



II. Testing program は説明書のように行えなかったので私の成功した方法を書いています。

赤外線コントロールカー




The points of section

赤外線リモコンは、遠隔操作のために広く使用されている方法です。車には赤外線受信機が装備されているため、赤外線リモートコントローラを使用して制御することができます。

Learning parts:

-  赤外線リモコンと受信機を理解する
-  リモートコントロールの原理を理解する

準備:

-  カー（バッテリー付き）
-  USB ケーブル
-  赤外線受信モジュールと赤外線リモートコントローラ

赤外線リモートコントローラ：



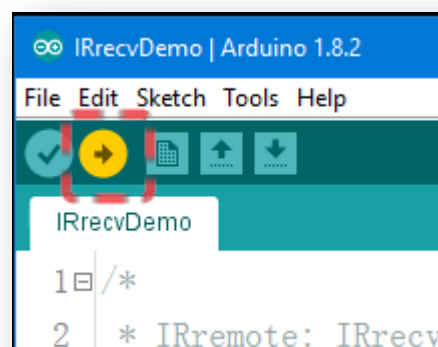
II. Testing program

このプログラムでは、最初にライブラリファイルを追加する必要がある。UNO をコンピュータに接続し、次のように Arduino IDE を開きます

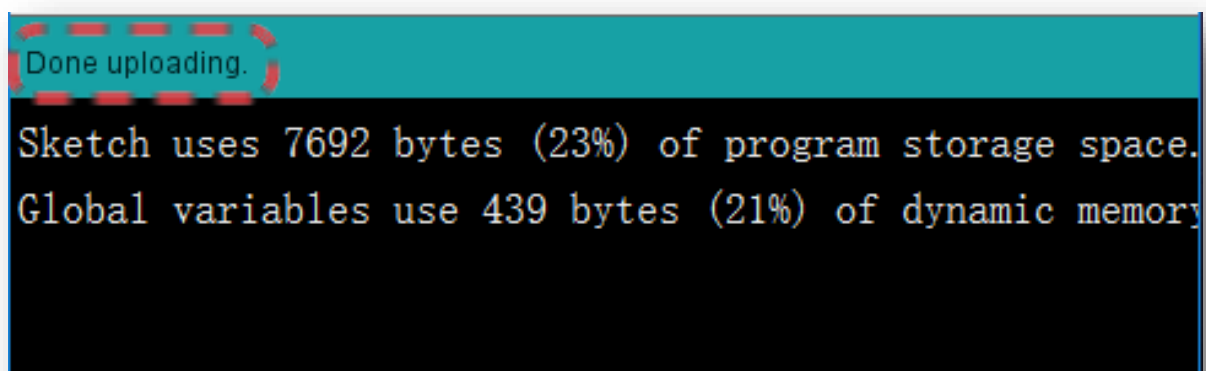
Lesson3InfraredRemoteControl Car→IRremote.zip→IRremote→examples→IRrecvDemo→IRrecvDemo.ino



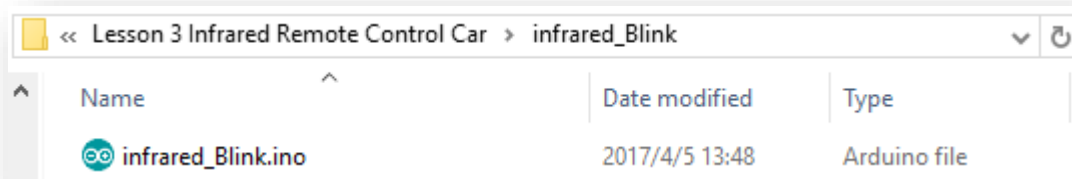
Click compile button.



