Sorteador de número via Bluetooth.

Danielli Freitas

Giovana Oliveira

Julia Daniluski

Lais Boscolo

Rayanne Picini

Sorteador de número via Bluetooth: Programando Arduino

Objetivo:

Esse código serve para o Arduino receber comandos via Bluetooth e executar ações baseadas nesses comandos. Nesse caso, um sorteador de números com os 7 segmentos onde é enviado o comando ‘S’, a partir de um aplicativo feito pelo MIT App Creator, para o Arduino sortear um número de 0 a 9. É um sistema simples para sortear e mostrar números controlado por Bluetooth, permitindo interação entre um app e o Arduino.

Sorteador de número via Bluetooth: Programando Arduino

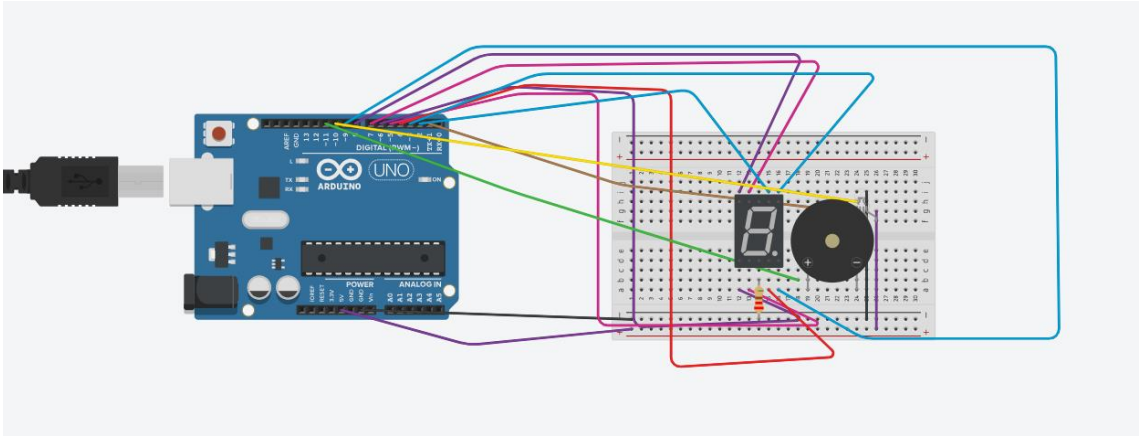
Material necessário:

- 1 Arduino;
- 1 Resistor (220 ohm);
- 1 Visor de 7 segmentos (catódico);
- 1 Protoboard;
- Jumpers cables;
- 1 led RGB (para representar o conector bluetooth).

Sorteador de número via Bluetooth: Programando Arduino

Montagem no Tinkercad:

A montagem mostra um display de 7 segmentos conectado a um Arduino Uno através de uma protoboard. Cada segmento do display (a, b, c, d, e, f, g e o ponto) está ligado aos pinos digitais do Arduino. O pino comum do display vai para o GND com resistor.



Sorteador de número via Bluetooth: Programando Arduino

Código:

Primeiramente o código define que o buzzer (dispositivo que emite a musica) está conectado no pino digital 11 do Arduino.

E logo depois são definidas as frequências em Hz de notas musicais. Essas frequências serão usadas pelo tone(BUZZER, frequência) para tocar a melodia.

```
// --- Buzzer ---
#define BUZZER 11

// Frequência das notas (em Hz)
#define NOTE_F3  175
#define NOTE_C4  262
#define NOTE_D4  294
#define NOTE_E4  330
#define NOTE_F4  349
#define NOTE_FS4 370
#define NOTE_G4  392
#define NOTE_A4  440

// Melodia do "Pião da Casa Própria" (arranjo solo buzzer)
int melody[] = {
  // Intro com baixo e melodia
  NOTE_F3, NOTE_F4, NOTE_F3, NOTE_F4,
  NOTE_F3, NOTE_F4, NOTE_F3, NOTE_F4,
  NOTE_F3, NOTE_FS4, NOTE_G4, NOTE_A4, NOTE_A4, NOTE_G4,
```

Essas notas estão organizadas em um vetor chamado melody, que dentro dele está a sequência musical do programa "Pião da Casa Própria".

