifier não permite qualquer modificação no modelo treinado, depois de sair da página. Por isso, deve prever tempo suficiente para completar esta atividade durante uma hora de ensino/aprendizagem.

Na atividade 4<sup>th</sup>, aprenderá a integrar este modelo na aplicação que já concebeu no App Inventor e a utilizar o modelo exportado para classificar os comandos de voz recebidos, de modo a instruir o carro robótico a mover-se em conformidade.

### 3.6.3 Experiência 3

A secção anterior deu algumas dicas sobre como apresentar esta atividade ao seu aluno. A primeira coisa a fazer é explicar-lhes qual é o objetivo desta atividade (ou seja, aprender a utilizar uma ferramenta de ML para treinar um modelo que pode ser utilizado pelo carro robótico numa fase posterior). Pode incentivá-los a pensar em diferentes cenários em que a classificação de áudio poderia ajudar o carro robótico a aprender com o seu ambiente, mas, como ponto de partida, incentive-os a pensar em como podem treinar um modelo para classificar diferentes comandos de voz.

Para tal, e antes de abrir a ferramenta Personal Audio Classifier, incentive-os a escrever os comandos de voz que querem gravar e qual o resultado esperado de cada um desses comandos de voz (por exemplo, o que esperam que o carro robótico faça, se o comando de voz for "correto"). Para facilitar este processo, pode aconselhá-los a criar uma tabela como a que se segue e a anotar as suas ideias.

Comando de voz	Resultado

Depois, peça-lhes que abram a ferramenta Personal Audio Classifier e explique-lhes brevemente como a utilizar. Incentive-os a criar uma série de etiquetas e a gravar algumas amostras de áudio em cada categoria. Algumas dicas que lhes pode dar são:

- criar tantas etiquetas/categorias quantos os movimentos que o carro robótico irá efetuar
- experimentar diferentes sotaques ou tentar dar ênfase a diferentes partes de uma palavra gravada
- anotar o nome exato que dão a cada etiqueta/categoria. Isto é importante para a atividade seguinte (i.e., 4<sup>th</sup> Activity)



Depois de terem criado todas as categorias, peça-lhes para treinarem e testarem o modelo. Ao testar, aconselhe-os a verificar também o nível de confiança de cada previsão. Pode incentivá-los a anotar numa tabela como a seguinte se o comando de voz gravado foi corretamente percebido e classificado e qual o nível de confiança de cada previsão.

Som	Classificado com êxito	Nível de confiança (%)
	SIM / NÃO	
	SIM / NÃO	
	SIM / NÃO	
	SIM / NÃO	

Quando estiverem satisfeitos com o resultado, peça-lhes que exportem o modelo, guardando-o assim localmente no seu computador.

Durante esta atividade, pode discutir com eles os diferentes aspetos deste processo. Para iniciar um diálogo com eles, pode colocar algumas das seguintes **questões**:

- O que é que uma ferramenta de aprendizagem automática pode fazer?
- Que parâmetros devem ser tidos em conta no treino do modelo?
- É importante avaliar um modelo treinado antes de o utilizar numa aplicação de IA?
- Como é que um modelo treinado com preconceitos, que classifica comandos de voz, pode afetar os automóveis sem condutor?
- Como é que podemos evitar modelos treinados enviesados?

Através desta atividade, os alunos aprenderão:

- Como utilizar a ferramenta ML Personal Audio Classifier, ou ferramentas ML semelhantes
- Como treinar um modelo com base numa classificação planeada
- Como testar e avaliar um modelo treinado
- Como é que os dados enviesados podem afetar a precisão de um modelo treinado

## 3.7 Atividade 4: Introduzir a ideia de Interação Natural através da integração de um modelo treinado numa aplicação de IA

### 3.7.1 Descrição

Nesta 4ª Atividade, os alunos aprenderão a integrar o modelo treinado, produzido no contexto da 3ª Atividade, na aplicação criada na 2ª Atividade, de modo a observar como o desempenho do carro robótico pode ser afetado quando um modelo treinado é integrado na aplicação. Desta forma, os alunos tomam

consciência de como os sistemas de IA podem ser propensos a erros devido às limitações da IA na interação de uma forma natural.

### 3.7.2 Integrar o modelo treinado na aplicação de IA

Para as necessidades desta atividade, pode continuar a trabalhar na aplicação criada na Atividade 2, ou pode usar o ficheiro "*Robotic\_car\_SpeechRecognizer.aia*". Em alternativa, pode considerar a criação de um novo projeto *.aia*, mas é altamente recomendável continuar a trabalhar no ficheiro anterior, uma vez que seria mais fácil para os alunos compararem as diferenças no desempenho do carro robótico, quando utilizam o Reconhecedor de Fala e quando utilizam o Classificador de Áudio Pessoal.

#### Adicionar alguns componentes extra

Primeiro, é necessário adicionar alguns componentes extra ao ecrã da aplicação (no menu do designer). Especificamente, é necessário adicionar **a)** um componente **WebView**, **b)** a extensão **Personal Audio Classifier** e **c)** uma etiqueta adicional onde serão apresentados os resultados da classificação.

**a)** O componente **WebView** permite que as aplicações alojem um URL e redirecionem o utilizador para uma página Web específica. Para efeitos desta atividade, este componente é adicionado para permitir que o **Personal Audio Classifier** carregue a base de dados incorporada no modelo treinado.

Para encontrar este componente, vá ao sub-menu **Interface do Utilizador** e arraste-o e largue-o no ecrã, sob o esquema de **Disposição Horizontal** (Figura 31).