コンパイルを完了します。 そうでない場合は、IRremote ライブラリが正常にインストールされていないことを示します。 IRremote ライブラリを再度追加してください。



"¥ Elegoo Smart Robot カーキット V3.0 ¥ レッスン 3 赤外線リモコンカー ¥ infrared_Blink ¥ infrared_Blink.ino"
のパスにあるコードファイルを開き、プログラムをコントローラボードにアップロードします。



Code preview :

```
//www.elegoo.com

#include <IRremote.h>

#define RECV_PIN 12      //Infrared signal receiving pin
#define LED 13           //define LED pin
#define L 16738455
#define UNKNOWN_L 1386468383

bool state = LOW;        //define default input mode
unsigned long val;

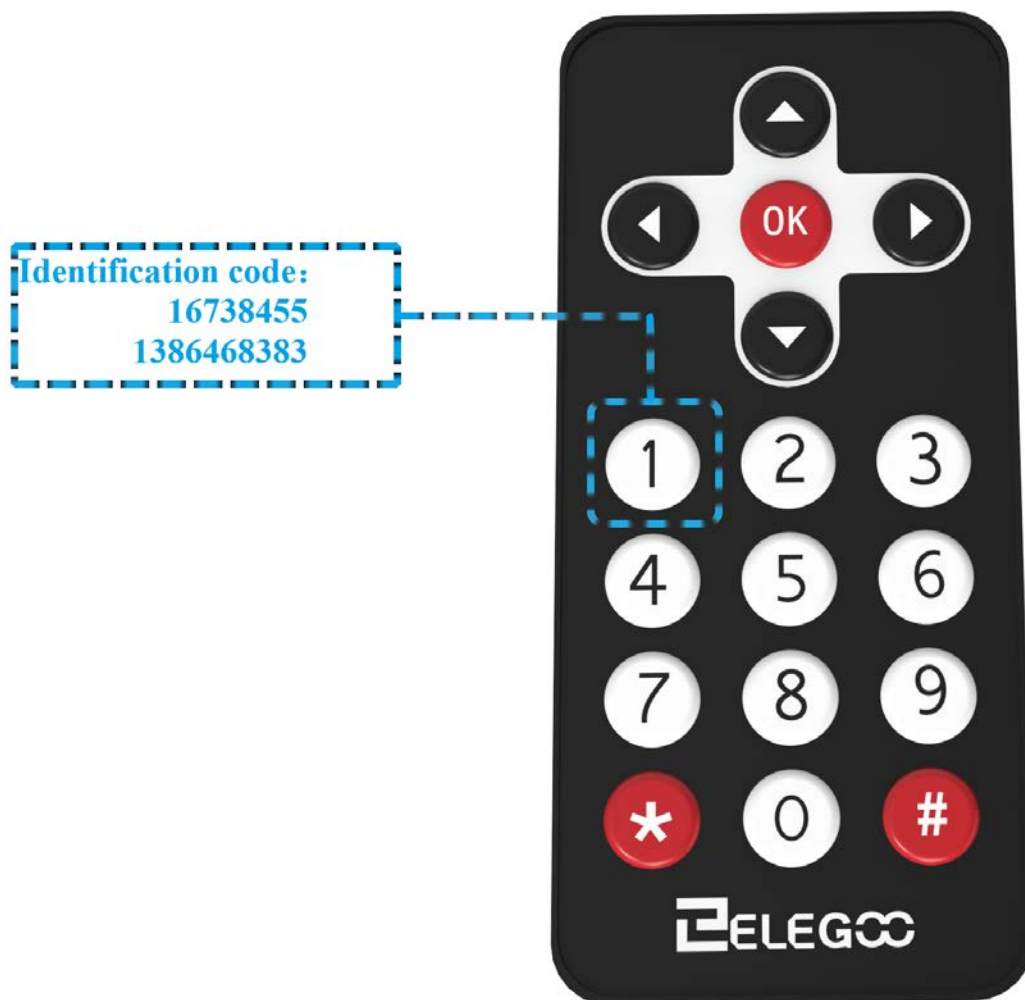
IRrecv irrecv(RECV_PIN); //initialization
decode_results results;   //Define structure type

void stateChange() {
    state = !state;
    digitalWrite(LED, state);
}

void setup() {
    pinMode(LED, OUTPUT); //initialize LED as an output
    Serial.begin(9600);   // debug output at 9600 baud
    irrecv.enableIRIn();  // Start receiving
}

void loop() {
    if (irrecv.decode(&results)) {
        val = results.value;
        Serial.println(val);
        irrecv.resume();    // Receive the next value
        delay(150);
        if(val == L || val == UNKNOWN_L) {
            stateChange();
        }
    }
}
```

車にコンピュータを接続した後、電源スイッチをオンにして地面に乗せてください。車に向かって “1” のボタンを押して、車を観察し、ラベル “L” の LED ランプ点灯を確認したら、 拡張ボードの電源を切る。



