

Folha de actividades:

Título: Criar um jogo de labirinto com o BBC Micro:bit

Objetivo:

- Compreender o conceito de perceção em IA, utilizando o acelerómetro do Micro:bit para controlar uma personagem de jogo num labirinto.
- Desenvolver competências de programação para criar um jogo interativo.

Materiais:

- BBC Micro:bit com cabo USB.
- Computador com o ambiente de codificação MakeCode instalado.

Instruções:

Etapa 1: Introdução

- Ligue o Micro:bit ao seu computador através de USB.
- Abra o ambiente de codificação MakeCode num navegador da Web.

Passo 2: Criar um novo projeto

- Inicie um novo projeto MakeCode para o seu jogo de labirinto.

Etapa 3: Compreender a perceção

- A perceção em IA envolve sentir e compreender o ambiente. Nesta atividade, vai utilizar o acelerómetro do Micro:bit para detetar movimentos de inclinação, permitindo que a personagem do jogo se mova dentro do labirinto.

Passo 4: Conceber e personalizar o seu labirinto

Neste passo, vai desenhar o labirinto utilizando a grelha fornecida no ambiente de codificação MakeCode. Pode personalizar o labirinto adicionando paredes e caminhos abertos para criar um puzzle desafiante. O labirinto deve ter um ponto de partida e um destino claros, que é o ponto final do jogo. Mantenha-o desafiante: Considere o nível de dificuldade do seu labirinto. O caminho desde o início até ao fim deve representar um desafio para o jogador. O jogador deve navegar pelo labirinto inclinando o Micro:bit e evitando as paredes para chegar ao destino.

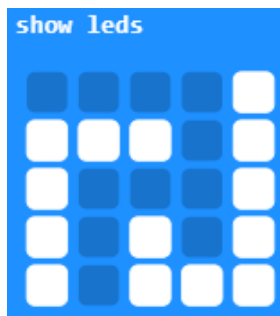


Figura 1 Grelha de LEDs utilizando o bloco "show led" em MakeCode

Cria a disposição do labirinto utilizando a grelha fornecida no MakeCode. Utiliza as formas dos blocos para representar as paredes, os caminhos abertos, o início ($x = 0, y = 0$) e o fim ($x = 1, y = 4$) do labirinto. Personaliza o esquema para corresponder ao labirinto fornecido ou cria o teu próprio desenho de labirinto. Também podes criar mais níveis para o jogador.

*****Nota*****

O conjunto completo de coordenadas **x,y** para a grelha que o micro:bit oferece é apresentado na tabela abaixo.

Tabela 1 Coordenadas X, Y para a grelha micro:bit

(0,0)	(1,0)	(2,0)	(3,0)	(4,0)
(0,1)	(1,1)	(2,1)	(3,1)	(4,1)
(0,2)	(1,2)	(2,2)	(3,2)	(4,2)
(0,3)	(1,3)	(2,3)	(3,3)	(4,3)
(0,4)	(1,4)	(2,4)	(3,4)	(4,4)

A localização do jogador no ecrã do Micro:bit será indicada por um LED vermelho intermitente. Os LEDs vermelhos sólidos simbolizam as paredes, enquanto os LEDs apagados representam os caminhos do labirinto.

As coordenadas são utilizadas para manipular eficazmente os LEDs do Micro:bit! As coordenadas x vão de 0 à esquerda a 4 à direita, enquanto as coordenadas y vão de 0 em cima a 4 em baixo. Consequentemente, o LED no canto superior esquerdo é denotado como $x=0, y=0$, e correspondentemente, o LED no canto inferior direito é representado como $x=4, y=4$.

Passo 5: Codificar o jogo

Utiliza os blocos seguintes para programar o comportamento do jogo:



Figura 2 Início do programa em MakeCode

Primeiro, precisamos de criar algumas variáveis. Lembre-se de que as variáveis funcionam como contentores que armazenam informações. Neste caso, são necessárias duas variáveis para monitorizar a localização do jogador. Uma é designada para registar a posição x do jogador, enquanto a outra é dedicada a seguir a posição y do jogador.

Além disso, é necessária uma variável para monitorizar o nível do labirinto, permitindo a possibilidade de vários níveis. É necessária outra variável para monitorizar o estado do jogo, indicando se está ativo ou se terminou.

