

```
#include <MeetAndroid.h>
```

```
//Motor esquerdo
```

```
const int E_PWM = 6;
```

```
const int E_TRAS = 7;
```

```
const int E_FRENTE = 8;
```

```
//Motor direito
```

```
const int D_PWM = 3;
```

```
const int D_TRAS = 5;
```

```
const int D_FRENTE = 4;
```

```
const int PWM_MIN = 70;
```

```
const int PWM_MAX = 255;
```

```
//Amarino
```

```
MeetAndroid amarino;
```

```
const float MIN = 2;
```

```
const float MAX = 9;
```

```
const int X = 0;
```

```
const int Y = 1;
```

```
const int Z = 2;
```

```
long timestampUltimaLeitura;
```

```
float eixos[3];
```

```
//---
```

```
void setup() {
```

```
  //Inicialização do motor esquerdo
```

```
  pinMode(E_PWM, OUTPUT);
```

```
  pinMode(E_FRENTE, OUTPUT);
```

```
  pinMode(E_TRAS, OUTPUT);
```

```
  //Inicialização do motor direito
```

```
  pinMode(D_PWM, OUTPUT);
```

```
  pinMode(D_FRENTE, OUTPUT);
```

```
  pinMode(D_TRAS, OUTPUT);
```

```
  //Inicialização do Amarino
```

```
  Serial.begin(57600);
```

```
  amarino.registerFunction(accelerometro, 'A'); // a string
```

```
}
```

```
void loop() {
```

```
  amarino.receive(); // you need to keep this in your loop() to receive events
```

```
  // distancia = readDistance();
```

```
  andar();
```

```
  delay(20);
```

```
}
```

```
void andar() {
```

```
  //Verifica se a última leitura foi há mais de 2 segundos. Se sim, pára o carrinho.
```

```
if(millis() - timestampUltimaLeitura > 2000) {  
    parar();  
}  
  
//Se a leitura for recente, então comanda o carrinho  
  
else {  
    int intensidadeY = intensidade(Y);  
    int intensidadeX = intensidade(X);  
  
    //Está indo pra frente?  
    if(eixos[X] >= MIN) {  
        //Está indo para um dos lados também?  
        if (eixos[Y] >= MIN) {  
            amarino.send("frente-esquerda");  
            frenteDireita(intensidadeX);  
            frenteEsquerda(intensidadeXY());  
        } else if(eixos[Y] < -MIN) {  
            amarino.send("frente-direita");  
            frenteDireita(intensidadeXY());  
            frenteEsquerda(intensidadeX);  
        } else {  
            amarino.send("frente");  
            frenteDireita(intensidadeX);  
            frenteEsquerda(intensidadeX);  
        }  
    } else if(eixos[X] < -MIN) {  
        amarino.send("tras");  
        trasDireita(intensidadeX);  
    }  
}
```

```

        trasEsquerda(intensidadeX);
    } else if (eixos[Y] < -MIN) {
        amarino.send("esquerda");
        frenteDireita(0);
        frenteEsquerda(intensidadeY);
    } else if(eixos[Y] >= MIN) {
        amarino.send("direita");
        frenteDireita(intensidadeY);
        frenteEsquerda(0);
    } else {
        parar();
    }
}
}
}

```

```

int intensidade(int eixo) {
    int intensidade = map(abs(eixos[eixo]), MIN, MAX, PWM_MIN, PWM_MAX);
    if(intensidade > PWM_MAX) {
        intensidade = PWM_MAX;
    } else if(intensidade < PWM_MIN) {
        intensidade = 0;
    }
    return intensidade;
}

```

```

int intensidadeXY() {
    int intensidadeXY = intensidade(X) - intensidade(Y);
}

```

```
if(intensidadeXY < 0) {  
    intensidadeXY = 0;  
}  
return intensidadeXY;  
}
```

```
void frenteEsquerda(int intensidade) {  
    analogWrite(E_PWM, intensidade);  
    digitalWrite(E_FRENTE, HIGH);  
    digitalWrite(E_TRAS, LOW);  
}
```

```
void frenteDireita(int intensidade) {  
    analogWrite(D_PWM, intensidade);  
    digitalWrite(D_FRENTE, HIGH);  
    digitalWrite(D_TRAS, LOW);  
}
```

```
void trasEsquerda(int intensidade) {  
    analogWrite(E_PWM, intensidade);  
    digitalWrite(E_FRENTE, LOW);  
    digitalWrite(E_TRAS, HIGH);  
}
```

```
void trasDireita(int intensidade) {  
    analogWrite(D_PWM, intensidade);  
    digitalWrite(D_FRENTE, LOW);  
}
```

```
digitalWrite(D_TRAS, HIGH);  
}
```

```
void parar() {  
    digitalWrite(E_PWM, LOW);  
    digitalWrite(D_PWM, LOW);  
}
```

```
/*  
*/  
void acelerometro(byte flag, byte numOfValues)  
{  
    timestampUltimaLeitura = millis();  
  
    int length = amarino.stringLength();  
    char data[length];  
    amarino.getString(data);  
    String leitura = data;  
  
    //Separa a string recebida do Amarino em 3 strings  
    //Ex: 1.42342;-3.43242;-1.34232  
    int offset = 0;  
    int count = 0;  
    for(int i = 0; i < length; i++) {  
        //Toda vez que encontrar um ';', le de 'offset' até 'i'  
        if(data[i] == ';') {  
            eixos[count] = stringToFloat(leitura.substring(offset, i));
```

```
offset = i + 1;
```

```
count++;
```

```
//Se o último ';' na string tiver o mesmo índice de 'i', então pega
```

```
//o restante da string no próximo elemento do array.
```

```
if(leitura.lastIndexOf(';') == i) {
```

```
    eixos[count] = stringToFloat(leitura.substring(offset, length));
```

```
}
```

```
}
```

```
}
```

```
// amarino.send("x");
```

```
// amarino.send(eixos[X]);
```

```
// amarino.send("y");
```

```
// amarino.send(eixos[Y]);
```

```
// amarino.send("z");
```

```
// amarino.send(eixos[Z]);
```

```
}
```

```
/**
```

```
 * Converts a String to a float
```

```
 */
```

```
float stringToFloat(String str) {
```

```
    char arr[str.length()];
```

```
    str.toCharArray(arr, sizeof(arr));
```

```
    return atof(arr);
```

```
}
```