Ⅲ. 原則の導入

1. 赤外線遠隔制御システムの原理

ユニバーサル赤外線遠隔制御システムは、送信と受信の2つの部分から成り、送信部は赤外線リモコンで構成され、受信部は赤外線受信管で構成されています。赤外線リモコンで送信される信号は、バイナリパルスコードのシリアルです。無線送信中に他の赤外線信号が散逸しないようにするには、特定の搬送波周波数で変調し、赤外線放射フォトトランジスタを通じて発射するのが一般的です。赤外線受信管は、他のノイズ波をフィルタリングし、所与の周波数の信号のみを受信し、それらを復調するバイナリパルスコードに復元する。受信管を内蔵し、赤外発光ダイオードから弱い電気信号に変換された光信号を変換し、IC内部のアンプで増幅し、自動利得制御、バンドパスフィルタリング、復調、波形整形、元に戻す赤外線受信モジュールの信号出力ピンを介して電気機器に入力される符号化により回路を認識する。

2. Protocol of infrared remote controlling (赤外線リモコンのプロトコル)

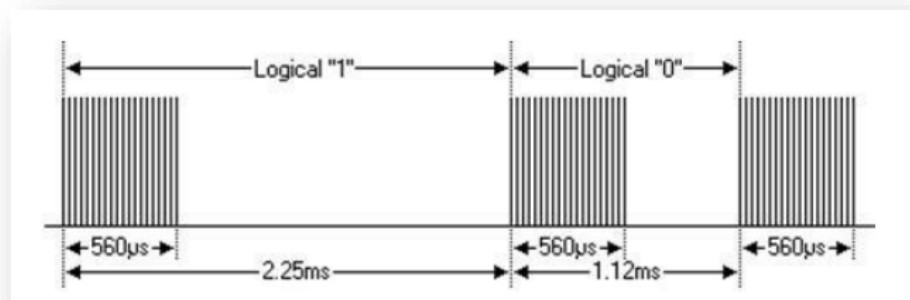
一致した IR リモート制御の符号化方式は、NEC プロトコルである。

次に、NEC のプロトコルが何であることを学びましょう。

Features:

- (1) 8 address bit, 8 order bit
- (2) 信頼性を保証するために、アドレスビットとオーダービットを2回送信する
- (3) パルス位置変調
- (4) キャリア周波数は 38kHz です
- (5) あらゆるビットの時間は 1.125ms または 2.25ms です

論理 0 と論理 1 の定義は次のとおりです。



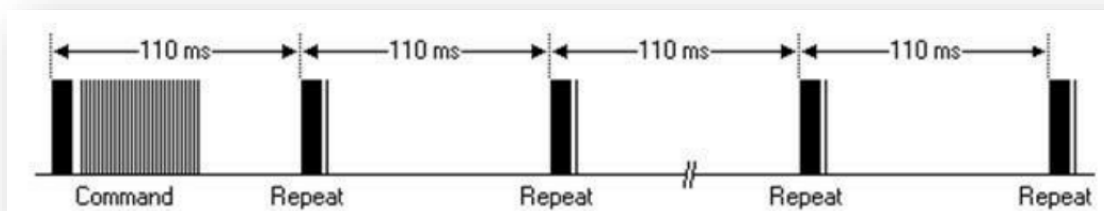
プロトコルは以下の通りです:

即座に送信パルスを緩める:



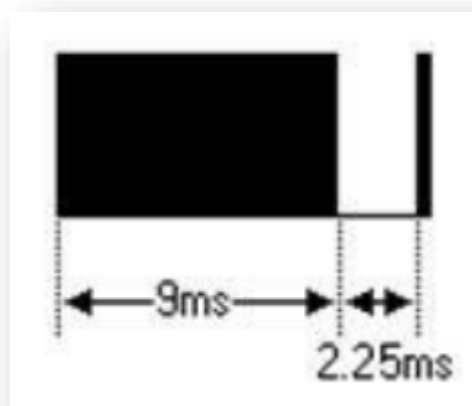
注：これはLSB（最下位ビット）を最初に送信するプロトコルです。上記パルスの転送アドレスは 0x59、次は 0x16 です。1つのメッセージは 9ms の高レベルから始まり、その後は 4.5ms の低レベル（2 レベルのガイダンスコード）とアドレスコードと注文コードによって開始されます。住所と注文は 2 回送信されます。2 回目は、すべてのビットが反転され、使用する受信メッセージを確認するために使用できます。あなたがそれに興味がないなら、総送信時間は固定です、あなたは反転の信頼性を無視することができ、16 ビットでアドレスと注文を展開することができます！なぜなら、すべてのビットが反対であれば、長さが繰り返されるからです。

