



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.

Introdução ao MIT App Inventor - criação de uma aplicação para controlar remotamente o 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. criar uma aplicação para o controlo remoto do carro robótico DIY	3
1.1 Introdução	3
1.2 Planear a conceção da aplicação	3
1.3 Conceber a aplicação	3
Criar o esquema para colocar/arranjar os botões	4
1.4 Programação da aplicação	15
Codificação do botão Desconectar (ou seja, Dis)	19
1.5 Criar a aplicação	22
1.6 Emparelhar a aplicação com o veículo robótico	23

1. criar uma aplicação para o controlo remoto do carro robótico DIY

1.1 Introdução

Este documento apresenta uma atividade de aquecimento para apresentar aos alunos o ambiente do MIT App Inventor. Através desta atividade, os alunos irão aprender a criar uma aplicação que permitirá controlar remotamente o carro robótico, através de um dispositivo inteligente. Assim, e neste sentido, vão aprender a desenhar a interface da aplicação e a programar os itens nela incluídos. Aconselha-se que, antes desta atividade, instrua os seus alunos a criar o script descrito no ficheiro "T2.4_Programação_do_carro_robótico.pdf", ou a descarregá-lo para o carro robótico, utilizando o ficheiro .hex correspondente.

1.2 Planear a conceção da aplicação

Antes de avançar com a conceção da aplicação, é fundamental conhecer os componentes que terão de ser incluídos. Um dos principais objetivos é encontrar uma forma de estabelecer a ligação e a comunicação entre o nosso dispositivo inteligente e o nosso carro robótico. Para o efeito, vamos definir algumas mensagens que serão enviadas/transmitidas pelo nosso dispositivo inteligente e recebidas pelo nosso carro robótico através de Bluetooth. Cada vez que uma mensagem for enviada/transmitida pelo dispositivo inteligente e recebida pela placa micro:bit, o carro robótico corresponderá de uma forma diferente.

Tendo isto em mente, precisamos de conceber uma aplicação que nos permita:

- controlar o movimento do nosso carro robótico. Para isso, precisamos de 5 botões: 4 deles movem o nosso carro robótico para a frente, para trás, para a direita e para a esquerda, e 1 que pára qualquer movimento
- permitir que a nossa aplicação seja ligada através do Bluetooth do nosso dispositivo
- permitir que a nossa aplicação procure/detecte dispositivos Bluetooth disponíveis e se ligue a um dispositivo escolhido
- desligar a nossa aplicação do dispositivo Bluetooth ligado
- opcionalmente, informar-nos sobre o estado atual da conectividade

Tudo isto ajuda-nos a criar uma ideia concreta do resultado do processo de conceção

1.3 Conceber a aplicação

O design é um processo bastante livre, que se baseia maioritariamente na estética do criador. As instruções que se seguem são indicativas e apresentam uma versão bastante simplificada do aspeto da interface que a nossa aplicação pode ter.

A Figura 1 apresenta uma pré-visualização da interface que vamos criar com base nas necessidades que registámos na secção anterior.

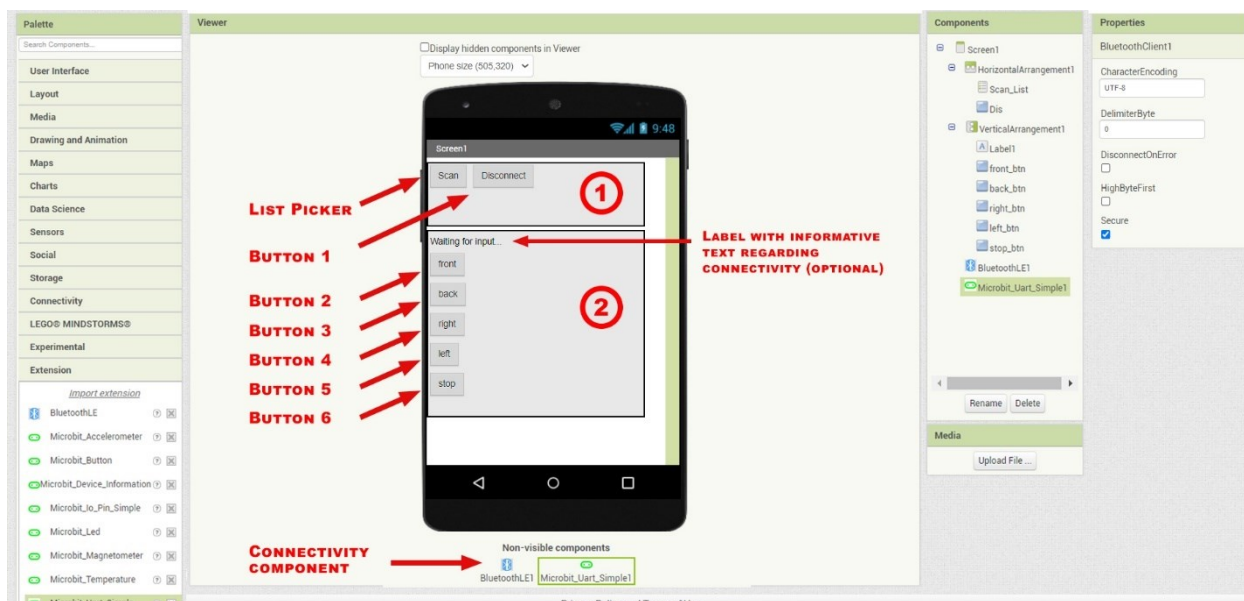


Figura 1: Uma pré-visualização da interface que está prestes a conceber

Para organizar melhor todos os componentes no nosso ecrã, queremos criar dois layouts. Um que irá alojar o botão Scan/search e o botão Disconnect e que nos permitirá organizá-los numa linha (ou seja, um ao lado do outro) (1), e outro que irá alojar os botões de navegação e que nos permitirá organizá-los numa coluna (ou seja, um por baixo do outro) (2).

Vejamos mais detalhadamente a funcionalidade de cada botão:

- 1) **Scan:** O botão "Scan" deve abrir uma lista de todos os dispositivos Bluetooth Low Energy disponíveis na área. A partir dessa lista, o utilizador deve escolher o endereço Bluetooth do Micro:bit. De seguida, a ligação será estabelecida automaticamente. Este botão é diferente porque redireciona o utilizador para uma lista com todas as ligações Bluetooth disponíveis. Para ativar esta funcionalidade, vamos adicionar um botão "ListPicker", que será descrito mais à frente nesta diretriz.
- 2) **Desligar:** Quando o botão "Disconnect" é premido, a ligação entre o micro:bit e o dispositivo inteligente do utilizador é desativada.
- 3) **Botões de navegação (frente, trás, etc.):** Quando um destes botões é premido, o nosso carro robótico desloca-se na direção correspondente.

Sugestão: Durante esta fase, pode aconselhar/incentivar os seus alunos a criar um esboço ou um diagrama da interface e dos componentes nela incluídos, ou/e uma lista com todos os itens necessários. Desta forma, eles poderão organizar melhor os passos para a realização da presente tarefa.

Criar o esquema para colocar/arranjar os botões

O primeiro passo para a criação da nossa aplicação é definir os dois layouts que irão alojar e organizar todos os botões e etiquetas necessários. Para isso, vamos adicionar dois itens de layout, nomeadamente um layout Horizontal que irá alojar os botões ListPicker e Disconnect, e um Vertical que irá alojar os botões

de navegação. Assim, a partir do separador "Layout", arrastamos os componentes "HorizontalArrangement" (1) e "VerticalArrangement" (2) e largamo-los no ecrã (Figura 2).