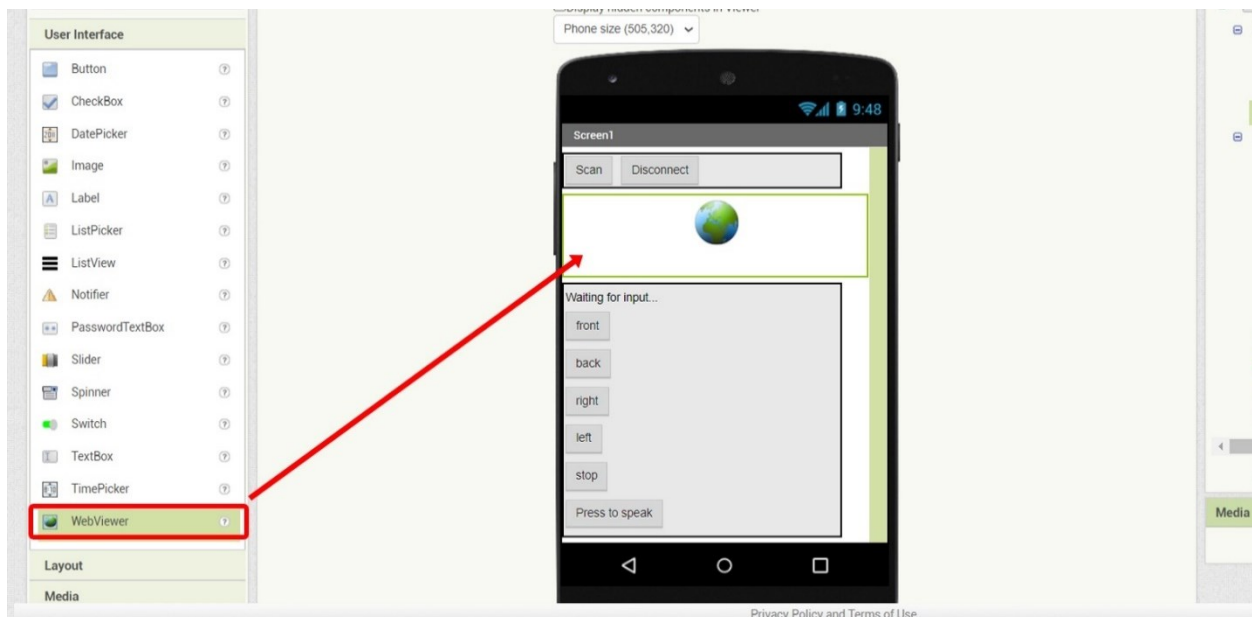
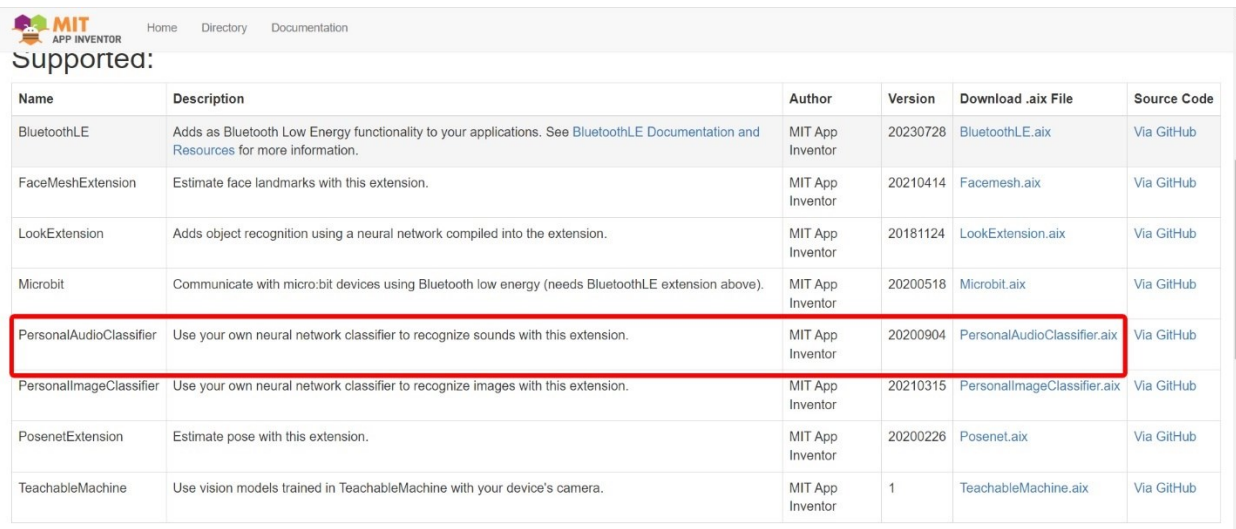


Figura 31: Adicionar o componente *WebView*

**b)** O passo seguinte é adicionar o componente **Personal Audio Classifier**, que permite à aplicação utilizar o modelo treinado previamente criado pelo ambiente de treino PAC. Para tal, é necessário descarregar esta extensão localmente para o computador. Aceda a esta ligação: <https://mit-cml.github.io/extensions/> e guarde o ficheiro *.aix* correspondente no seu disco local (Figura 32). Em seguida, adicione esta extensão ao ambiente do App Inventor da mesma forma que adicionou as extensões para Bluetooth e micro:bit.



Name	Description	Author	Version	Download .aix File	Source Code
BluetoothLE	Adds as Bluetooth Low Energy functionality to your applications. See <a href="#">BluetoothLE Documentation and Resources</a> for more information.	MIT App Inventor	20230728	<a href="#">BluetoothLE.aix</a>	<a href="#">Via GitHub</a>
FaceMeshExtension	Estimate face landmarks with this extension.	MIT App Inventor	20210414	<a href="#">Facemesh.aix</a>	<a href="#">Via GitHub</a>
LookExtension	Adds object recognition using a neural network compiled into the extension.	MIT App Inventor	20181124	<a href="#">LookExtension.aix</a>	<a href="#">Via GitHub</a>
Microbit	Communicate with micro:bit devices using Bluetooth low energy (needs BluetoothLE extension above).	MIT App Inventor	20200518	<a href="#">Microbit.aix</a>	<a href="#">Via GitHub</a>
PersonalAudioClassifier	Use your own neural network classifier to recognize sounds with this extension.	MIT App Inventor	20200904	<a href="#">PersonalAudioClassifier.aix</a>	<a href="#">Via GitHub</a>
PersonalImageClassifier	Use your own neural network classifier to recognize images with this extension.	MIT App Inventor	20210315	<a href="#">PersonalImageClassifier.aix</a>	<a href="#">Via GitHub</a>
PosenetExtension	Estimate pose with this extension.	MIT App Inventor	20200226	<a href="#">Posenet.aix</a>	<a href="#">Via GitHub</a>
TeachableMachine	Use vision models trained in TeachableMachine with your device's camera.	MIT App Inventor	1	<a href="#">TeachableMachine.aix</a>	<a href="#">Via GitHub</a>

Figura 32: Descarregar a extensão Personal Audio Classifier clicando em PersonalAudioClassifier.aix

Depois de importar a extensão PersonalAudioClassifier, arraste-a e solte-a no ecrã da aplicação concebida. O PersonalAudioClassifier é também um componente não visível, pelo que aparecerá na secção de componentes não visíveis.

Em seguida, selecione o componente PersonalAudioClassifier no menu Components (Componentes) para modificar as suas propriedades (Figura 33). Especificamente, clique no campo "Nenhum..." nas propriedades Modelo (1) e WebViewer (2) para carregar o ficheiro do modelo treinado .mdl (que criou anteriormente no ambiente de treino PAC) e para seleccionar o componente WebViewer1 na lista flutuante (3).

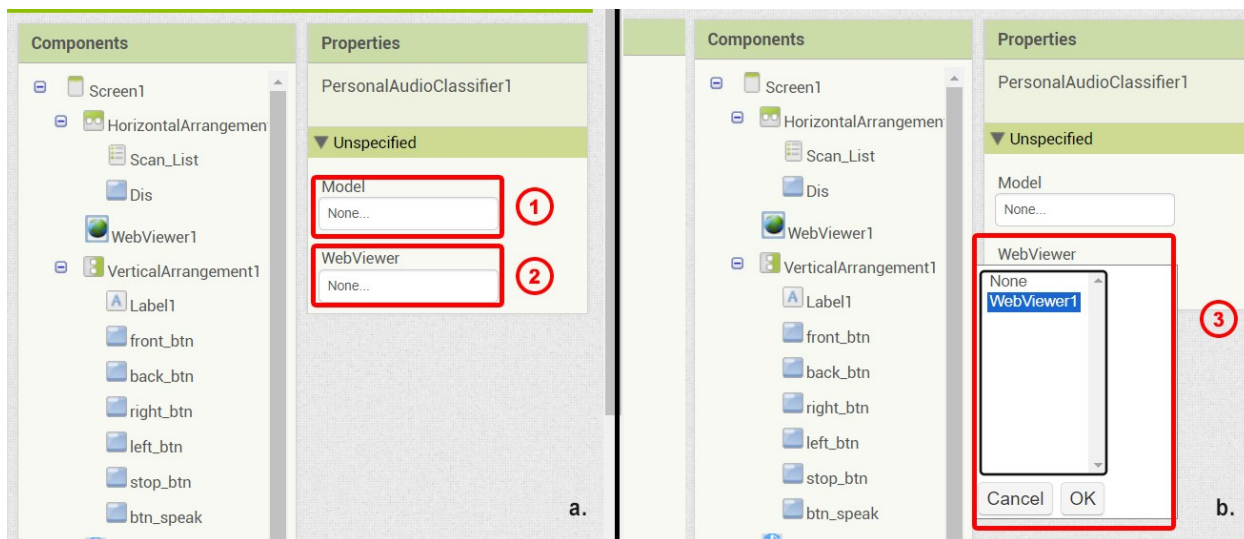


Figura 33a. Modificar as propriedades do PersonalAudioClassifier; b. Carregar o ficheiro do modelo treinado e seleccionar o componente WebViewer1

c) O último passo é adicionar mais uma etiqueta, que apresentará os resultados da classificação, sempre que um novo som for registado pela aplicação.