プレスパルスは、時間がたつと緩められます



リモコンのボタンが押されていても、コマンドが送信された。ボタンを押したままにすると、最初の 110ms のパルスは上と異なり、重複したコードは 110ms ごとに送信されます。重複コードは、9ms の高レベルパルスと 2.25 の低レベルと 560 μ s の高レベルで構成されています。

繰り返しパルス：



注：インパルス波形がセンサーの統合に入った後、センサーの統合がデコードされ、信号が拡大され、プラスチックでなければならないため、赤外線信号がなく、出力端子が高レベルであり、低レベルである 信号があるとき。したがって、出力信号のレベルは送信端子とは反対である。誰もがオシロスコープを通して受信パルスを見ることができ、波形が見えるプログラムを理解できます。

3. リモートコントロールカーのプログラミングのアイデア

NEC コードの特性と受信側の波に基づいて、受信側の波を 4 つの部分に分けている。すなわち、先頭コード（パルス 9ms と 4.5ms）、アドレスコード（8 ビットアドレスコードと 8 ビット アドレスフェッチ）、16 ビット・アドレス・コード（8 ビット・アドレス・コードと 8 ビット・アドレス・フェッチを含む）、16 ビット・オーダ・コード（8 ビット・オーダ・コードと 8 ビット・オーダ・フェッチを含む）、リピート・コード（9ms のパルス、2.25ms、560us）。

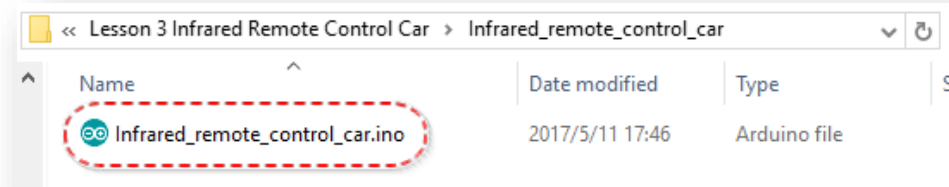
論理 “01”、論理 “1”、リーディングパルス、リピートパルス、テストされた時間に応じて区別されて受信された波のハイレベルとローレベルをテストするタイマを利用する。各キーの注文コードが異なることにより、発注コードで行為が行われるため、先行コードと住所コードが正しく保管されているかどうか判断されます。車の実験中には、前後に進んで左右に回して停止するように車を制御する必要があります。つまり、5つのキーが必要です。その値は次のとおりです。

Remote control character	key value
Middle red button	16712445, 3622325019
Above triangle	16736925, 5316027
Below triangle	16754775, 2747854299
Left triangle	16720605, 1386468383
Right triangle	16761405, 553536955