Os valores iniciais são definidos para começar no nível 1, e gameOn é inicializado como True. Isto porque, ao ligar o Micro:bit, a intenção é começar o jogo imediatamente. Embora o ponto de partida para a localização do jogador possa ser escolhido arbitrariamente, ele precisa de ser recordado mais tarde quando se configura o nível do labirinto para garantir que o jogador não começa dentro de uma parede. Para este exemplo, o jogador é iniciado em x=0 e y=0.

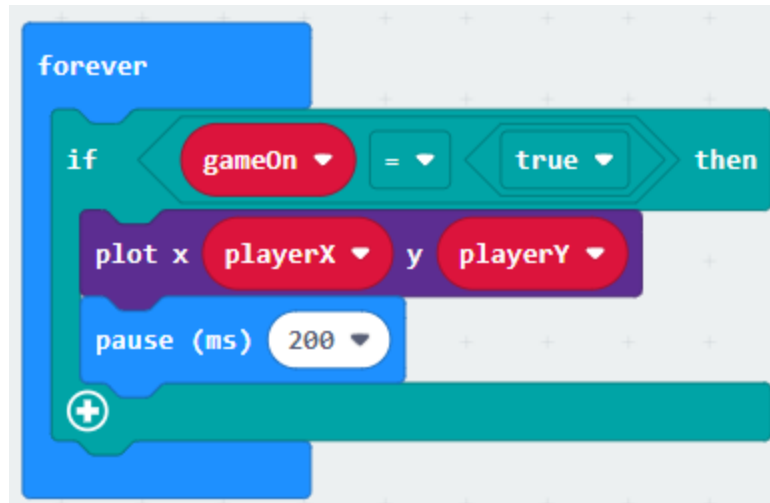


Figura 3 Primeiro loop infinito

Agora que as variáveis iniciais estão no lugar, vamos garantir que o nosso jogador é mostrado no ecrã do Micro:bit!

Para conseguir um efeito de piscar distinto para o jogador, vamos utilizar o bloco "plot x y" alternado com o bloco "pause" dentro de um loop infinito. A intenção é que o jogador fique a piscar continuamente. Quando as paredes do labirinto forem introduzidas, o Micro:bit irá sobrescrever o jogador cada vez que desenhar as paredes. Ao incorporar um bloco de pausa aqui, garantimos que o jogador não será imediatamente reposicionado, resultando no efeito de piscar desejado.

A utilização das variáveis `playerX` e `playerY` criadas anteriormente é crucial. Porquê? Se fossem introduzidos valores numéricos diretamente aqui, isso limitaria a flexibilidade de fazer o jogador mover-se. A utilização de variáveis permite-nos modificar os valores de `playerX` e `playerY`, permitindo que o ciclo para sempre trace a nova localização do jogador.

É essencial ter em atenção que o bloco de pausa funciona em milissegundos (por exemplo, 200 ms = 0,2 segundos) e a velocidade de intermitência pode ser personalizada ajustando a duração da pausa.