Arraste e largue um componente de etiqueta no ecrã, sob o componente WebViewer e, no menu Propriedades, altere o texto da etiqueta para "A aguardar classificação" ou algo semelhante. Também pode mudar o nome do componente de etiqueta - a partir do menu Componentes - para algo significativo, como Etiqueta\_de\_classificação (Figura 34).

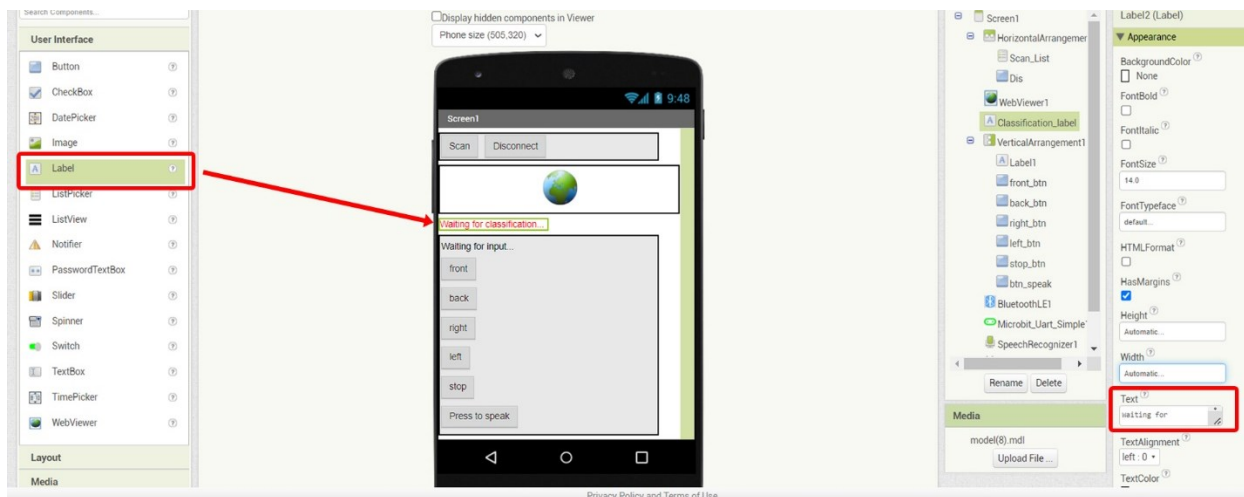


Figura 34: Adicionar uma segunda etiqueta para apresentar os resultados da classificação

Após este passo, pode programar os novos componentes.

### Programação do componente PersonalAudioClassifier

O passo seguinte é programar o componente PersonalAudioClassifier e, especificamente, a sua funcionalidade após a classificação áudio de um som de entrada ter sido concluída.

Para esta etapa da programação, deve ser implementado o comando do bloco de eventos **"when PersonalAudioClassifier1 .GotClassification...result...do"** (2), localizado no menu flutuante do componente PersonalAudioClassifier (1) (Figura 35). Arraste e largue este comando de bloco de eventos para a área onde o código está montado.

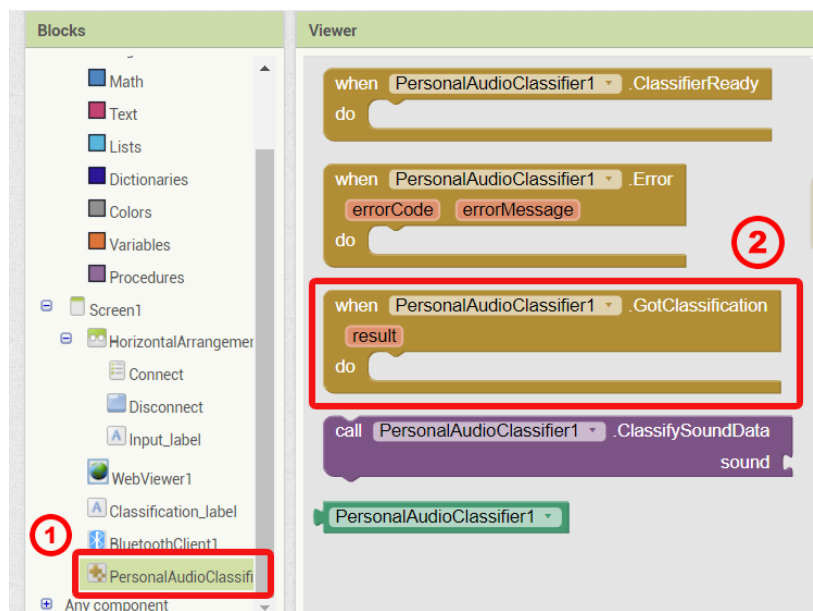


Figura 35: Encontrar o comando do bloco de eventos "When PersonalAudioClassifier1 .GotClassification...result ..do..."

O componente Personal Audio Classifier adiciona um botão Gravar à aplicação. Este botão não é visível na área de design. Só será visível no dispositivo inteligente (depois de o desenho e a programação da aplicação terem sido concluídos, e depois de a aplicação ter sido construída e instalada num dispositivo inteligente) (Figura 36). O comando do bloco Evento acima determina o que a aplicação deve fazer com os resultados da gravação, depois de o comando de voz recebido ter sido classificado (ou seja, dar instruções ao carro robótico para se mover em conformidade).