Outro vetor, chamado noteDurations, define o tempo de duração de cada nota.

```
  // Repetição
  NOTE_F3, NOTE_F4, NOTE_F3, NOTE_F4,
  NOTE_F3, NOTE_F4, NOTE_F3, NOTE_F4,
  NOTE_F3, NOTE_FS4, NOTE_G4, NOTE_A4, NOTE_A4, NOTE_G4,

  // Final
  NOTE_F3, NOTE_F4, NOTE_G4, NOTE_A4, NOTE_F4,
  NOTE_F3, NOTE_G4, NOTE_A4
};

int noteDurations[] = {
  8,8,8,8, 8,8,8,8, 8,8,8,8,4,4,
  8,8,8,8, 8,8,8,8, 8,8,8,8,4,4,
  8,8,8,8,4, 8,8,2
};
```

O código simula o bluetooth serial para conseguir testar no Tinkercad o código com o monitor serial. Em seguida define as variáveis e configura os pinos do Arduino para controlar um display de 7 segmentos e mostra como acender os segmentos para exibir os números de 0 a 9.

```
// --- Bluetooth + Display 7 segmentos ---
#define bluetooth Serial

const byte segmentPins[7] = {2, 3, 4, 5, 6, 7, 8}; // segmentos a-g
const byte decimalPin = 9; // ponto decimal

byte digitos[10][7] = {
  {1,1,1,1,1,1,0}, // 0
  {0,1,1,0,0,0,0}, // 1
  {1,1,0,1,1,0,1}, // 2
  {1,1,1,1,0,0,1}, // 3
  {0,1,1,0,0,1,1}, // 4
  {1,0,1,1,0,1,1}, // 5
  {1,0,1,1,1,1,1}, // 6
  {1,1,1,0,0,0,0}, // 7
  {1,1,1,1,1,1,1}, // 8
  {1,1,1,0,0,1,1}  // 9
};

```

O trecho a seguir configura os pinos do display como saída, inicia o Bluetooth e prepara o gerador de números aleatórios.
O bluetooth.begin(9600) serve para iniciar o uso de Bluetooth no código. Logo após essa configuração, o Arduino toca automaticamente a melodia de abertura (a música do pião), reproduzindo cada nota com a função tone e pausando o tempo necessário com delay.

```
void setup() {
  for (byte i = 0; i < 7; i++) {
    pinMode(segmentPins[i], OUTPUT);
    digitalWrite(segmentPins[i], LOW);
  }
  pinMode(decimalPin, OUTPUT);
  digitalWrite(decimalPin, LOW);

  pinMode(BUZZER, OUTPUT);

  bluetooth.begin(9600);
  randomSeed(analogRead(A0));

  // Toca melodia de abertura
  for (int i = 0; i < sizeof(melody)/sizeof(int); i++) {
    int duration = 1000 / noteDurations[i];
    tone(BUZZER, melody[i], duration);
    delay(duration * 1.30);
    noTone(BUZZER);
  }
}
```

Esta função acende os segmentos do display um por vez, em sequência, simulando um giro circular. Depois de algumas voltas, apaga tudo no final. A função animarGiroComMusica() utiliza um vetor ordem para definir qual segmento acender em qual momento, criando o efeito visual de rotação.
Durante essa animação, a função tocar Melodia Durante Animação é chamada constantemente para tocar as notas da música sem bloquear o código, usando millis.

```
void limparDisplay() {
  for (byte seg = 0; seg < 7; seg++) {
    digitalWrite(segmentPins[seg], LOW);
  }
  digitalWrite(decimalPin, LOW);
}

void mostrarDigito(int digito, bool ponto) {
  if (digito < 0 || digito > 9) return;

  for (byte seg = 0; seg < 7; seg++) {
    digitalWrite(segmentPins[seg], digitos[digito][seg]);
  }
  digitalWrite(decimalPin, ponto ? HIGH : LOW);
}

// Toca melodia em tempo real (sem delay bloqueante)
void tocarMelodiaDuranteAnimacao(unsigned long &ultimoSom, int &indiceNota) {
  int totalNotas = sizeof(melody)/sizeof(int);
  if (indiceNota >= totalNotas) return;
```