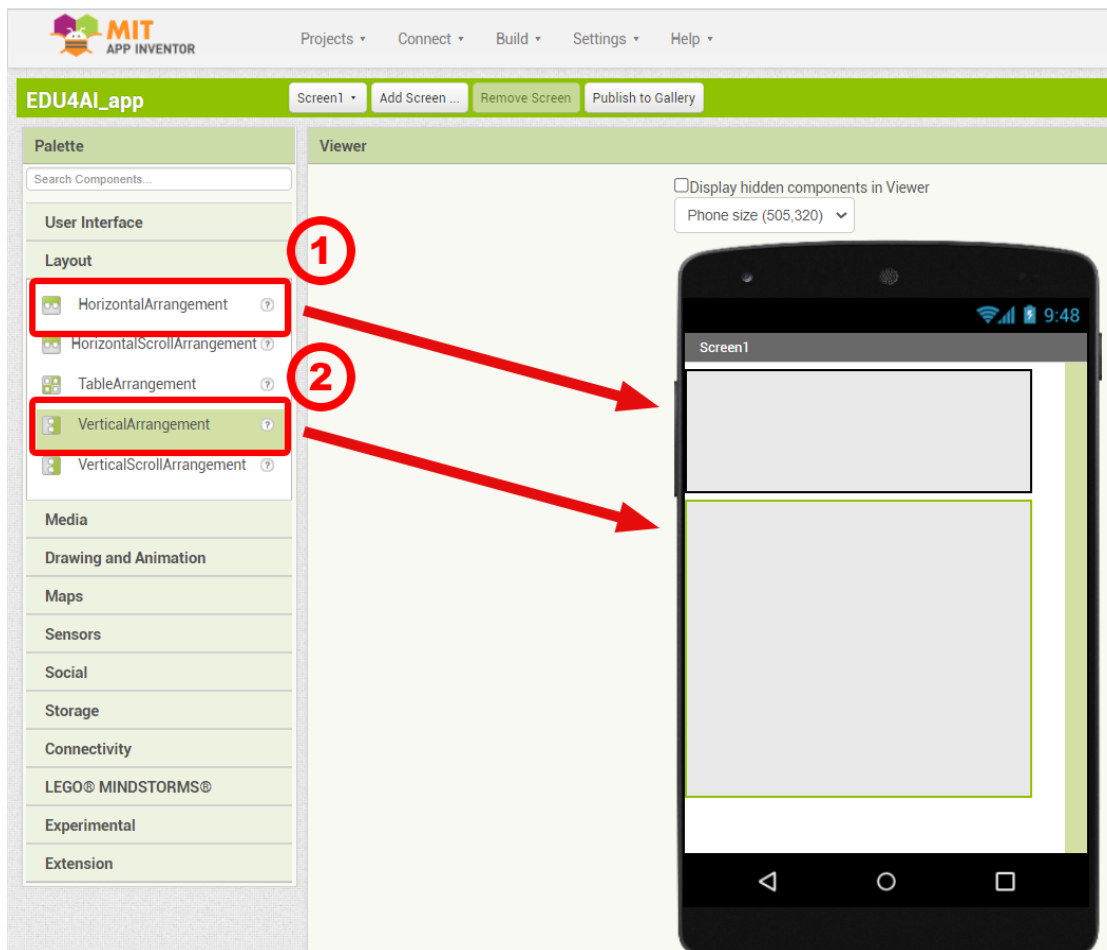


Figura 2: Arrastar e largar os dois itens do Layout no ecrã

Depois de colocar os dois componentes de apresentação no ecrã, podemos ajustar uma série de propriedades, como a altura e a largura, na secção "Properties" (4). Para isso, é necessário seleccionar o componente correspondente na lista Components (3). No exemplo apresentado na Figura 3, definimos a largura do componente "HorizontalArrangement" para 90 por cento e a largura e a altura do componente "VerticalArrangement" para 60 e 90 por cento, respetivamente.

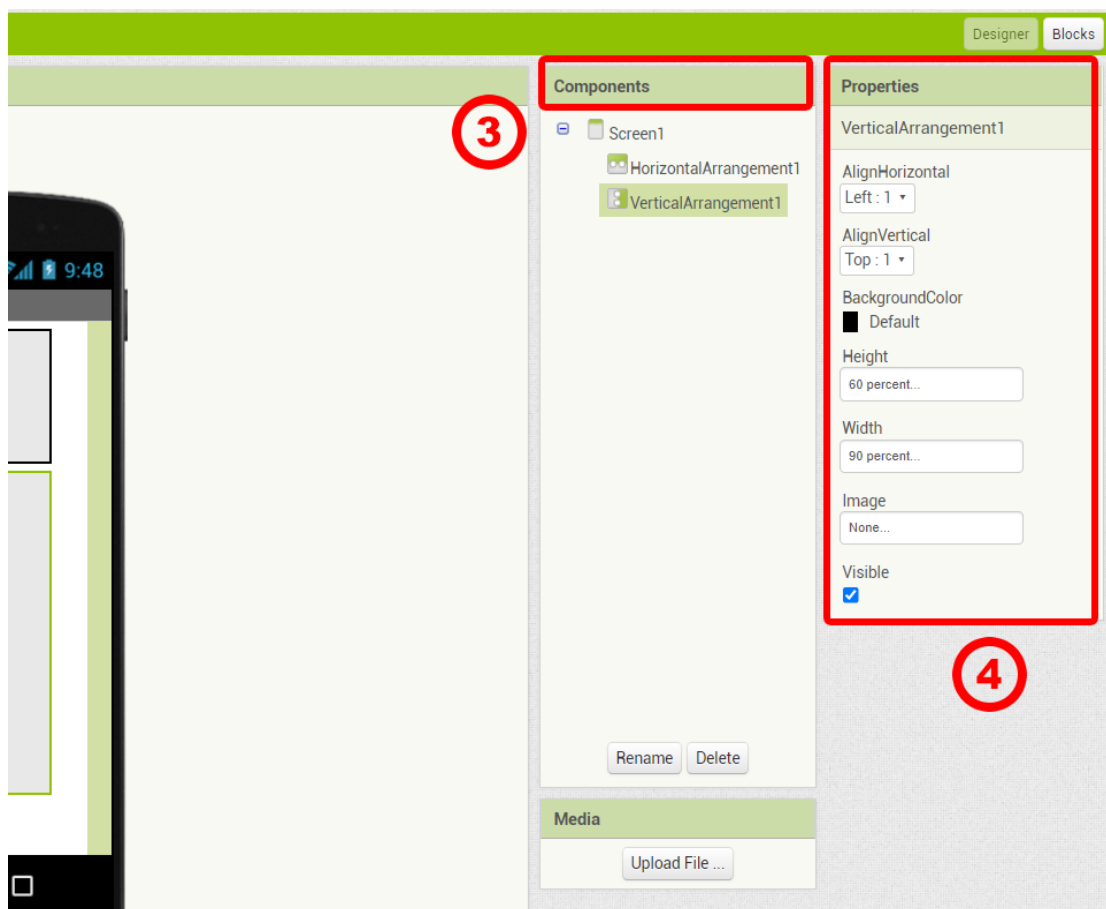


Figura 3: Ajustar as propriedades de cada componente

Sugestão: Se desejar, pode rever e modificar as propriedades acima mencionadas depois de adicionar os botões aos componentes da apresentação.

Adicionar o ListPicker e os botões

O passo seguinte é adicionar o ListPicker e os botões ao ecrã. Para isso, vamos ao submenu "User Interface" (Interface do utilizador) e, a partir da secção "Palette" (Paleta), arrastamos e largamos no ecrã 7 itens, nomeadamente um "ListPicker" e 6 botões. O ListPicker será utilizado para procurar/digitalizar e revelar todos os dispositivos Bluetooth disponíveis.

Especificamente, arrastamos o item "ListPicker" (2) e um botão (1) (ou seja, o botão de desconexão) e largamo-los na disposição "HorizontalArrangement" e, em seguida, arrastamos mais 5 botões (1) (ou seja, os botões de navegação) e largamo-los na disposição "VerticalArrangement" (Figura 4).

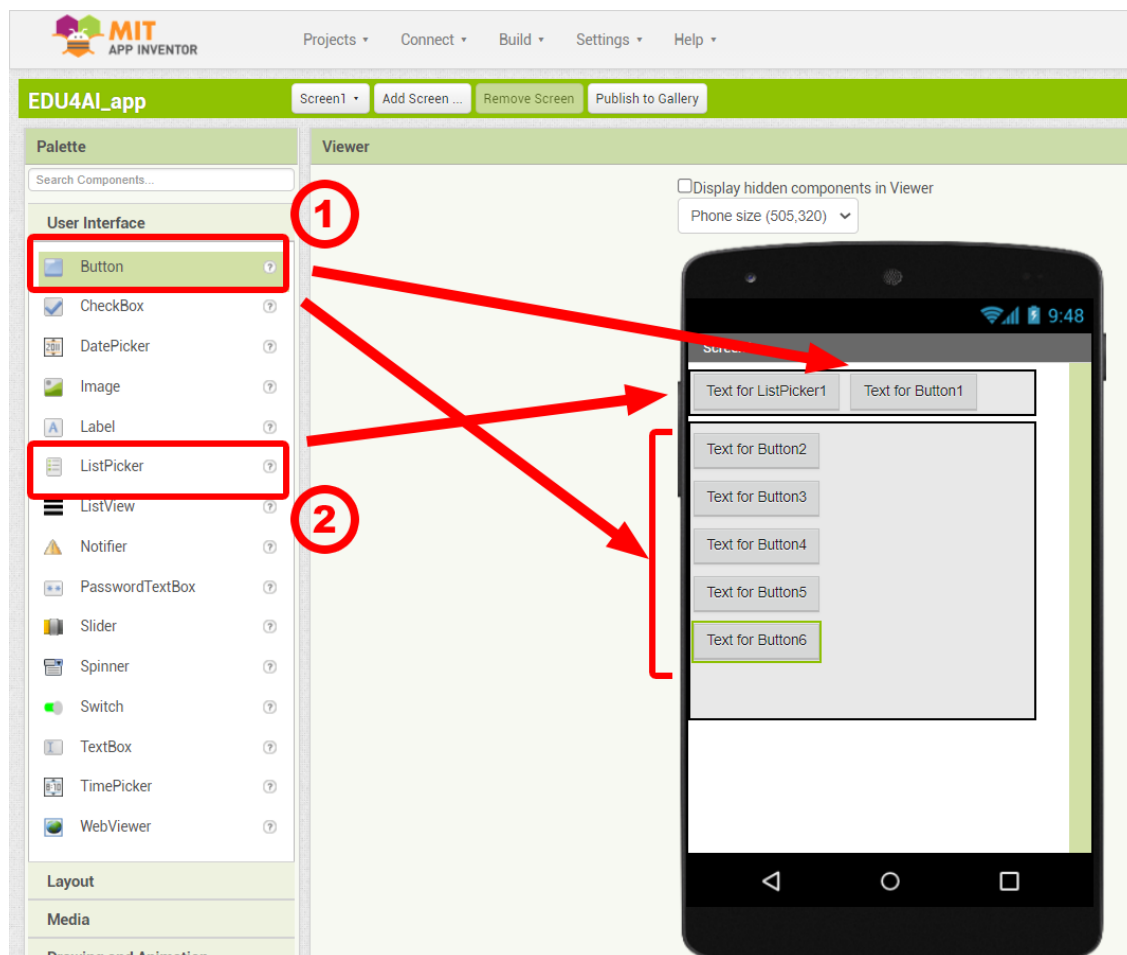


Figura 4: Adicionar um item ListPicker e 6 botões no ecrã

Para otimizar o aspeto da nossa interface, podemos alterar o nome de cada botão através do separador "Texto" (4) localizado no menu "Propriedades". Para o fazer, temos de seleccionar o item correspondente na lista "Components" (3) (Botão 3 no nosso exemplo) e, em seguida, alterar o nome manualmente (Figura 5).