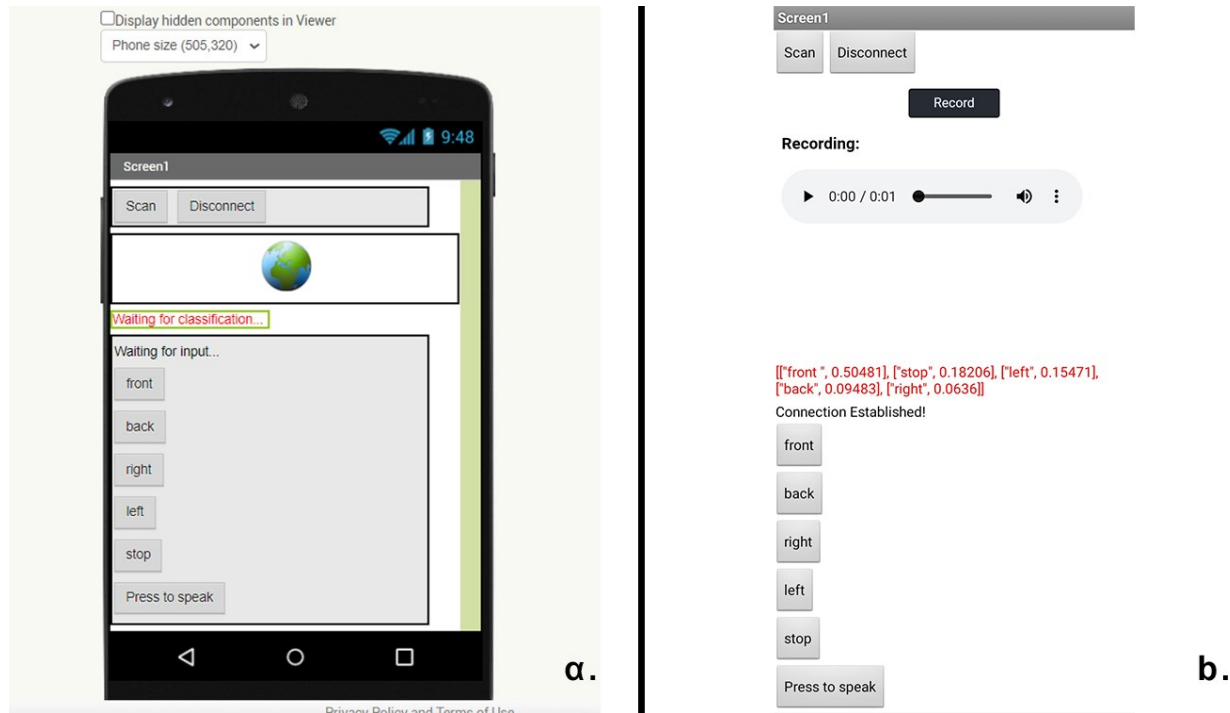


Figura 36a. a aplicação tal como aparece na vista do designer; b. pré-visualização da aplicação quando instalada num dispositivo inteligente

Em primeiro lugar, é necessário adicionar um comando que permita ao componente Classification\_label apresentar os resultados da classificação do áudio. Para tal, seleccione o componente Etiqueta\_de\_Classificação (1) e, a partir do menu flutuante, arraste e coloque o comando de bloco "set Etiqueta\_de\_Classificação". Text to" (2) dentro do comando do manipulador de eventos (Figura 37a). Em seguida, mova o cursor sobre o campo do resultado (3) e arraste e encaixe o comando de bloco "get result" (4) no comando "set Classification\_label". Text to" (Figura 37b).

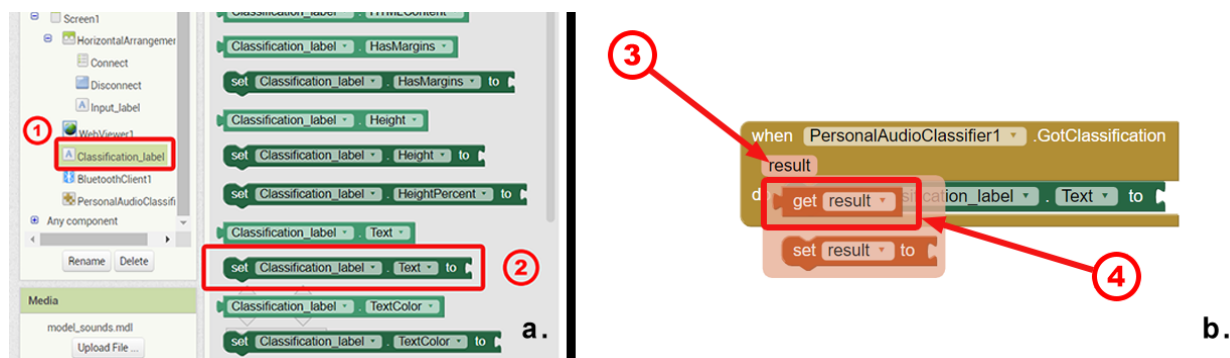


Figura 37a. Encontrar o comando de bloco "set... Text to"; b. Encontrar o comando de bloco "set result".

Através deste processo, o texto "waiting for classification", em Classification\_label (ver Figura 34), mudará para os resultados da classificação de áudio, com base no modelo de classificação que carregámos, e seguido do nível de confiança (por exemplo, se o comando de voz frontal for registado, então o texto da etiqueta pode potencialmente mudar para algo como o mostrado na Figura 36b, ou seja, "[front, 0.50], [stop, 0.18], [left, 0.15], etc.").



Em seguida, no comando "set Classification\_label" . Text to", coloque uma condição "if then...else if then ...else" (2), localizada no menu Controlo (1) (Figura 38), para determinar o que a aplicação fará com base nos resultados obtidos no processo de classificação.

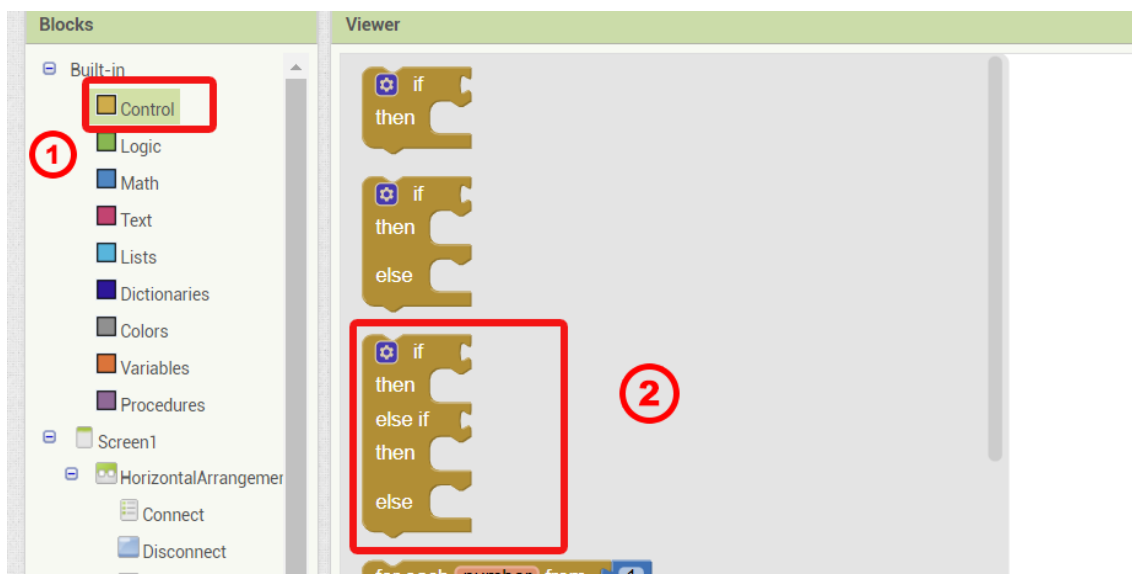
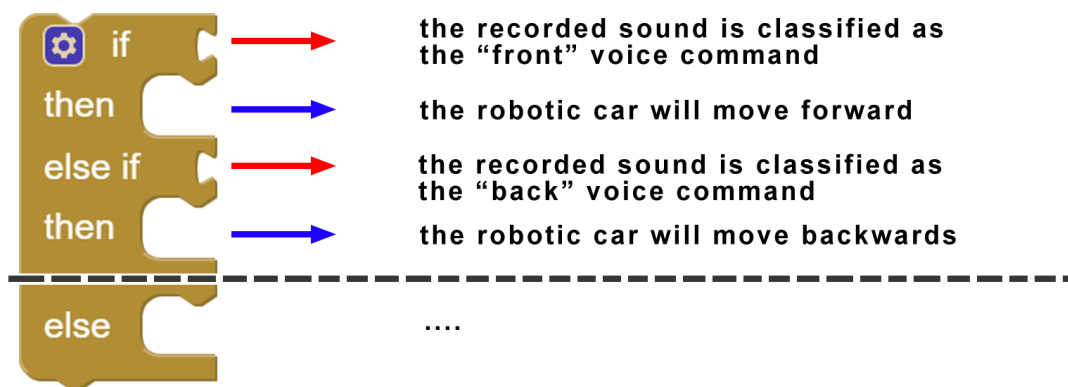


Figura 38: Encontrar a condição "se então...senão se então...senão"

O diagrama seguinte apresenta os parâmetros que precisamos de inserir dentro da condição "if then... else if then ...else".