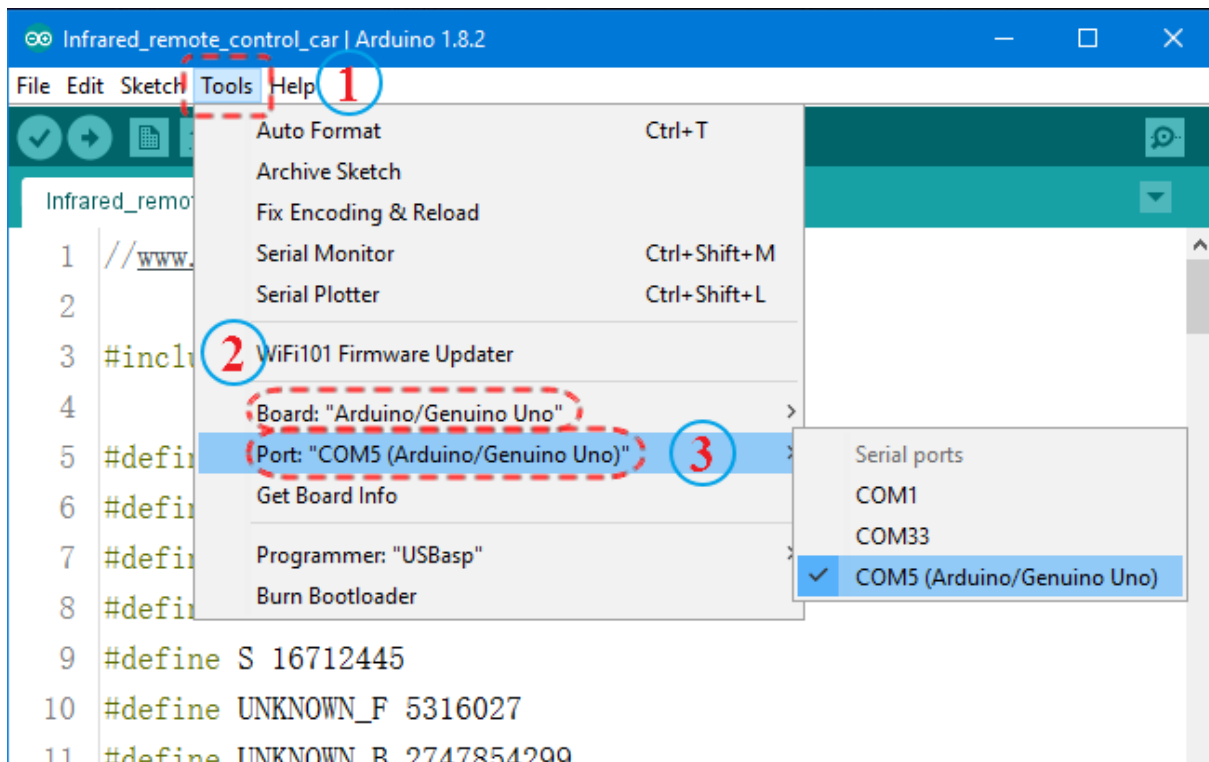


IV. 遠隔制御車を作る

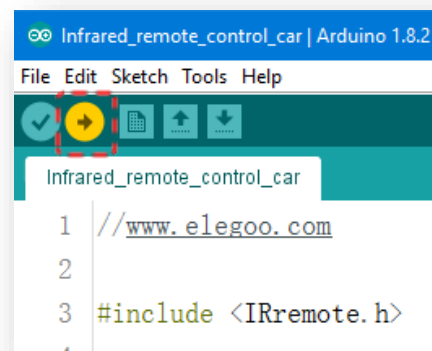
パス内のコードファイルを開く ¥ Elegoo Smart Robot カーキット V3.0 ¥ Infrared_remote_control_car ¥ Infrared_remote_control_car.ino “を選択し、以下のようにプログラムを車にアップロードします。