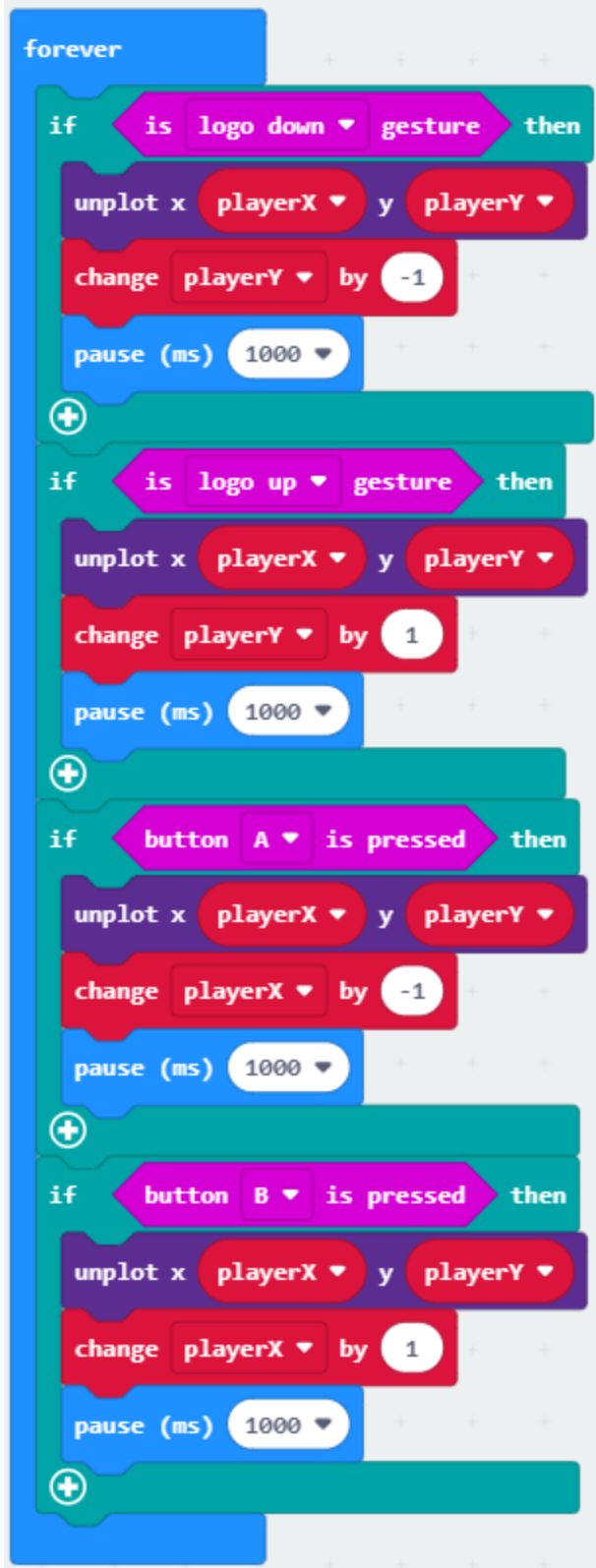


Figura 4 Segundo loop infinito

Agora precisamos de configurar os movimentos do jogador (esquerda, direita, cima e baixo). Vamos utilizar os dois botões integrados e a função de deslizar o logótipo.

Vamos definir o gesto do logótipo para cima para mover para cima, o gesto do logótipo para baixo para mover para baixo, o botão A para mover para a esquerda e o botão B para mover para a direita.

Para isso, utilizamos instruções if. Essas instruções avaliam se uma condição é verdadeira; se for, todos os blocos dentro do bloco if são executados. Quando incorporamos uma instrução if num ciclo para sempre, verificamos continuamente se a condição é verdadeira.

Para o movimento do jogador, modificamos as variáveis playerX ou playerY. É crucial lembrar que diminuir ou aumentar o jogadorX causa um movimento para a esquerda ou para a direita, respetivamente, enquanto diminuir ou aumentar o jogadorY resulta num movimento para cima ou para baixo, respetivamente. Dado que traçamos consistentemente a localização do jogador utilizando estas variáveis, quaisquer alterações reflectem automaticamente a nova posição do jogador.

Vale a pena notar que uma breve pausa de 300ms é adicionada após cada pressão de botão. Isto evita que o Micro:bit mova o jogador através de múltiplos espaços rapidamente com cada pressão de botão, uma vez que o código corre rapidamente sem a pausa.



Figura 5 Terceiro loop infinito

Continuamos com a criação do nível do labirinto. Várias tarefas requerem atenção: em primeiro lugar, mostrar as paredes do labirinto no ecrã LED; em segundo lugar, verificar continuamente se o jogador colide com uma parede (indicando o fim do jogo); e, em terceiro lugar, avaliar perpetuamente se o jogador completa com sucesso o nível do labirinto.

É utilizado um ciclo para sempre. Dentro deste ciclo, é utilizada uma instrução "if" para verificar se a variável de nível é igual a 1. Consequentemente, este segmento de código só será executado quando a variável de nível for igual a 1. Se quisermos adicionar mais níveis, devemos certificar-nos de que esta variável muda em conformidade.

Dentro da instrução 'if', as paredes do labirinto são apresentadas utilizando o bloco 'show leds'. Os LEDs são iluminados para representar as paredes, enquanto os LEDs apagados indicam os caminhos do labirinto. É necessário ter cuidado para garantir que a posição inicial do jogador, definida anteriormente em x=0, y=0, não coincide com uma parede do labirinto.

A tarefa subsequente consiste em verificar se o jogador colide com uma parede. Isto é conseguido através de instruções "if"

adicionais, que verificam se as variáveis playerX e playerY estão alinhadas com as coordenadas de uma parede na grelha de LEDs 5x5.