Em particular, precisamos de um conjunto de comandos que reconheçam e verifiquem qual a categoria/etiqueta classificada que está a ser chamada. Estes comandos serão colocados dentro da instrução "if" ou "else if". Também precisamos de mais alguns comandos que instruem o carro robótico a efetuar o movimento adequado, de acordo com os resultados da classificação. Estes comandos serão colocados dentro da instrução "then".



Para verificar que categoria/etiqueta classificada é chamada, deve ser utilizado um bloco igual (=) (2) do menu Lógica (1) (Figura 39a) e um bloco de comando (4) "selecionar lista de itens da lista... índice...", situado no menu Listas (3) (Figura 39b).

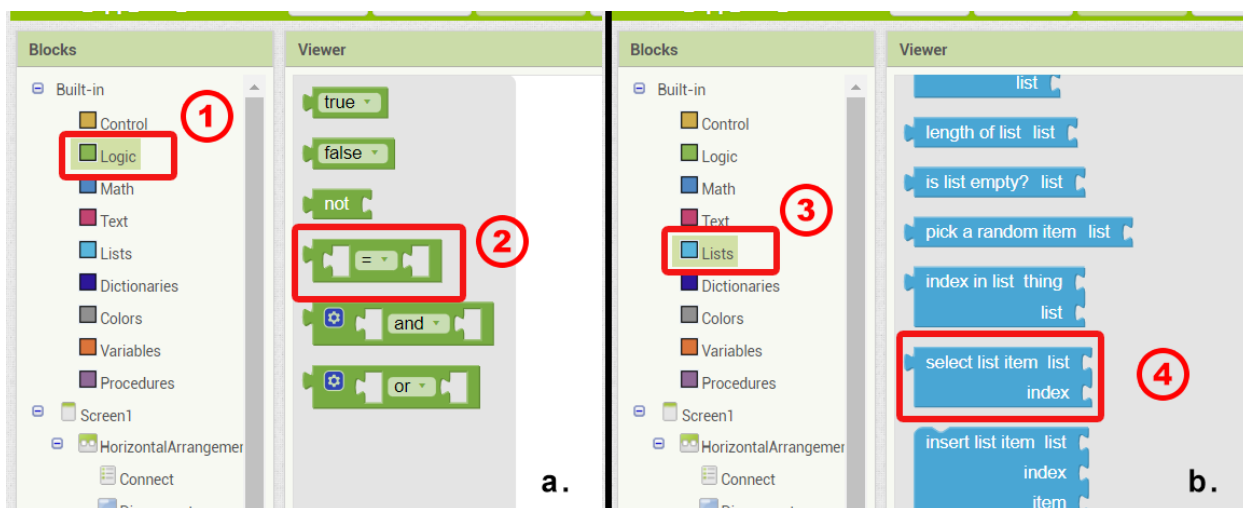


Figura 39a. Encontrar o bloco igual (=); b. Encontrar o bloco "selecionar índice da lista de itens".

Primeiro, coloque o bloco igual (=) dentro da instrução if, como indicado no seguinte bloco de comandos.

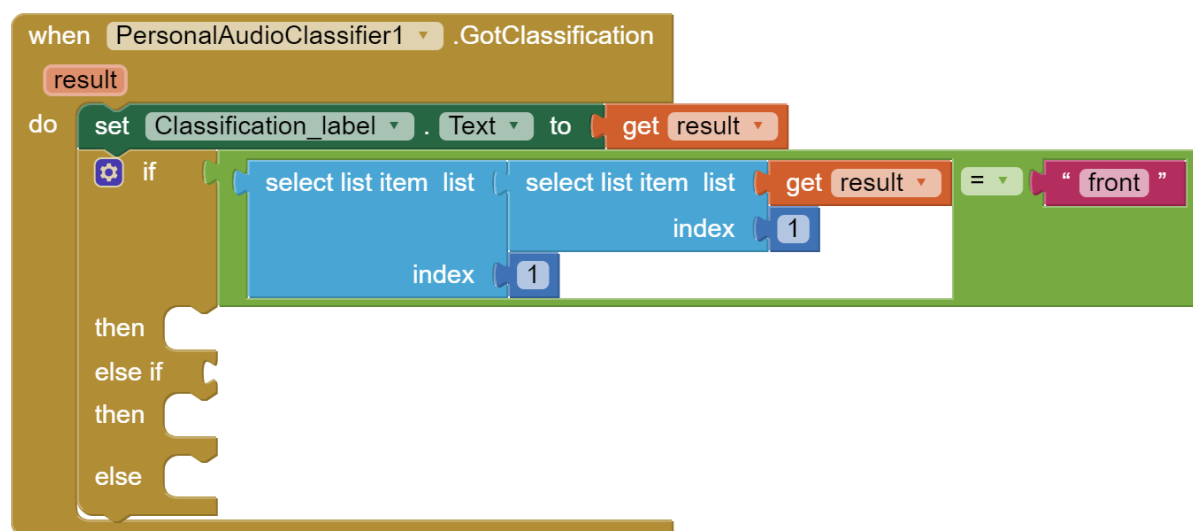


O resultado devolvido é, de facto, uma **lista** com as classificações do modelo carregado. A lista contém cinco sub-listas. Cada sub-lista contém uma das etiquetas/categorias (i.e., "frente", "trás", etc.) que o classificador pensa ser uma correspondência, seguida do nível de confiança. No exemplo indicativo acima mencionado: [frente, 0,50], [parar, 0,18], [esquerda, 0,15], etc., o classificador está 50% confiante de que o comando recebido é "frente", 18% confiante de que o comando recebido é "parar", etc. Precisamos de retirar o primeiro item da primeira sub-lista ("trás" no caso acima mencionado) e testar se é a categoria a que pertence o som recebido.

Para tal, serão utilizados **dois** blocos de comandos "selecionar lista item lista... índice...".



Para começar, vamos obter o primeiro item (índice) do **resultado obtido** (a nossa lista principal). Em seguida, este item tornar-se-á a nossa nova lista principal, da qual extrairemos novamente o primeiro item (índice), que, no nosso exemplo, é "frente" (digitado dentro de um **bloco de entrada de texto**). O primeiro item da lista está sempre no índice 1 e, no nosso caso, é digitado dentro de um **bloco numérico básico**.