Estas duas funções servem para mostrar o número de 1 a 9 que foi sorteado e depois desligam todos os segmentos do display, apagando qualquer número ou ponto que estiver sendo mostrado.

```
  unsigned long agora = millis();
  int duracao = 1000 / noteDurations[indiceNota];

  if (agora - ultimoSom >= duracao * 1.30) {
    tone(BUZZER, melody[indiceNota], duracao);
    ultimoSom = agora;
    indiceNota++;
  }
}

// Anima segmentos enquanto toca música
void animarGiroComMusica(byte voltas = 2, int delayMs = 70) {
  byte ordem[] = {0, 1, 2, 3, 4, 5, 0, 6};
  int totalEtapas = voltas * (sizeof(ordem) / sizeof(ordem[0]));

  unsigned long ultimoSom = millis();
  int indiceNota = 0;

  for (int i = 0; i < totalEtapas; i++) {
    limparDisplay();
    byte idx = ordem[i % (sizeof(ordem) / sizeof(ordem[0]))];
    digitalWrite(segmentPins[idx], HIGH);
```

Quando é enviado o comando 'S' ou 's', o código sorteia um número de 0 a 9, pisca o ponto decimal por 1 segundo e depois mostra o número fixo no display por 4 segundos antes de apagar. Durante o sorteio, a função animarGiroComMusica() é chamada para simular o giro com som.

```
    tocarMelodiaDuranteAnimacao(ultimoSom, indiceNota);

    delay(delayMs);
  }

  limparDisplay();
}

void processarComando(char c) {
  if (c == 'S' || c == 's') {
    int sorteado = random(0, 10); // número de 0 a 9

    animarGiroComMusica(4, 70); // agora com música do pião

    unsigned long inicio = millis();
    while (millis() - inicio < 1000) {
      mostrarDigito(sorteado, true);
      delay(250);
      mostrarDigito(sorteado, false);
      delay(250);
    }
```

O último trecho do código verifica se chegou algum comando via Bluetooth e, se sim, lê o caractere e executa a função que processa esse comando.

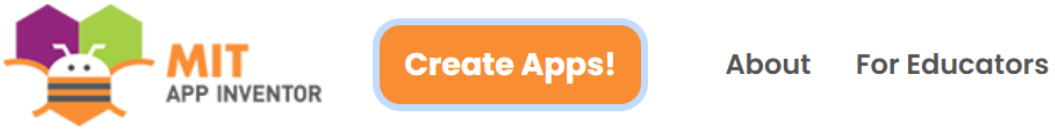
```
    mostrarDigito(sorteado, false);
    delay(1500);
    limparDisplay();
  }
}

void loop() {
  if (bluetooth.available()) {
    char c = bluetooth.read();
    processarComando(c);
  }
}
```

Sorteador de número via Bluetooth: Criando aplicativo.

Este projeto consiste na criação de um aplicativo mobile de sorteio integrado a uma placa Arduino, desenvolvido no site MIT App Inventor.

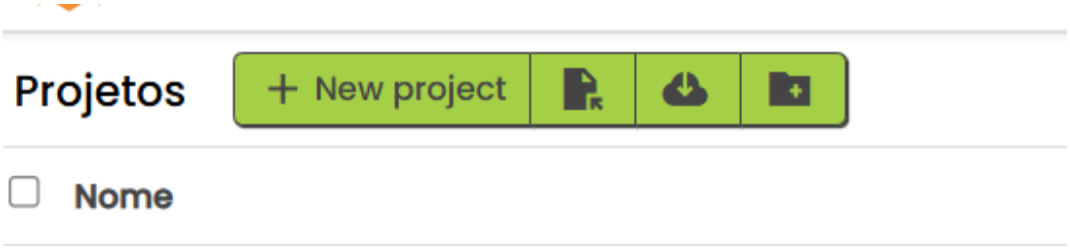
Após finalizar a programação no Arduino, foi desenvolvido o aplicativo com o nome Show Numérico, inspirado na sua principal função: sortear números de forma prática e divertida.

Para começar a utilizar a plataforma, acesse o site de desenvolvimento digitando na barra de endereços do seu navegador o seguinte link: https://appinventor.mit.edu.