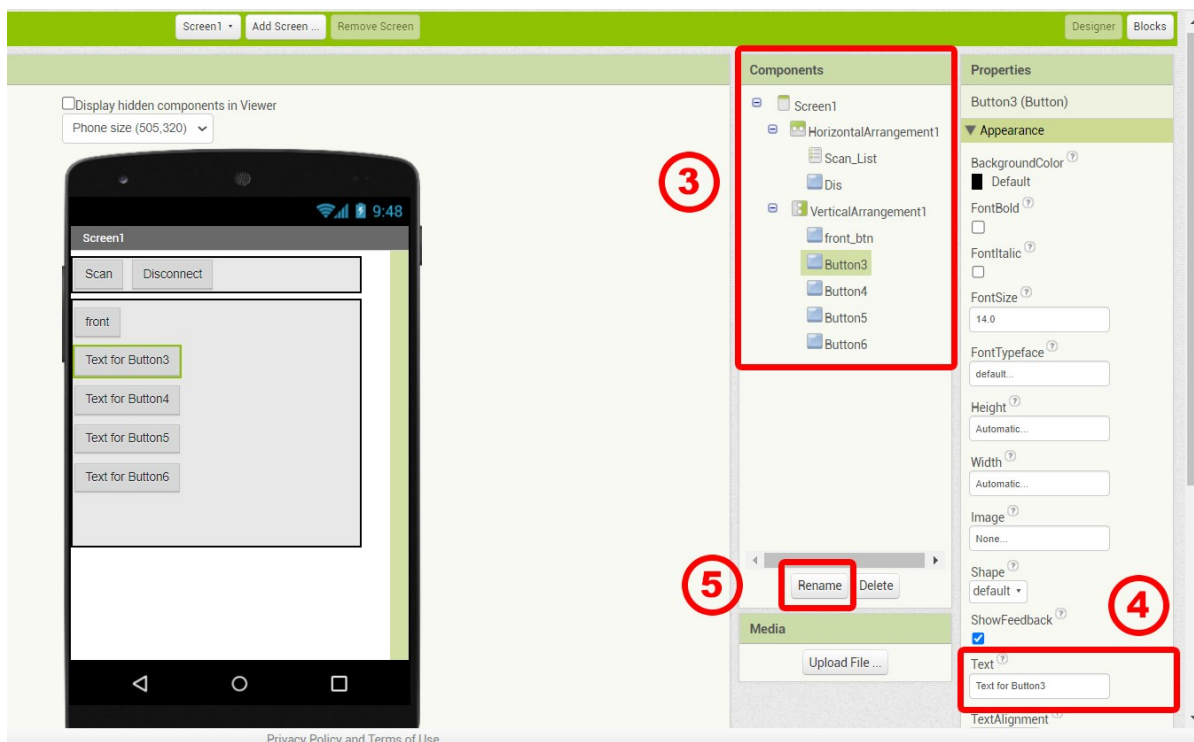


Figura 5: Alterar o conteúdo do texto dos botões

Outra boa prática é alterar os nomes dos botões. Isto também facilitará a fase de codificação mais tarde. Pode fazê-lo clicando no botão "Renomear" (5) e inserindo o novo nome na caixa de texto "Novo nome" localizada no menu pop-up (Figura 5, Figura 6).

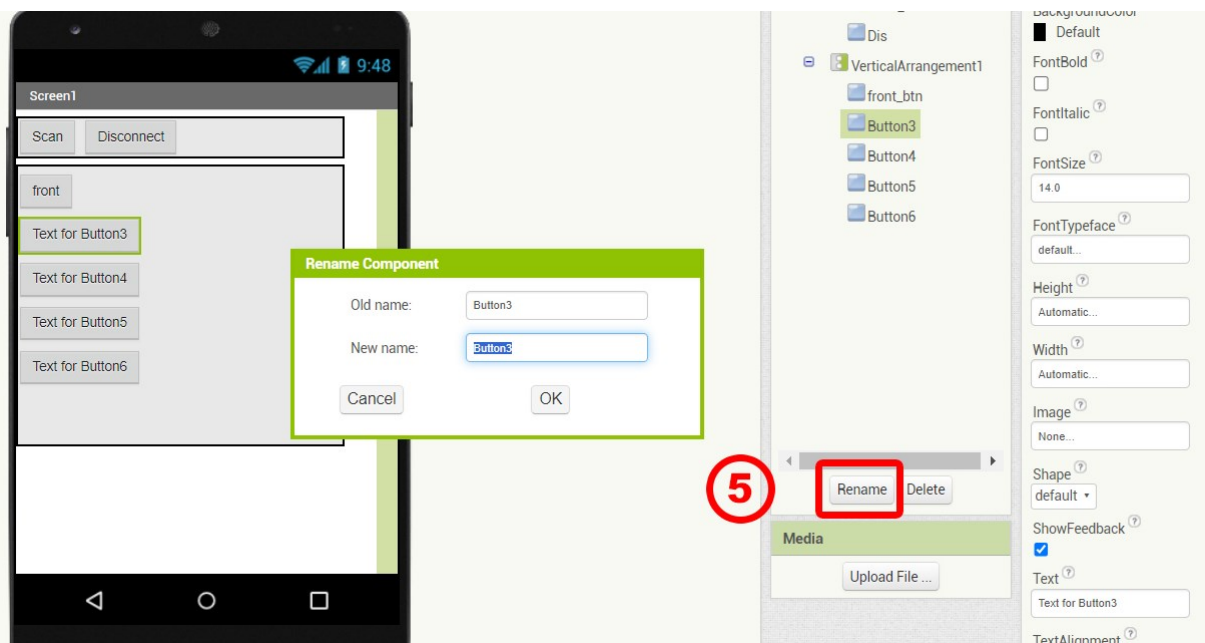


Figura 6: Renomear o componente Button

Sugestão: Os nomes dos botões são indicativos e não afetam a sua funcionalidade. No entanto, uma boa prática é utilizar nomes que sejam significativos para todo o processo (por exemplo, dar o nome "front_btn" ao botão que vai fazer avançar o carro, "back_btn" ao que vai fazer recuar o carro, etc.). No nosso exemplo, estamos a mudar o nome dos componentes da seguinte forma:

Componente	Nome do texto (alterado a partir do separador Texto nas propriedades)	Novo nome (alterado de Renomear na lista de componentes)
ListPicker	Digitalização	Lista de verificação
Botão1	Desligar	Dis
Botão2	frente	front_btn
Botão 3	voltar	back_btn
Botão 4	correto	botão direito
Botão5	esquerda	left_btn
Botão6	paragem	stop_btn

Nota importante: não utilizar a mesma palavra para o nome do texto e para o nome do botão, pois isso causará um mau funcionamento de App Inventor, impedindo-o de construir o aplicativo.

Resumindo, o ListPicker "Scan" irá procurar dispositivos Bluetooth disponíveis, enquanto o botão "Disconnect" irá parar a ligação entre o nosso dispositivo inteligente e o módulo Bluetooth. Os restantes botões serão utilizados para navegar no nosso carro robótico.

Sugestão: Tenha em conta que as diretrizes acima mencionadas são indicativas. Pode experimentar diferentes formas e cores para cada botão, ou mesmo diferentes disposições, criando assim uma interface mais única e visualmente apelativa.

Adicionar uma notificação de texto

O passo seguinte é adicionar uma etiqueta de texto ao nosso ecrã. Este passo não é obrigatório, mas é muito útil, uma vez que nos informará se a ligação com o nosso dispositivo inteligente está estabelecida ou não. Procure "Label" (Etiqueta) no submenu "User Interface" (Interface do utilizador) e arraste e largue-a no ecrã (*Figura 7*). Altere o conteúdo do texto para "Waiting for input" (ou para algo semelhante da sua preferência, da mesma forma que alterou o conteúdo do texto para os restantes componentes).