Por fim, o código verifica se o jogador navega com sucesso pelo labirinto. Neste exemplo, o fim do labirinto está em x=1, y=4. Se estas condições forem cumpridas, toca uma melodia de sucesso, a posição do jogador é reposta no início do labirinto e aparece uma cara sorridente no Micro:bit. Se tivermos acrescentado níveis adicionais, também precisamos de alterar a variável nível em 1.

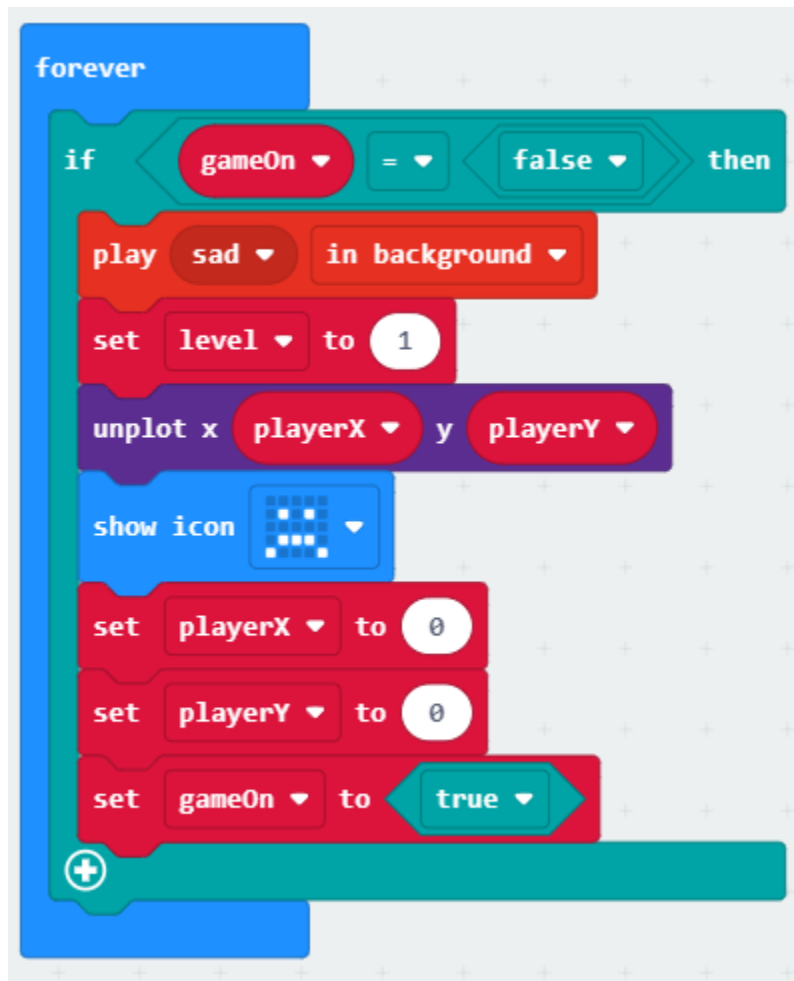


Figura 6 Quarto loop infinito

No caso de um game over, temos de implementar uma ação desencadeada pela variável "gameOn" que indica uma colisão com uma parede.

Dentro de um loop infinito, é utilizada uma instrução 'if' para avaliar o valor da variável 'gameOn'. Se for igual a 'false', o código de fim de jogo é executado.

Neste caso, uma melodia triste é reproduzida em segundo plano, o "nível" é reiniciado, o LED do jogador é apagado, é apresentada uma cara triste e o jogo começa desde o início.

Passo 6: Teste o seu jogo

- Teste o seu jogo guiando a personagem através do labirinto. Funciona tudo corretamente?

Passo 7: Jogar e partilhar

- Partilhe o seu jogo de labirinto com outros. Carregue-o no seu Micro:bit e desafie os seus amigos a completar o labirinto.

Este projeto permite aos alunos experimentar o conceito de perceção em IA, utilizando o acelerómetro do Micro:bit para controlar uma personagem dentro de um labirinto. Os alunos podem conceber os seus próprios labirintos, criar diferentes níveis de dificuldade e partilhar os seus jogos com os colegas para maior diversão e aprendizagem.