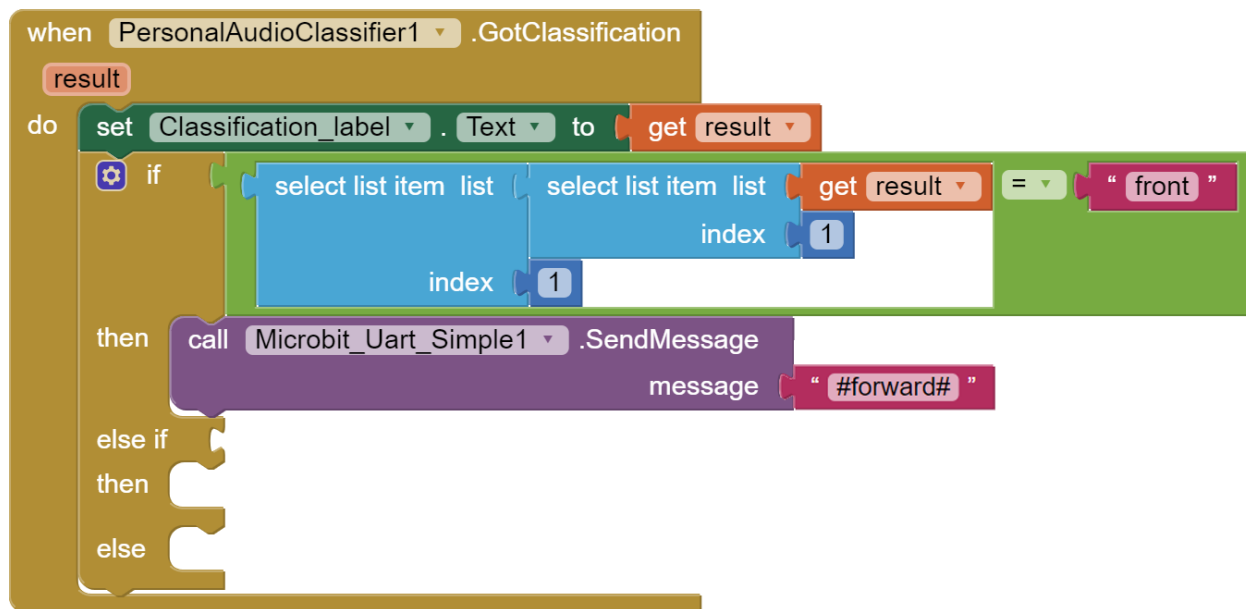


**Nota:** O bloco de números básicos está localizado no menu **Matemática**, enquanto o bloco de entrada de texto está localizado no menu **Texto**.

O próximo passo é dar instruções ao carro robótico para avançar. Para isso, temos de dar instruções à aplicação para transmitir (através de Bluetooth) a mensagem correspondente (ou seja, #forwardt#) ao carro robótico (ou seja, tal como foi declarado no guião Makecode).

Portanto, dentro da instrução "then", coloque um bloco "call Microbit\_Uart\_Simple1.SendMessage message". Em seguida, anexe um **bloco de entrada de texto** "" e digite a palavra "#forward#".

Este é o aspeto do código depois de completar as duas primeiras declarações da condição "if then...else if then...else".



Repita o mesmo processo para os outros comandos para programar a sua aplicação, dando instruções ao carro robótico para executar outros movimentos, de acordo com o som de entrada classificado.

**Nota:** se pretender adicionar mais condições "else if", clique na engrenagem azul (1) ao lado da declaração "if" e, a partir do menu flutuante (2), arraste e largue tantas condições "else if" quantas as necessárias (Figura 40).

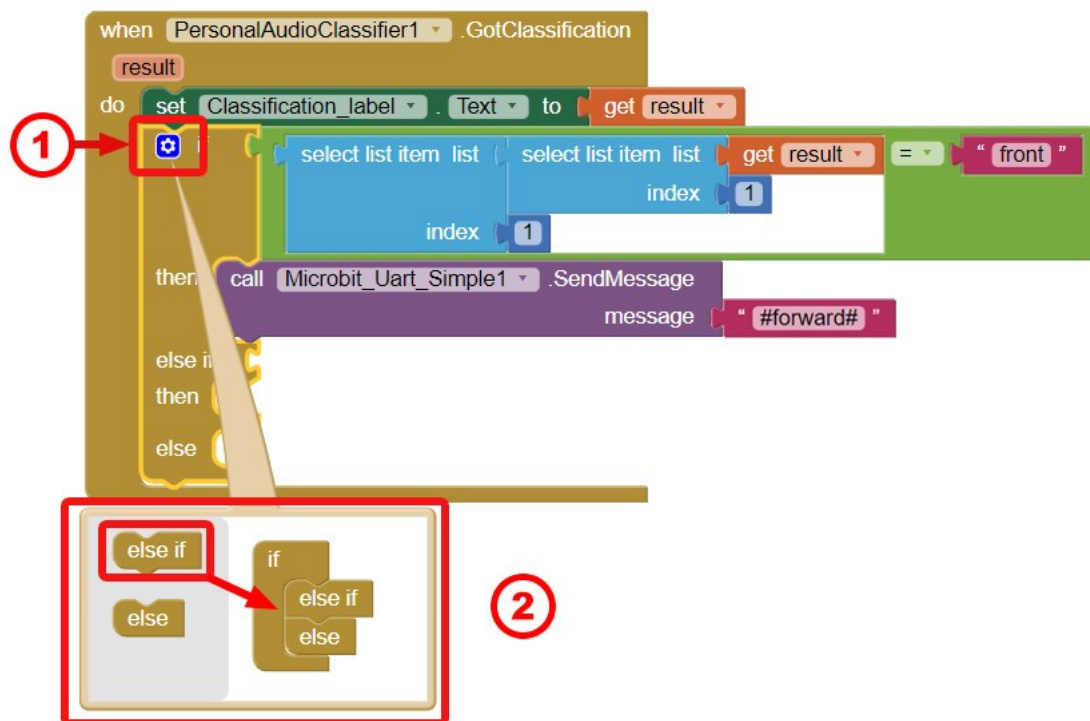
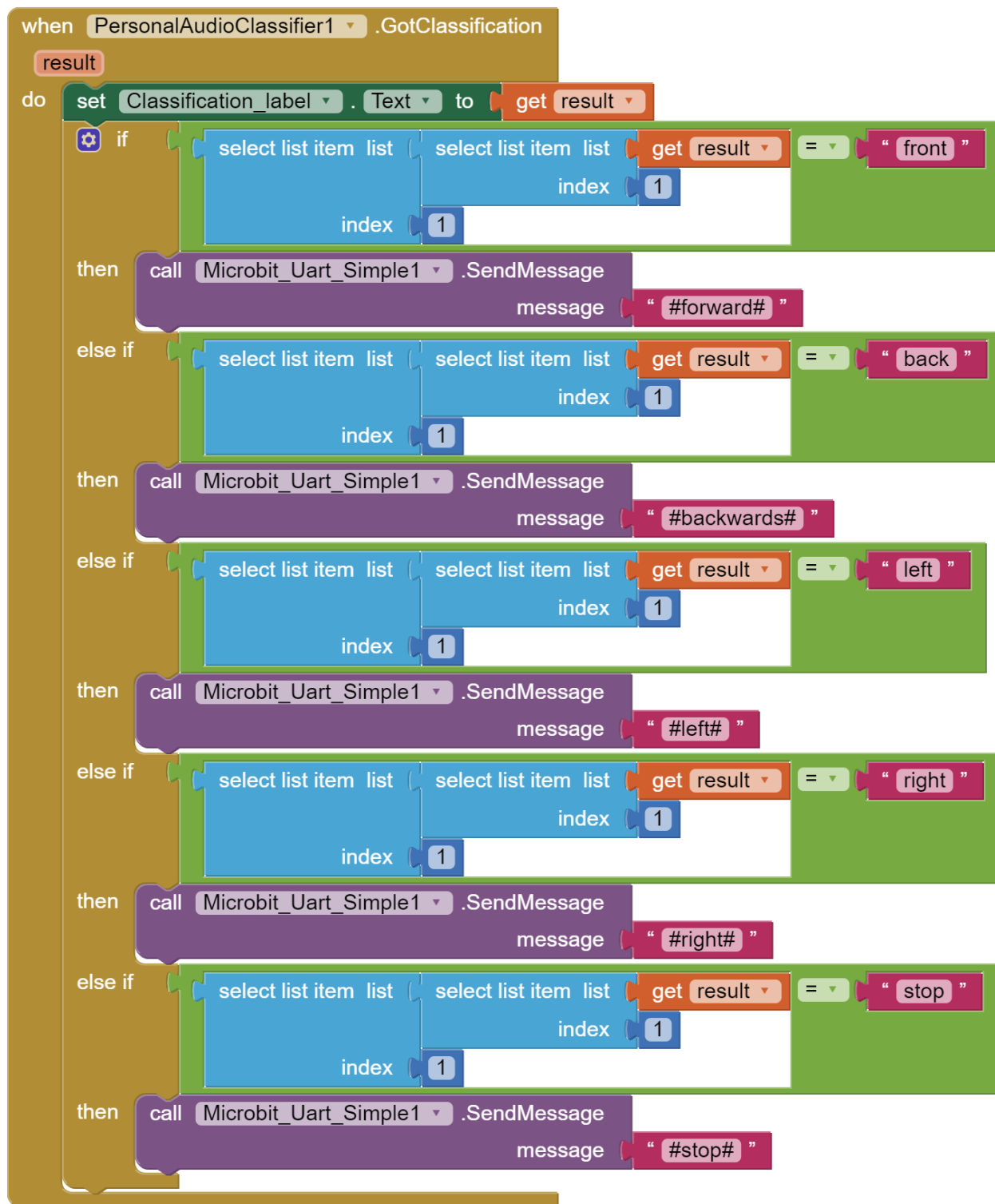