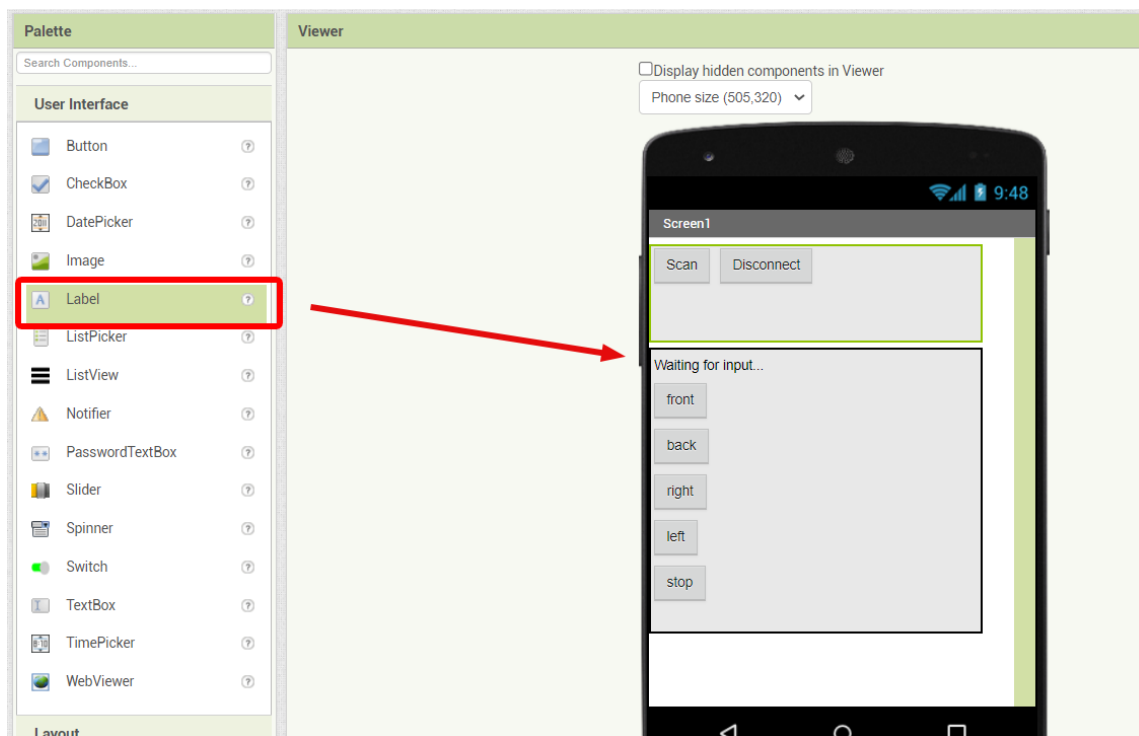


Figura 7: Adição de uma etiqueta para fins informativos sobre a conectividade

Adicionar extensões

O próximo passo é adicionar alguns componentes que permitirão a ligação entre a nossa aplicação e o carro robótico. Para isso, temos de adicionar as seguintes extensões à nossa aplicação: a extensão BluetoothLE e a Microbit_Uart_Simple. A primeira é usada para estabelecer a ligação Bluetooth entre o nosso dispositivo inteligente e o Micro:bit, enquanto a segunda é usada para enviar as mensagens apropriadas depois de a ligação ter sido estabelecida.

Para poder utilizar estas extensões, é necessário descarregá-las localmente para o seu computador. Para o fazer, clique aqui <https://mit-cml.github.io/extensions/> e descarregue para o seu computador o ficheiro *BluetoothLE.aix* e o ficheiro *Microbit.aix* (Figura 8).

MIT APP INVENTOR

Home Directory Documentation

Supported:

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

Figura 8: Encontrar as extensões a descarregar

Depois de descarregar as extensões, volte a App Inventor. Na secção Paleta, clique no separador Extensão e, em seguida, clique na seleção Importar extensão (6) (Figura 9)

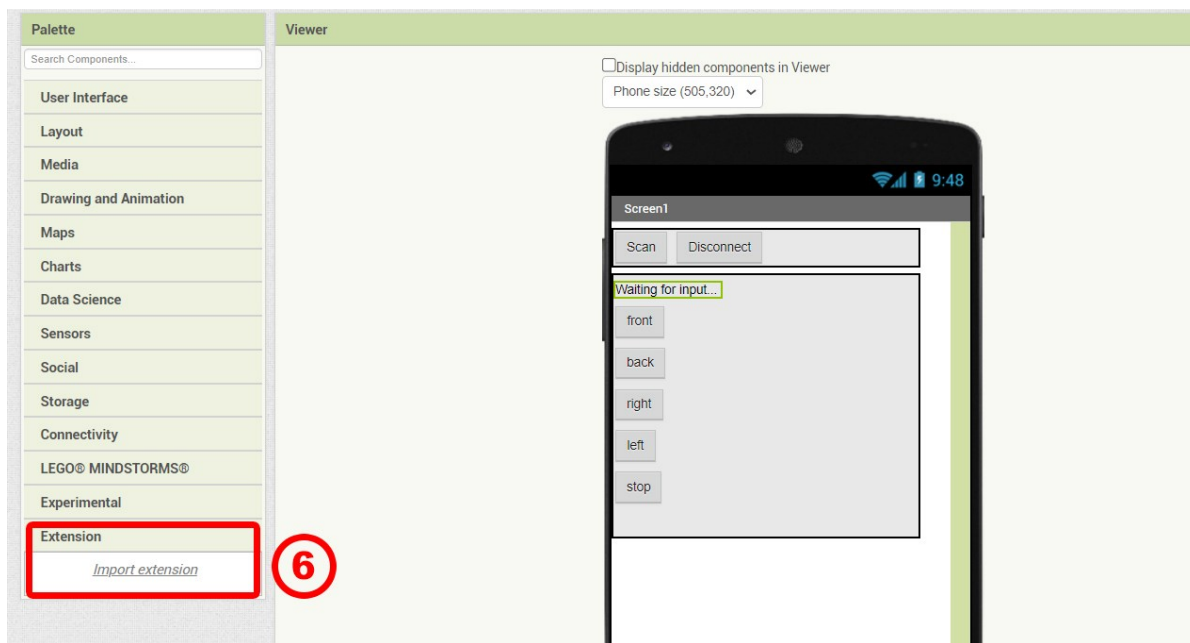


Figura 9: A seleção da extensão Importar

No menu instantâneo, clique no botão Escolher ficheiro (7) para procurar na sua pasta local e seleccionar a extensão descarregada (Figura 10). Certifique-se de que a opção "From my computer" (Do meu computador), localizada acima do botão Choose File (Escolher ficheiro), está seleccionada.

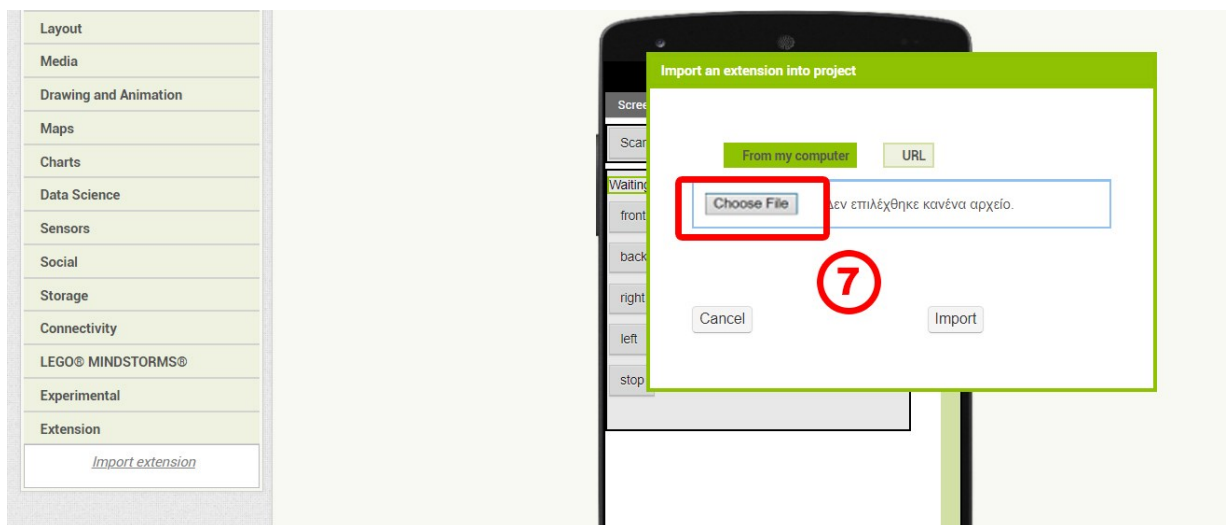


Figura 10: Clique no botão Escolher ficheiro para procurar o ficheiro de extensão na sua pasta local

Depois de encontrar e seleccionar o ficheiro de extensão, clique no botão Importar (8) (Figura 11).