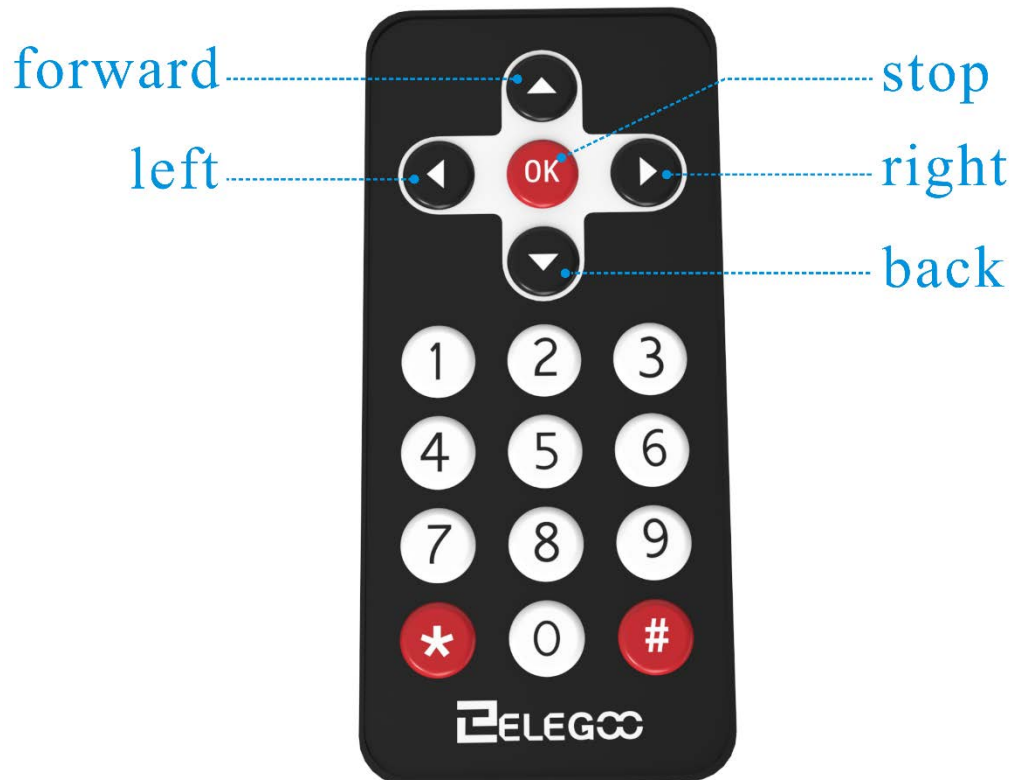
Arduino Uno Board とシリアルポートを選択します。



アップロードボタンを押す



アップロードが完了したら、車をコンピュータに接続します。その後、電源スイッチをオンにして、車を地上に置きます。リモコンのボタンを押すと、指示通りにカーが移動するのがわかります。



ほら、今あなたは赤外線制御車で楽しく遊ぶことができます。

Code preview:

```
//www.elegoo.com

#include <IRremote.h>

#define F 16736925
#define B 16754775
#define L 16720605
#define R 16761405
#define S 16712445
#define UNKNOWN_F 5316027
#define UNKNOWN_B 2747854299
#define UNKNOWN_L 1386468383
#define UNKNOWN_R 553536955
#define UNKNOWN_S 3622325019

#define RECV_PIN 12
#define ENA 5
#define ENB 6
```

```
#define IN1 7
#define IN2 8
#define IN3 9
#define IN4 11
#define carSpeed 150

IRrecv irrecv(RECV_PIN);
decode_results results;
unsigned long val;
unsigned long preMillis;

void forward(){
  digitalWrite(ENA,HIGH);
  digitalWrite(ENB,HIGH);
  digitalWrite(IN1,HIGH);
  digitalWrite(IN2,LOW);
  digitalWrite(IN3,LOW);
  digitalWrite(IN4,HIGH);
  Serial.println("go forward!");
}
void back(){
  digitalWrite(ENA,HIGH);
  digitalWrite(ENB,HIGH);
  digitalWrite(IN1,LOW);
  digitalWrite(IN2,HIGH);
  digitalWrite(IN3,HIGH);
  digitalWrite(IN4,LOW);
  Serial.println("go back!");
}
void left(){
  analogWrite(ENA,carSpeed);
  analogWrite(ENB,carSpeed);
  digitalWrite(IN1,LOW);
  digitalWrite(IN2,HIGH);
  digitalWrite(IN3,LOW);
  digitalWrite(IN4,HIGH);
  Serial.println("go left!");
}
void right(){
  analogWrite(ENA,carSpeed);
  analogWrite(ENB,carSpeed);
  digitalWrite(IN1,HIGH);
  digitalWrite(IN2,LOW);
  digitalWrite(IN3,HIGH);
```

```
digitalWrite(IN4,LOW);
Serial.println("go right!");
}
void stop(){
digitalWrite(ENA, LOW);
digitalWrite(ENB, LOW);
Serial.println("STOP!");
}

void setup() {
Serial.begin(9600);
pinMode(IN1,OUTPUT);
pinMode(IN2,OUTPUT);
pinMode(IN3,OUTPUT);
pinMode(IN4,OUTPUT);
pinMode(ENA,OUTPUT);
pinMode(ENB,OUTPUT);
stop();
irrecv.enableIRIn();
}

void loop() {
if (irrecv.decode(&results)){
preMillis = millis();
val = results.value;
Serial.println(val);
irrecv.resume();
switch(val){
case F:
case UNKNOWN_F: forward(); break;
case B:
case UNKNOWN_B: back(); break;
case L:
case UNKNOWN_L: left(); break;
case R:
case UNKNOWN_R: right();break;
case S:
case UNKNOWN_S: stop(); break;
default: break;
}
}
else{
if(millis() - preMillis > 500){
stop();
```

```
    preMillis = millis();  
  }  
}  
}
```