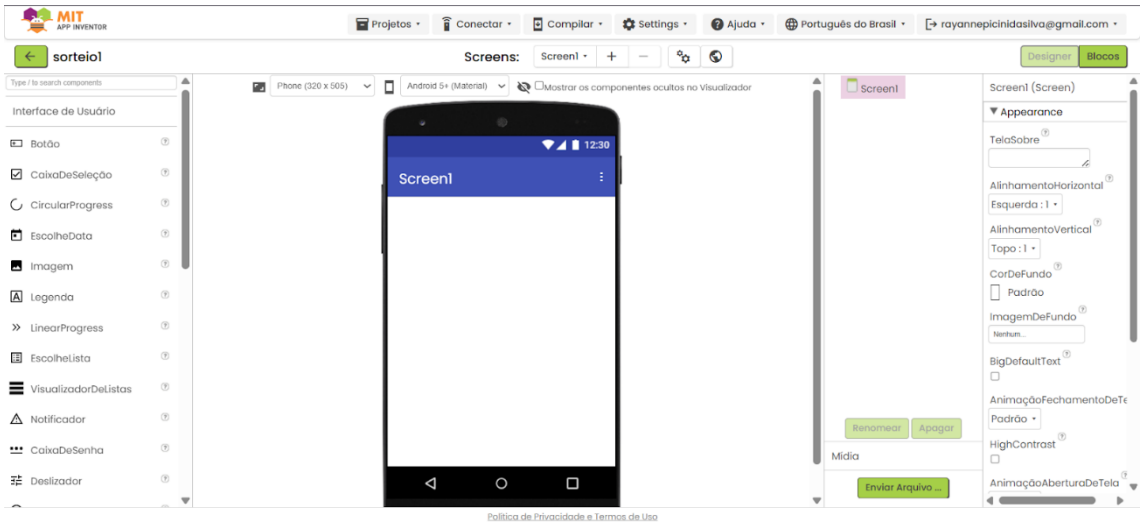
Você será redirecionado para a página inicial do App Inventor. Clique em "Continue to App Inventor" e faça login com sua conta Google. Assim que estiver na tela principal, clique no botão “Start new project”.

Digite um nome para o seu projeto (sem espaços) e clique em OK.



Em seguida, você será direcionado para a interface de design do aplicativo. Agora iremos montar um aplicativo que faça sorteios por meio do bluetooth.

Para isso, será necessário o uso da programação em blocos. Porém, primeiramente será necessária a construção do layout do app.



O aplicativo possui duas funcionalidades principais, representadas por dois botões:

- "Sortear": realiza o sorteio de um número aleatório normalmente.
- "Voz": permite que o sorteio seja feito por comando de voz. Basta o usuário dizer “sortear” ou “draw” (em inglês) e o sorteio é iniciado automaticamente, sem precisar tocar na tela

Esquema de Funcionamento

1. O aplicativo se conecta via Bluetooth ao Arduino.
2. Ao pressionar o botão "Sortear", o app envia um comando para o Arduino realizar o sorteio.
3. Ao usar o botão "Voz", o app escuta o comando falado.
4. Se o usuário disser “sortear” ou “draw”, o comando é enviado automaticamente.

Este app permite que o usuário conecte-se a dispositivos Bluetooth e visualize o status da conexão.

Os principais componentes usados são: - ListPicker1: para listar dispositivos pareados. ClienteBluetooth1: gerencia a conexão Bluetooth. - Temporizador1: verifica a conexão em tempo real. - Label (conectadoounao): exibe 'Conectado' ou 'Não Conectado'. - ActivityStarter1: ativa o Bluetooth ao iniciar.

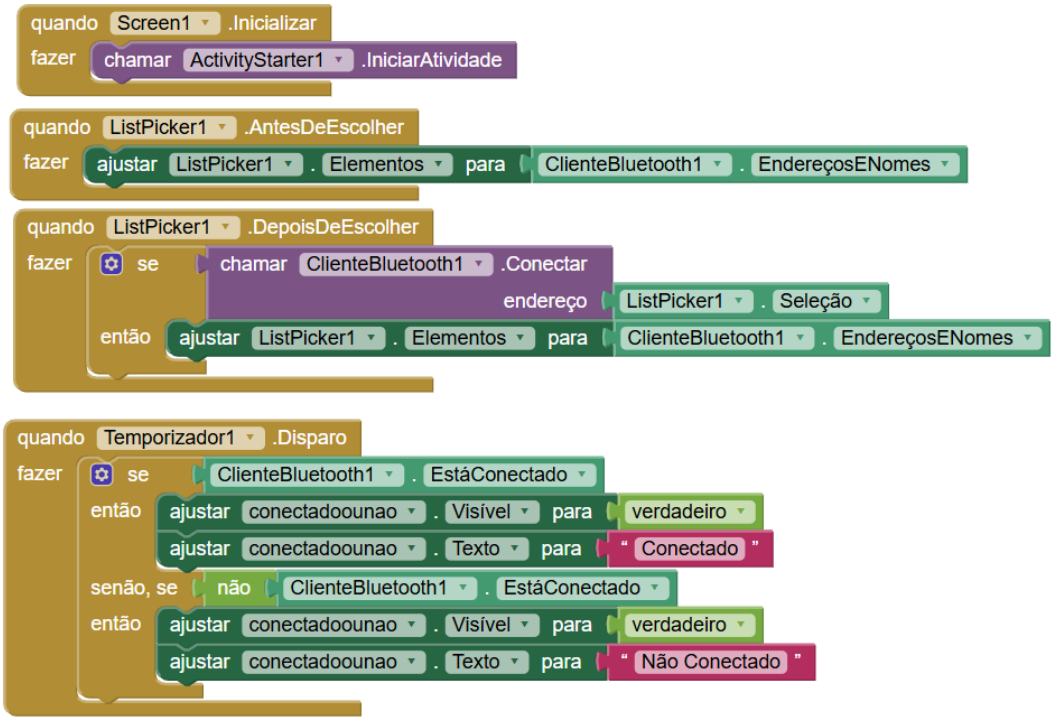
Passos lógicos dos blocos:

1. Ao iniciar o app, o Bluetooth é ativado.

2. Quando o usuário clica no ListPicker1, ele carrega os dispositivos pareados.

3. Após selecionar, o app tenta conectar ao dispositivo escolhido.

4. O Temporizador1 verifica se está conectado e atualiza a label de status.



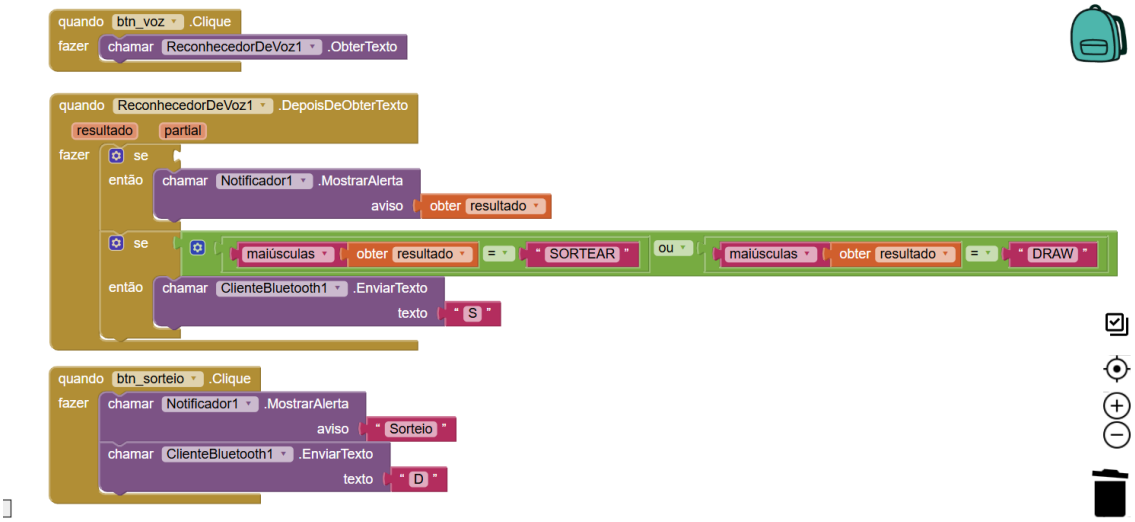
Nesta etapa, foram adicionados comandos de voz e botões que enviam texto via Bluetooth. Os componentes utilizados foram:

- btn_voz: botão que ativa o componente.
- ReconhecedorDeVoz1: responsável por interpretar comandos de voz como "SORTEAR" ou "DRAW".
- btn_sorteio: botão que envia manualmente um comando via Bluetooth.
- Notificador1: componente que exibe alertas com os comandos executados.

Lógica de funcionamento:

1. Ao clicar no botão btn_voz, o microfone é ativado.
2. O texto captado pelo ReconhecedorDeVoz1 é analisado.

3. Se a palavra reconhecida for "SORTEAR" ou "DRAW", é enviado o caractere 'S' via Bluetooth.
4. Já o botão btn_sorteio envia diretamente o caractere 'D' via Bluetooth, executando o sorteio manualmente.



É possível ver um exemplo do aplicativo funcionando no link a seguir: https://youtu.be/Ag7gjjD6_00?si=T6HJRh1_Hwm34ld7