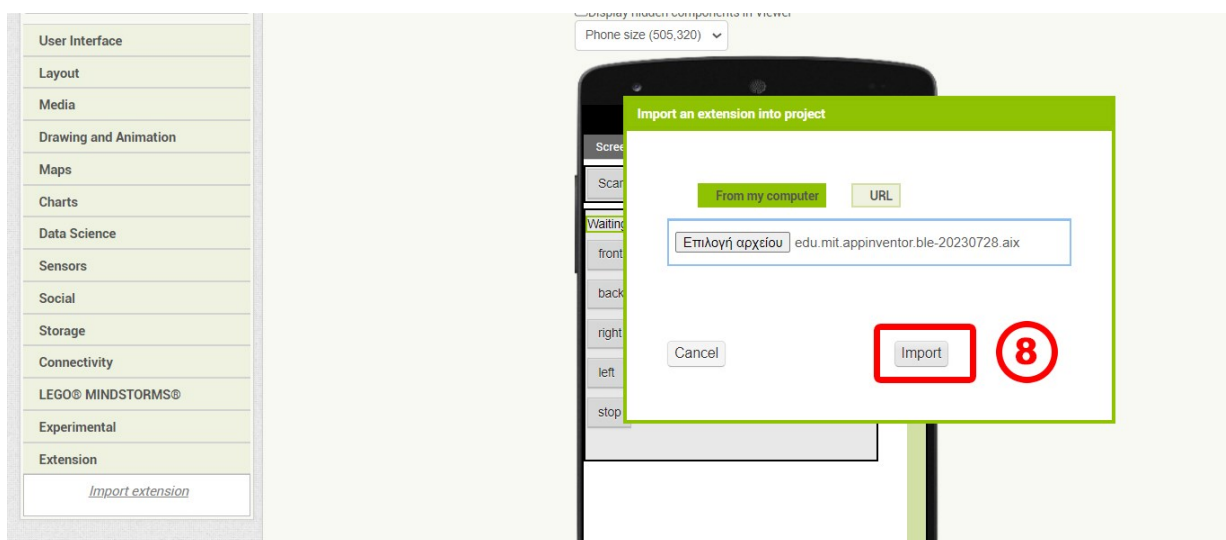


Figura 11: Clique no botão Importar para importar a extensão seleccionada

Passado algum tempo, a nova extensão aparecerá na secção Importar extensão, no separador Extensão (por exemplo, na Figura 12, a extensão BluetoothLE foi importada)



Figura 12: A extensão BluetoothLE foi importada

Para adicionar esta extensão à aplicação concebida, arraste-a e largue-a na área de conceção (Figura 13). As extensões são normalmente componentes não visíveis. Por conseguinte, estes componentes aparecem sob a área de desenho, na secção "Componentes não visíveis".

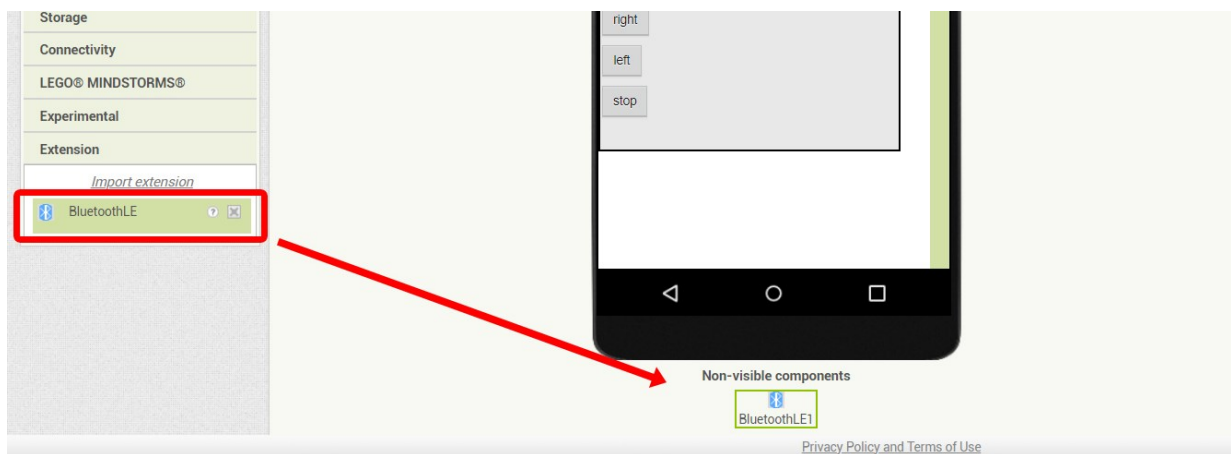


Figura 13: Arrastar e largar a extensão BluetoothLE na área de desenho

Repita o mesmo processo para importar a extensão Microbit_Uart_Simple.

Nota importante: depois de escolher e importar o ficheiro *Microbit.aix*, irá reparar que aparecerão várias extensões no separador Extension. Para efeitos desta atividade, apenas terá de arrastar e largar na área de desenho a extensão Microbit_Uart_Simple (Figura 14).

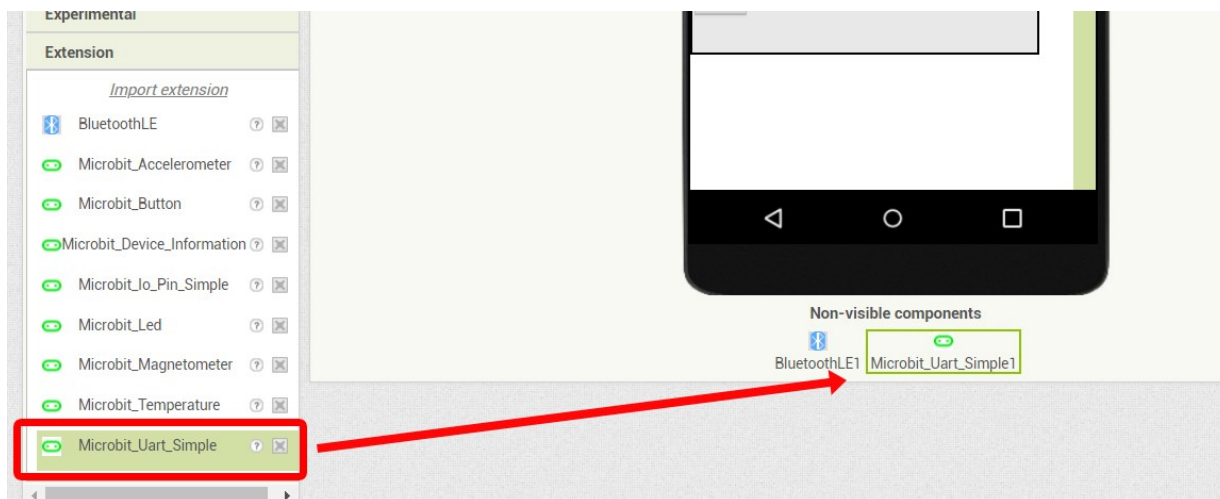
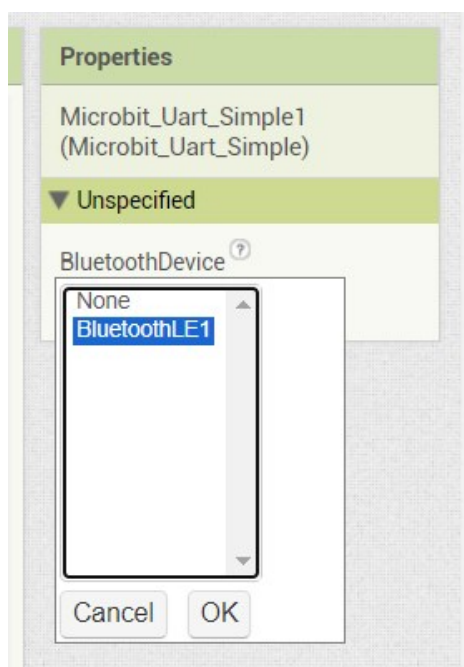


Figura 14: Só é necessário arrastar e largar o Microbit_Uart_Simple

Depois de importar a extensão Microbit_Uart_Simple, vá ao menu Properties (Propriedades) e, no campo Bluetooth device (Dispositivo Bluetooth), selecione BluetoothLE1 no menu flutuante.



Depois de adicionar todas as extensões necessárias, estamos prontos para passar à programação da aplicação.

1.4 Programação da aplicação

Para que a nossa aplicação funcione, temos de determinar o funcionamento dos componentes adicionados, especificando as ações que irão desencadear quando forem premidos (no ecrã da nossa aplicação). Para o fazer, temos de ir ao menu Blocks (Blocos) e criar o nosso código no espaço de visualização (*Figura 21*), arrastando e montando os blocos de comandos adequados.