Figura 40: Acrescentar mais condições else if

**Nota:** Em cada bloco de entrada de texto, é crucial a) utilizar a mesma etiqueta que utilizou durante a produção do modelo de classificação, b) inserir a mesma mensagem que declarou no script Makecode.

A imagem seguinte apresenta o guião completo para o exemplo de classificação que foi criado no presente documento.



Quando terminar todos os passos acima mencionados, a aplicação está pronta para ser carregada e instalada no seu dispositivo inteligente. Vá ao menu Build (Construir) e selecione "Android App (.apk)" no menu pendente para iniciar o processo de produção do ficheiro .apk. Isto pode demorar alguns minutos. Em seguida, instale a aplicação no seu dispositivo inteligente através da aplicação MIT AI2 Companion.

**Nota importante:** Em alguns casos (especialmente se o modelo treinado tiver demasiadas etiquetas/categorias), o modelo treinado pode não funcionar corretamente quando integrado na aplicação, conduzindo a resultados falsos. Se os alunos se depararem com um problema deste tipo, incentive-os a experimentar um modelo treinado **mais pequeno** (com apenas 2 ou 3 rótulos) e a pensar em cenários alternativos em que este serviço de IA seria mais útil (por exemplo, ligar e desligar o carro). No caso de um modelo treinado mais simples, o script PersonalAudioClassifier poderia ser parecido com o mostrado na Figura 41.

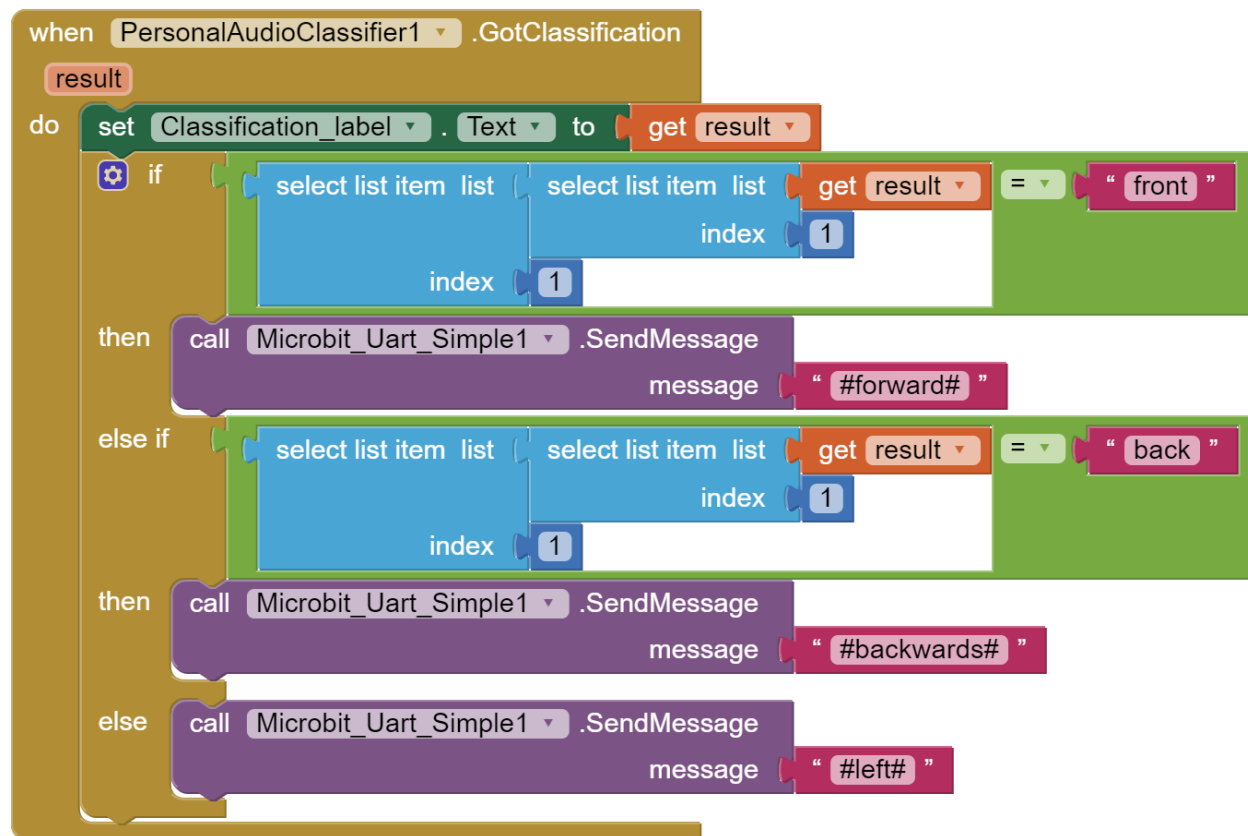


Figura 41: Um guião para programar o Personal Audio Classifier quando o modelo treinado contém apenas duas categorias, como frente e verso.

### 3.7.3 Experiência 4

Para apresentar esta atividade sem problemas aos seus alunos, pode utilizar o documento da ficha de trabalho correspondente (*Students\_worksheet\_for\_Activity\_4.pdf*). Antes de o fazer, certifique-se de que eles compreendem o objetivo desta atividade (ou seja, avaliar um modelo treinado integrando-o numa aplicação e descobrir as limitações da IA).

Depois de os alunos terem criado a aplicação, incentive cada equipa a testar a aplicação criada por outras equipas. Em seguida, incentive-os a partilhar as suas experiências em plenário.

Algumas **questões** que podem ser colocadas para iniciar o diálogo são

- A sua aplicação funcionou corretamente?
- Acha que vozes diferentes ou pronúncias diferentes afectam o resultado?

