ESP32 BLE 键盘库

该库允许您将 ESP32 用作蓝牙键盘并控制它的功能。
你也可能对此有兴趣：

- ESP32-BLE-鼠标
- ESP32-BLE-游戏手柄

特征

- 发送击键
- 发短讯
- 按下/释放单个键
- 支持媒体密钥
- 读取 Numlock/Capslock/Scrolllock 状态
- 设置电池电量（基本上可以，但不会显示在 Android 的状态栏中）
- 与安卓兼容
- 与 Windows 兼容
- 与 Linux 兼容
- 与 MacOS X 兼容（不稳定，有些人有问题，不适用于旧设备）
- 与iOS兼容（不稳定，有些人有问题，不适用于旧设备）

安装

- （确保您可以将 ESP32 与 Arduino IDE 一起使用。可以在此处找到说明。）
- 从发布页面下载此库的最新版本。
- 在 Arduino IDE 中，转到“Sketch”->“Include Library”->“Add .ZIP Library...”并选择刚刚下载的文件。
- 您现在可以转到“文件”->“示例”->“ESP32 BLE 键盘”并选择任何示例以开始使用。

例子

```
/**
 * 示例将ESP32变成这个蓝牙键盘，按该键写入单词，然后按Enter，按媒体键，按CTRL+ALT+DELETE
 */
#include <BleKeyboard.h>

BleKeyboard bleKeyboard;

void setup() {
  Serial.begin(115200);
  Serial.println("Starting BLE work!");
  bleKeyboard.begin();
}

void loop() {
  if(bleKeyboard.isConnected()) {
    Serial.println("Sending 'Hello world'...");
    bleKeyboard.print("Hello world");

    delay(1000);
```

```
    Serial.println("Sending Enter key...");
    bleKeyboard.write(KEY_RETURN);

    delay(1000);

    Serial.println("Sending Play/Pause media key...");
    bleKeyboard.write(KEY_MEDIA_PLAY_PAUSE);

    delay(1000);

   //
   // 下面是按多个键盘物质符的示例
   // 默认情况下被注释掉。.
   //
   /* Serial.println("Sending Ctrl+Alt+Delete...");
    bleKeyboard.press(KEY_LEFT_CTRL);
    bleKeyboard.press(KEY_LEFT_ALT);
    bleKeyboard.press(KEY_DELETE);
    delay(100);
    bleKeyboard.releaseAll();
    */

  }
  Serial.println("Waiting 5 seconds...");
  delay(5000);
}
```

API 文档

BleKeyboard 界面与键盘界面几乎相同，因此您可以在此处使用文档： https://www.arduino.cc/reference/en/language/functions/usb/keyboard/

请记住，您必须使用`bleKeyboard`而不是 just `Keyboard`并且您需要在脚本顶部的这两行：

```
#include <BleKeyboard.h>
BleKeyboard bleKeyboard;
```

除此之外，您还可以发送媒体键（USB 键盘库无法做到这一点）。支持如下：

- KEY_MEDIA_NEXT_TRACK
- KEY_MEDIA_PREVIOUS_TRACK
- KEY_MEDIA_STOP
- KEY_MEDIA_PLAY_PAUSE
- KEY_MEDIA_MUTE
- KEY_MEDIA_VOLUME_UP
- KEY_MEDIA_VOLUME_DOWN
- KEY_MEDIA_WWW_HOME
- KEY_MEDIA_LOCAL_MACHINE_BROWSER // 在 Windows 上打开“我的电脑”
- KEY_MEDIA_CALCULATOR
- KEY_MEDIA_WWW_BOOKMARKS
- KEY_MEDIA_WWW_SEARCH
- KEY_MEDIA_WWW_STOP
- KEY_MEDIA_WWW_BACK
- KEY_MEDIA_CONSUMER_CONTROL_CONFIGURATION // 媒体选择
- KEY_MEDIA_EMAIL_READER

您还可以设置蓝牙特定信息（可选）：而不是`BleKeyboard bleKeyboard;`您可以做`BleKeyboard bleKeyboard("Bluetooth Device Name", "Bluetooth Device Manufacturer", 100);`.（最大长度为 15 个字符，超出的任何内容都将被截断。）
第三个参数是设备的初始电池电量。要稍后调整电池电量，您只需调用例如`bleKeyboard.setBatteryLevel(50)`（将电池电量设置为 50%）。
默认情况下，电池电量将设置为 100%，设备名称为`ESP32 Bluetooth Keyboard`，制造商为`Espressif`。
还有一种`setDelay`方法可以设置每个按键事件之间的延迟。例如`bleKeyboard.setDelay(10)`（10 毫秒）。默认值为`8`.
此功能旨在补偿某些无法处理快速输入的应用程序和设备，如果在短时间内发送太多键，则会跳过字母。

NimBLE 模式

NimBLE 模式可显着节省 RAM 和闪存。

比较（SendKeyStrokes.ino 在编译时）

Standard

```
RAM:   [=         ]   9.3% (used 30548 bytes from 327680 bytes)
Flash: [========  ]  75.8% (used 994120 bytes from 1310720 bytes)
```

NimBLE mode

```
RAM:   [=         ]   8.3% (used 27180 bytes from 327680 bytes)
Flash: [====      ]  44.2% (used 579158 bytes from 1310720 bytes)
```

比较（运行时的 SendKeyStrokes.ino）

	Standard	NimBLE mode	区别
`ESP.getHeapSize()`	296.804	321.252	**+ 24.448**
`ESP.getFreeHeap()`	143.572	260.764	**+ 117.192**
`ESP.getSketchSize()`	994.224	579.264	**- 414.960**

如何激活 NimBLE 模式?

ArduinoIDE：

取消注释 BleKeyboard.h 中的第一行

```
#define USE_NIMBLE
```

PlatFormIO：

将您更改`platformio.ini`为以下设置

```
lib_deps =
  NimBLE-Arduino

build_flags =
  -D USE_NIMBLE
```