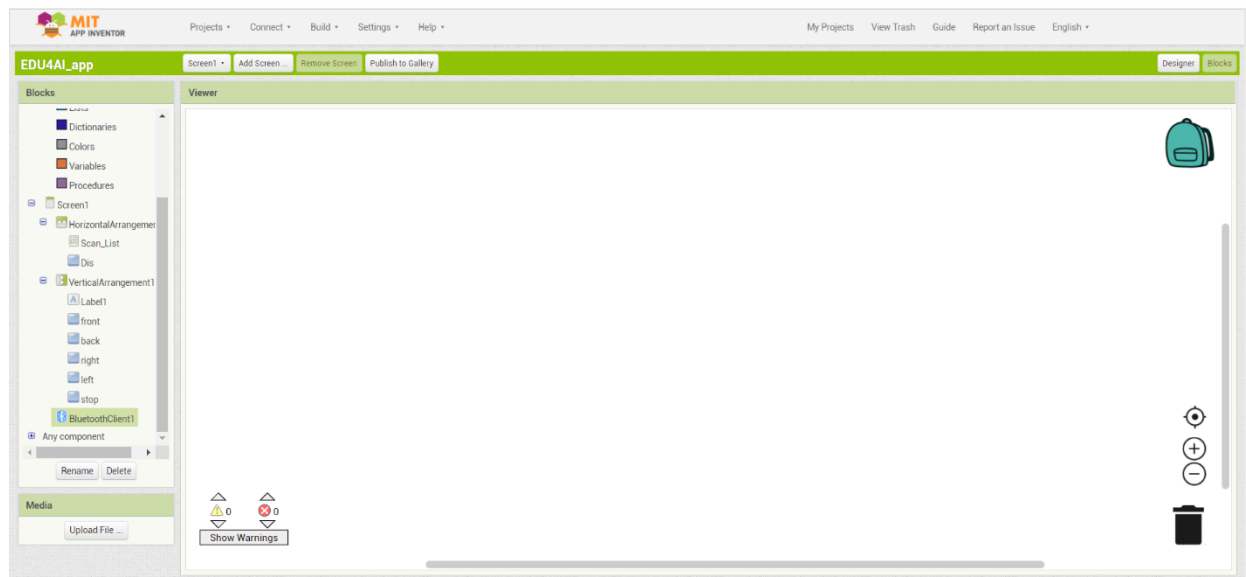


Figura 15: Menu de blocos do App Inventor

Codificação do componente ListPicker (i.e., Scan_List)

Começaremos por codificar o componente ListPicker, ou o componente "Scan_List", como é designado no nosso exemplo. O diagrama seguinte mostra as operações/acções que queremos que sejam executadas quando o componente ListPicker é premido, e o comando de **evento** correspondente que deve ser aplicado para este fim. Vejamos então como é que estes códigos serão estruturados.



Comando BeforePicking

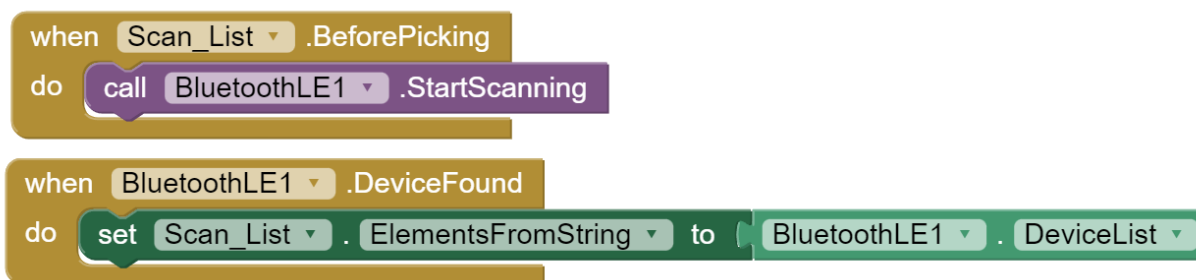
Clique no item "Scan_List" (1) e, no menu flutuante, seleccione o comando de evento "When Scan_List BeforePicking" (2) (Figura 16).

Quando Scan_List é premido, deve aparecer uma lista dos dispositivos Bluetooth disponíveis. "BeforePicking" significa que ainda não seleccionámos nenhum dos dispositivos Bluetooth apresentados.



Figura 16: Clicar no item Lista de digitalização e seleccionar o comando necessário no menu flutuante

Para podermos seleccionar elementos da nossa lista, esta deve ser inicialmente preenchida com todos os dispositivos Bluetooth disponíveis. (A lista está vazia no início, por isso, antes de escolhermos algo, tem de ser preenchida primeiro). Para o fazer, temos de começar a procurar dispositivos BLE. De seguida, a lista é preenchida com todos os dispositivos disponíveis que forem encontrados:



Nota: Os blocos de comandos utilizados no guião acima podem ser encontrados de forma semelhante, clicando no item correspondente (Scan_List ou BluetoothLE1) e encontrando os blocos relevantes no menu flutuante correspondente (alguns exemplos são apresentados na Figura 17).

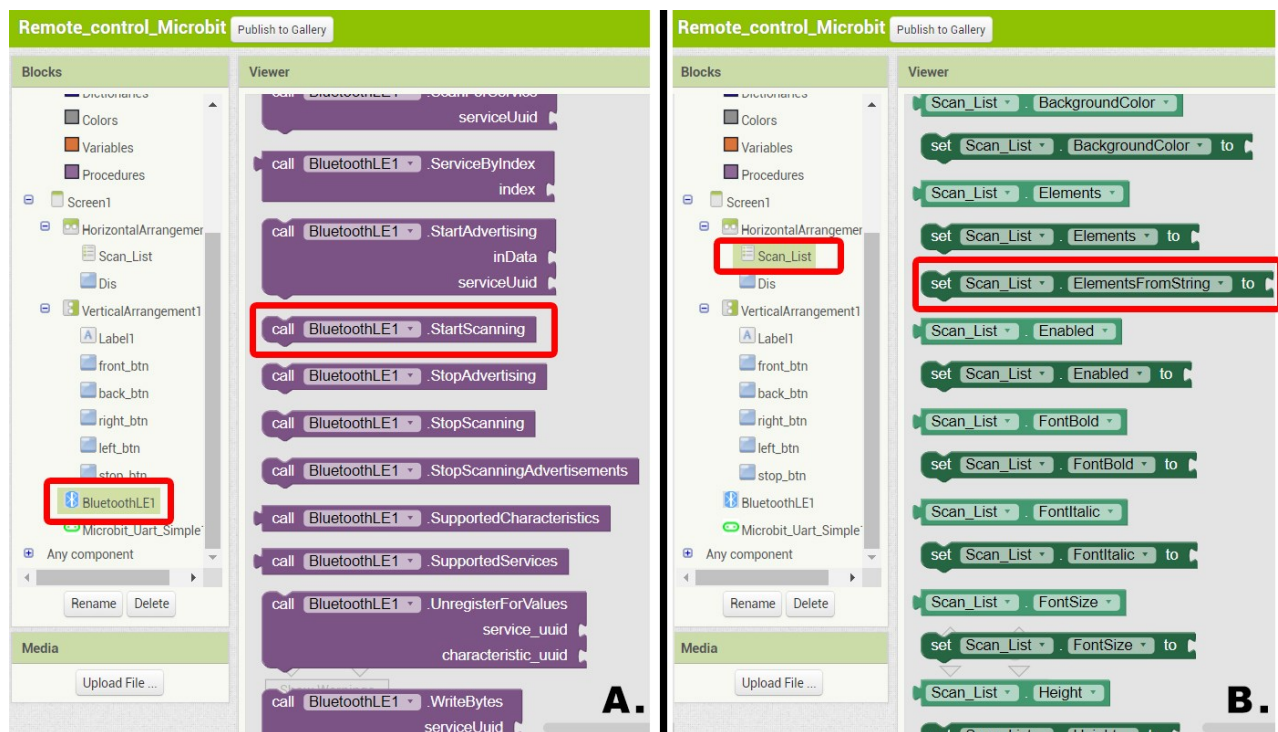
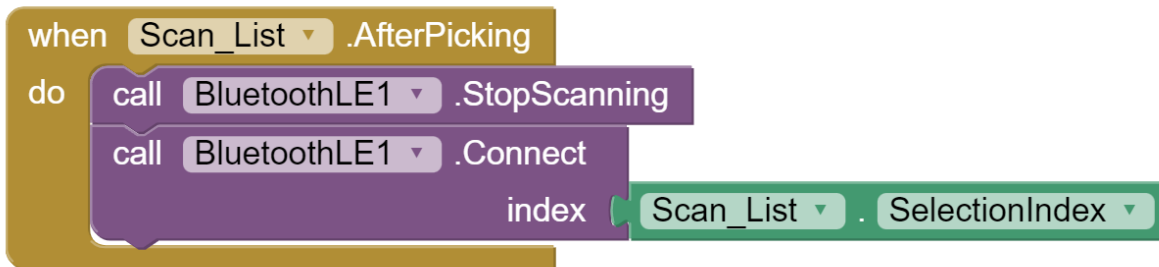


Figura 17: Encontrar alguns dos comandos de bloco

Comando **AfterPicking**

Agora, devemos determinar o que acontece quando o utilizador seleciona o endereço Bluetooth do micro:bit para estabelecer a ligação entre a aplicação e o carro robótico. Para isso, vamos precisar do comando de evento **"When Scan_List AfterPicking"**.

Dentro do **comando do evento**, colocaremos alguns comandos que determinarão a ação que será executada depois de termos escolhido um endereço Bluetooth da lista. Assim, vamos utilizar o comando **"call BluetoothLE1.StopScanning"**, para que o nosso dispositivo pare de procurar outros dispositivos Bluetooth LE. Em seguida, adicionamos o comando **"call BluetoothLE1.Connect"** e, no seu lado direito, encaixamos o comando **"Scan_List.SelectionIndex"**, para permitir que a aplicação se ligue ao dispositivo Bluetooth selecionado.