- Consegue lembrar-se de algum parâmetro que possa levar a um mau funcionamento?
- Quais são os riscos ou perigos de utilizar um modelo treinado que contém erros?
- Por vezes, um modelo treinado com demasiadas categorias pode levar a problemas de funcionamento. Consegue pensar num cenário alternativo para o seu carro robótico em que seria necessário um modelo treinado mais pequeno (com menos rótulos/categorias)?

Através desta atividade, o aluno compreenderá que:

- Há muito trabalho a fazer para que um sistema de IA possa interagir de uma forma mais natural
- Diferentes pronúncias ou tons de voz podem levar a disfunções
- Há sérios riscos na utilização de um sistema de IA na vida quotidiana, se o conjunto de dados não for devidamente treinado ou contiver enviesamentos

## 3.8 Atividade 5: Introduzir a ideia de impacto social

### 3.8.1 Descrição

Nesta atividade, os alunos serão introduzidos à Grande Ideia do 5º, nomeadamente o Impacto Social, refletindo sobre as experiências que adquiriram durante a realização das quatro atividades anteriores. Em particular, serão incentivados a refletir sobre as vantagens, as desvantagens e os riscos inerentes à utilização de serviços e ferramentas de IA, bem como sobre a monitorização de dados e a tomada de decisões com base em conjuntos de dados específicos. Esta atividade pode ser realizada separadamente ou combinada com as quatro anteriores. Desta forma, os alunos tomarão consciência de várias decisões éticas que devem ser tidas em conta aquando da conceção e utilização de serviços e tecnologias de IA e IoT.

Algumas questões sobre questões éticas já foram abordadas em atividades anteriores. Seguem-se alguns exemplos adicionais para iniciar o diálogo nesta direção:

1) Pode incentivar os seus alunos a pensar num cenário em que o carro robótico recolhe dados mais sensíveis (por exemplo, imagens de pessoas para permitir que o carro reconheça os peões) e transmite esses dados para a nuvem ou para outros serviços.

- Quais são as vantagens e desvantagens desta tecnologia?
- Que parâmetros devem ser tidos em conta no que respeita à segurança destes dados?

2) Incentive os seus alunos a refletir sobre o modo como um modelo tendencioso pode afetar a forma como um robô inteligente "pensa", levando à construção de representações tendenciosas do mundo. Incentive-os a imaginar um cenário em que um carro robótico sem condutor, utilizado para resgatar pessoas presas, é treinado para reconhecer sinais sonoros de socorro que recebe, mas não está devidamente treinado. Quais seriam as consequências desta deturpação da realidade?

3) Imagine um caso em que um carro robótico é treinado para reconhecer apenas vozes masculinas e apenas vozes masculinas de barítono. Como é que um modelo treinado de forma tão tendenciosa pode afetar a vida de várias outras pessoas?



### 3.9 Material e recursos

Tipo de recurso	Título	Tópico	Ligação
Ficheiro Pdf	T2.4_Criação_do_veículo_robotico	Diretrizes para a criação do carro robótico	
Ficheiro Pdf	T2.4_Atividades_de_aquecimento_para_o_veículo_robotico	Atividades de programação de aquecimento para se familiarizar com os comandos do carro robótico	
Ficheiro Pdf	Cartão_de_circuito_Atividade1	Material para ajudar os alunos a criar o circuito para a atividade 1 <sup>st</sup>	
Ficheiro Pdf	Atividade_meio_cozinha1	Documento com uma solução incompleta do código para a atividade 1 <sup>st</sup>	
Ficheiro Pdf	T2.4_App_Inventor_Aquecimento	Uma atividade de aquecimento para se familiarizar com o MIT App Inventor e criar uma aplicação para controlar o carro robótico à distância	
ficheiro .aia	Controlo remoto_Microbit	Um ficheiro App Inventor para utilizar o ficheiro produzido no âmbito da atividade App Inventor Warm Up e que pode ser utilizado para a atividade 2 <sup>a</sup>	
Ficheiro Pdf	T2.4_Programação_do_veículo_robotico	Um documento com diretrizes para a criação do guião de modo a programar o carro robótico para seguir as ordens recebidas pela aplicação concebida	
ficheiro .aia	Reconhecedor de fala de carro robótico	Um ficheiro App Inventor que pode ser utilizado para a atividade 4 <sup>th</sup>	
Ficheiro Pdf	Ficha de trabalho_dos_alunos_da_Atividade_2	Ficha de trabalho dos alunos para os ajudar a implementar a atividade 2 <sup>nd</sup>	
Ficheiro Pdf	Ficha de trabalho_dos_alunos_da_Atividade_4	Ficha de trabalho dos alunos para os ajudar a realizar a 4 <sup>a</sup> atividade	

### 3.10 O hardware do carro robótico

A Figura 42 apresenta os componentes eletrónicos básicos necessários para criar o carro robótico. Em particular, vai precisar de um microcontrolador BBC micro:bit (1), de um Kitronik Compact Motor Driver (2), de um suporte de pilhas 3AA (ou 4AA) (de preferência com fios pré-fixados) (3), de 2 motores de engrenagem DC (de preferência com fios pré-fixados) (4) e de 2 rodas (5).

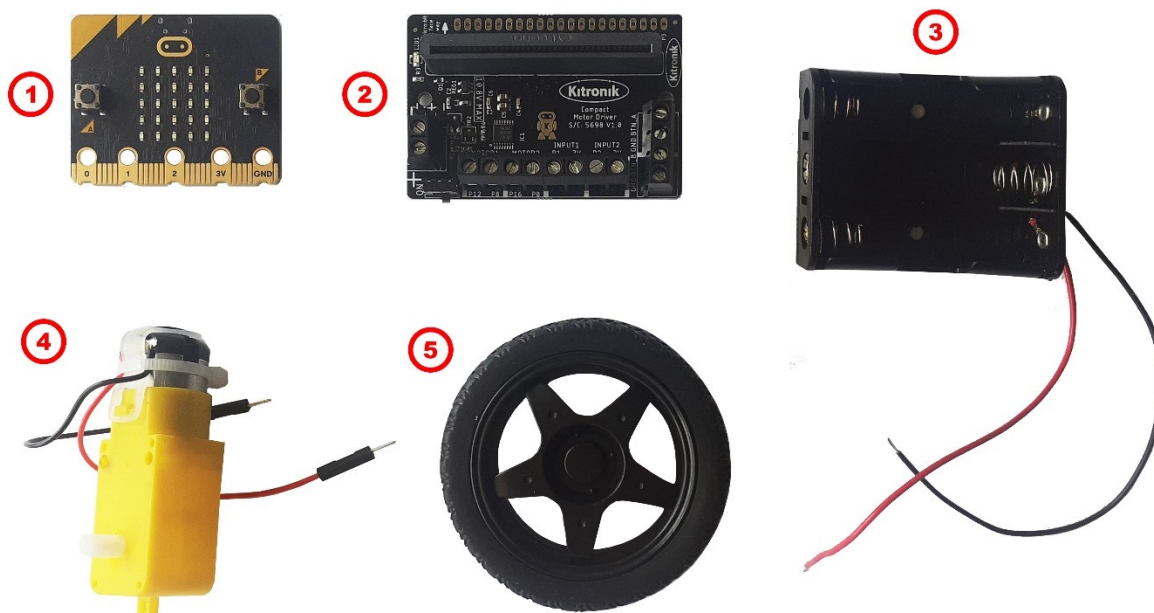


Figura 42: Os componentes electrónicos necessários para criar o carro robótico