Codificação da etiqueta (ou seja, etiqueta1)

Para nos certificarmos de que a ligação Bluetooth foi estabelecida, vamos programar a aplicação para enviar uma notificação relevante. Para o efeito, programamos o item Label e utilizamos os seguintes blocos de código:



Em particular, utilizamos o comando "set Label1.Text to" e, no seu lado direito, encaixamos um bloco de texto "", no qual escrevemos "Connection Established" (Ligação estabelecida).

Se a ligação for bem sucedida, a notificação "Connection Established" (Ligação estabelecida) aparecerá no ecrã, substituindo a mensagem "Waiting for input..." (À espera de entrada...) (Figura 7). Se a ligação não for estabelecida com êxito, a mensagem "Waiting for input..." não se altera.

Assim, após a seleção do dispositivo Bluetooth (nomeadamente o Bluetooth do micro:bit), o componente BluetoothLE tentará ligar-se ao endereço preferido (i.e., o micro:bit que o carro robótico está a utilizar). Se a tentativa for bem sucedida, o texto da etiqueta "Waiting for input..." muda para "Connection established!".

Nota: o bloco de introdução de texto encontra-se no menu flutuante do separador "Texto" (Figura 18).

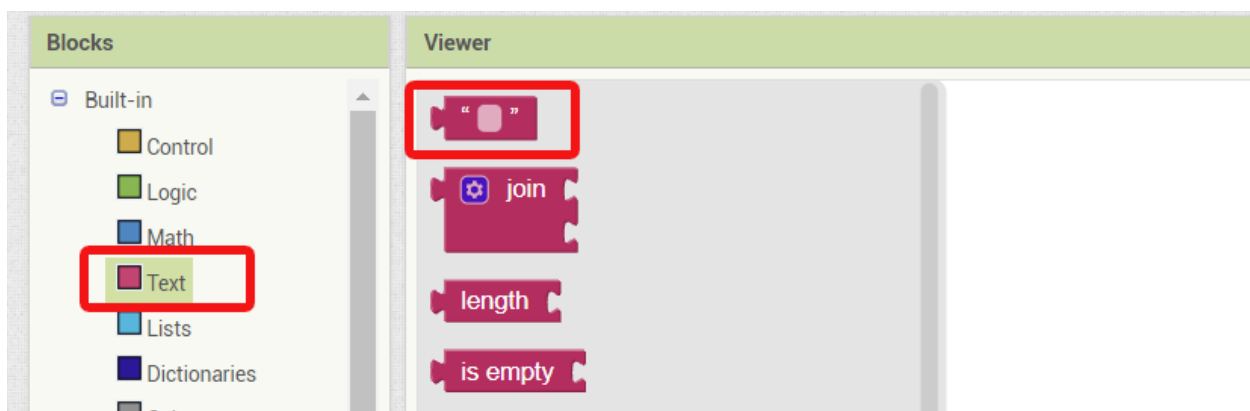


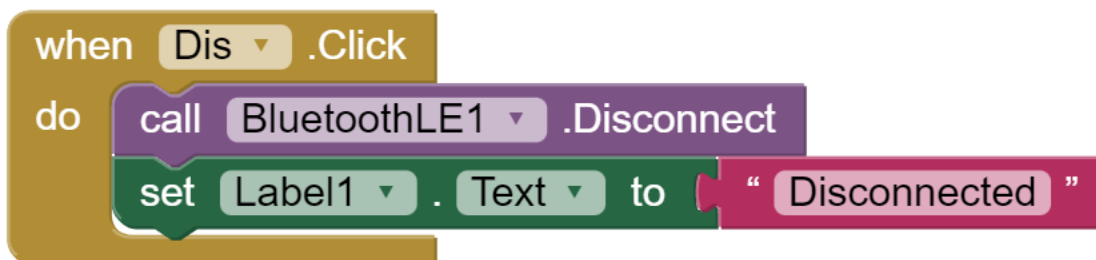
Figura 18: Encontrar o bloco de introdução de texto

Codificação do botão Desconectar (ou seja, Dis)

Neste passo, vamos programar o componente BluetoothLE para terminar a ligação, quando o botão "Disconnect" (Desligar) for premido. O texto da etiqueta também será alterado para "Connection timed out".

Para este passo de codificação, vamos precisar do comando de evento **"When Dis .Click...do"**.

Assim, através do seguinte script, **quando** o botão "Disconnect" (Desligar) é premido, a aplicação é instruída para **chamar** **BluetoothLE** e desligá-lo do dispositivo pré-selecionado. Além disso, definimos o **texto da etiqueta** como **"Disconnected" (Desligado)**, de modo a informar o utilizador de que a ligação foi terminada.



Nota: O guião para ativar os botões "Scan" e "Disconnect" pode ser difícil para os seus alunos. Por isso, dependendo do nível dos alunos, pode fornecer-lhes estas partes do guião e ensinar-lhes minuciosamente a parte seguinte do guião.

Codificar os botões para navegar no carro robótico

O último passo é programar os botões de navegação. Como mencionado anteriormente neste documento, quando um botão de navegação é premido, o nosso dispositivo inteligente transmite (via Bluetooth) uma mensagem específica à placa micro:bit do nosso carro robótico. Quando esta mensagem é recebida, o carro robótico comporta-se em conformidade, efetuando um movimento específico (ou seja, avançar, recuar, etc.). A tabela seguinte apresenta a mensagem que é transmitida quando um botão específico é premido:

<i>Botão que é premido</i>	<i>Palavra que é enviada</i>
"frente"	"#forward#"
"costas"	"#backwards#"
"direito"	"#direita#"
"esquerda"	"#esquerda#"
"parar"	"#stop#"

Nota importante: Cada mensagem que queremos transmitir deve começar e terminar com o símbolo "#", para que o micro:bit possa distinguir os limites da mensagem.

Por exemplo, quando transmitimos a palavra "#forward#", o micro:bit faz girar os motores DC para a frente.

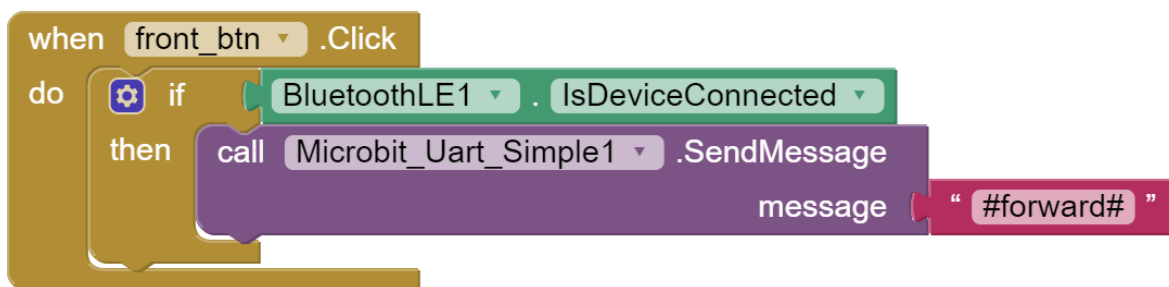
Para esta parte do script, e para cada um dos botões, vamos precisar do comando de evento "**When .Click...do**" e de uma condição "**if..then**".

Nota: A condição do bloco "se-então" pode ser encontrada no separador "Controlo" (Figura 19).



Figura 19: Encontrar a condição do bloco "se...então"

Assim, através do seguinte bloco de comandos damos instruções à nossa aplicação para verificar se o componente **BluetoothLE se ligou ao dispositivo** desejado e, **se o fizer**, a **mensagem** correspondente ("**#forward#**" neste exemplo) é **enviada chamando** o componente **Microbit_Uart_Simple**.



Nota: o valor "forward", escrito no interior do texto, é um valor declarado no guião criado no ambiente de programação MakeCode e atribuído ao movimento "moving forward".

Repita o mesmo processo para os outros quatro botões. O resultado deve ser semelhante ao apresentado na Figura 20.

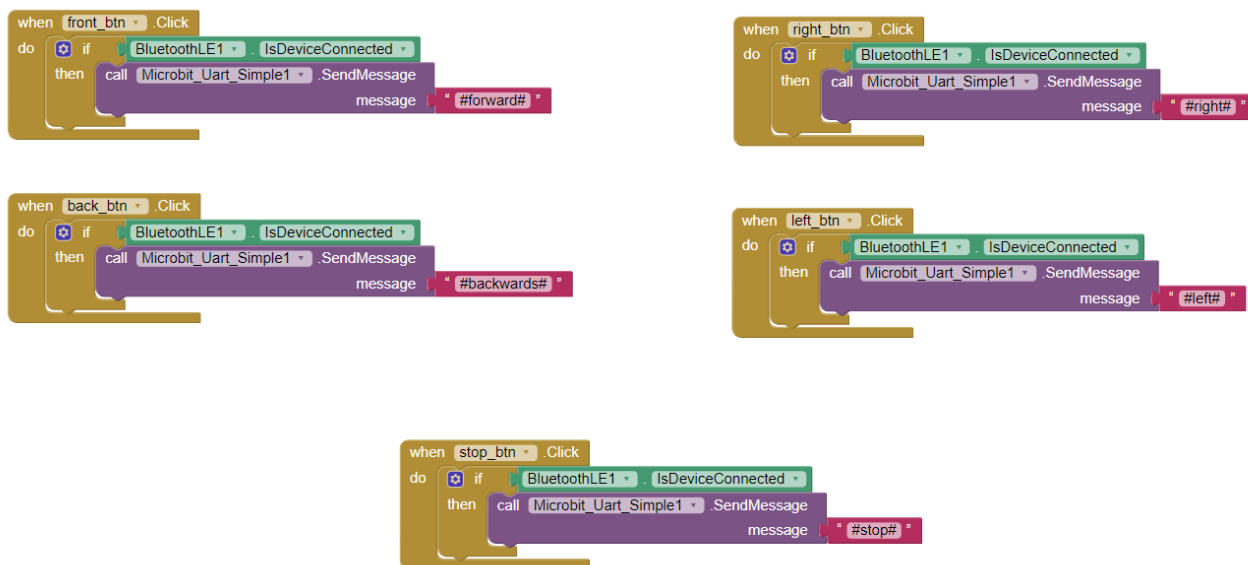


Figura 20: Criar os guiões para todos os botões da aplicação

Quando terminar todos os passos acima mencionados, o guião completo deve ter o aspeto ilustrado na Figura 21.

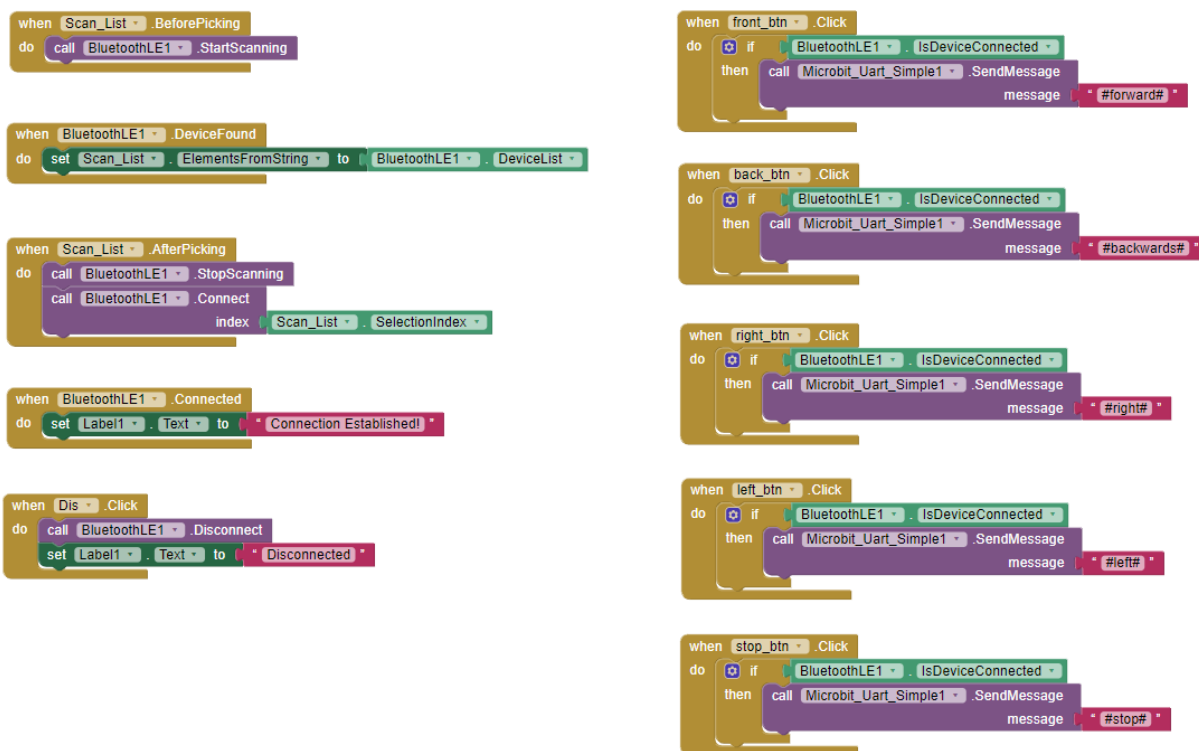


Figura 21: O guião completo para esta atividade de aquecimento

Agora que a aplicação está pronta, pode criar a aplicação e carregá-la no seu dispositivo inteligente.

1.5 Criar a aplicação

Quando terminar todos os passos acima mencionados (e todo o código for semelhante ao apresentado na *Figura 21*), a aplicação está pronta para ser carregada e instalada no seu dispositivo inteligente.

Assim, vá ao *menu Build (1)* (*Figura 22a*) e selecione "Android App (.apk)" no menu suspenso para iniciar o processo de produção do ficheiro .apk. Isto pode demorar alguns minutos.

Quando o processo de construção estiver concluído, aparecerá uma nova janela **(2)** (*Figura 22a*). O utilizador pode optar por descarregar o ficheiro .apk produzido ou pode digitalizar com o seu dispositivo inteligente o código QR incorporado, através da **aplicação MIT AI2 Companion** (*Figura 22b*), que deverá ter previamente descarregado e instalado no seu dispositivo inteligente.

Nota 1: O MIT AI2 Companion é uma aplicação/serviço que está disponível (gratuitamente) e pode ser encontrada no serviço "Play Store" do seu dispositivo inteligente. Esta aplicação funciona como um mediador, facilitando a instalação bem sucedida do ficheiro .apk produzido no seu dispositivo inteligente. Esta aplicação só está disponível para dispositivos Android.

Nota 2: Durante a instalação do ficheiro .apk, podem aparecer várias notificações relativas à segurança deste ficheiro. Ignore-as todas e peça ao seu dispositivo para prosseguir com o processo de instalação.

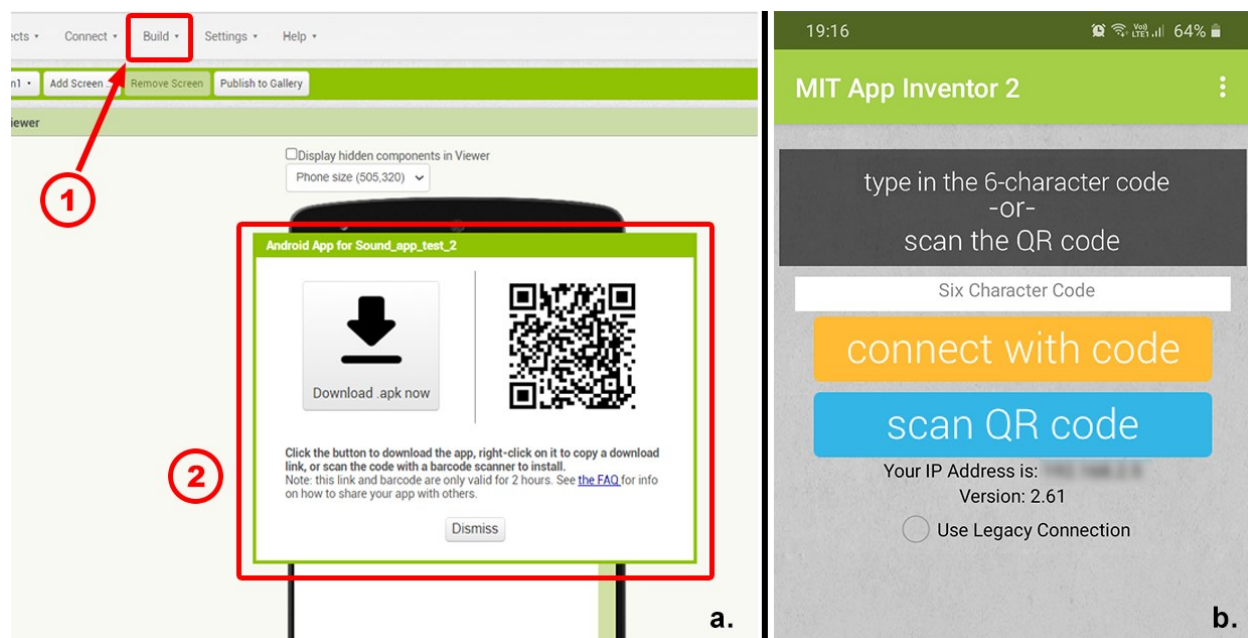


Figura 22a. Código QR produzido após a conclusão do processo de construção; b. Captura de ecrã da interface do MIT AI2 Companion

Depois de instalar com êxito a aplicação no seu dispositivo inteligente, pode testar se funciona corretamente. **Lembre-se de que** precisa de ter descarregado para o seu carro robótico o script produzido no ambiente de programação Makecode.

1.6 Emparelhar a aplicação com o veículo robótico

Como mencionado acima, para ligar a aplicação ao carro robótico, é necessário premir o botão "Scan" e seleccionar o endereço Bluetooth do micro:bit na lista. No entanto, é provável que o micro:bit não esteja incluído na lista de dispositivos Bluetooth disponíveis. Para resolver este problema, volte ao menu principal da aplicação e **active** o modo "avião" no seu dispositivo inteligente durante alguns segundos. Depois desative-o e prima novamente o botão Scan. O endereço Bluetooth do Micro:bit estará agora disponível.

Nota: Certifique-se de que a "Localização" também está ativada no seu dispositivo inteligente