Ttpyboard简介

1. ****介绍tpyboard****

为了最大使用好你的 ttpyboard 开发板，工作前的一些注意事项是需要知道的。  
  
****1.1 小心你的tpyboard开发板****  
因为 tpyboard 开发板没有防护罩的缘故，故需要注意的一些事项：  
. 轻力插拔 USB 线。尽管 USB 接头是焊接在电路板上且十分牢固的，一旦有所损坏将非常难以修理。  
. 静电能够损坏开发板上的元器件。如果你在你的工作范围积累了许多的静电（例如干冷的环境下），需要额外小心注意不要击穿开发板。如果开发板是装在静电袋里边的，这个袋子将是保存和携带该开发板的最好的抗静电工具（其由传导性泡沫的塑料组成）。  
如果在硬件层面你能够注意到这些事项，开发板使用起来不会有大问题。软件层面造成开发板损坏几乎是不可能的，所以大可随心所欲敲写你的代码。如果文件系统损坏，可以在接下来的内容中了解如何修复它。最糟糕的情况乃是需要重刷新MicroPython固件，但这可以轻易地通过一条USB线实现。  
  
****1.2 tpyboard的布局****  
USB 接头在板子的右上方，SD 卡槽在其左上方。卡槽和接口之下是四颗 LED 灯，从上到下依次为蓝色，橙色，绿色和红色。开关有两个，左边的用户开关和右边的复位开关。  
  
****1.3插入式供电****  
tpyboard 是通过 USB 线供电的。通过USB线连接PC是唯一适合的方法。如果连接成功，绿色的 LED 灯将亮起。  
  
****1.4外部电源供电****  
tpyborad 开发板可以通过干电池或其他外部电源供电  
连接电源时需要特别注意其正极负极，tpyboard 开发板上没有极性保护，所以任何东西连接其正极时都要非常非常小心。

****2.运行第一个脚本****  
从这篇教程开始将动手在 tpyboard 板上运行 Python 脚本，毕竟这是我们的目标！  
  
****2.1 连接开发板****  
通过 USB 线连接你的 PC 机（windows,mac,linux皆可）。你不可能搞错因为仅有这么一种连接方式。  
当连接成功后开发板将上电和进入开机程序，绿色的 LED 灯应该在半秒或更少的时间内亮起，当其熄灭时意味着开机程序已完成。  
  
****2.2 安装USB驱动****  
Windows：开发板将作为可移动磁盘出现。Window 将自动弹出窗口界面，或者你自己寻找进入。  
Windows系统同时会将开发板视为串口驱动，且设备将会自动连接。如果是这样，取消这样的连接，我们将在下一篇教程中展示串口驱动如何工作。  
.Mac ：在电脑桌面上开发板将作为可移动磁盘出现，其可能为“NONAME”，请点击打开里边的文件夹。  
.Linux ：开发版将作为可移动的多媒体设备出现。在 Ubuntu 下其将自动挂载并弹出开发板的文件夹。在其他的 Linux 系统下，开发板自动挂载或者需要手动实现。在命令行下敲入 lsblk 参看连接设备，然后敲入 mount /dev/sdb1（用对应的设备名替换sbd1）。或许你需要 boot 权限实现这一过程。  
至此开发板以移动磁盘的形式存在了，且有一个开发板驱动的窗口（或者命令行）显示出来。  
你所见到的设备是由开发板里边的 /flash实现的，其由以下四个文件关联组成：  
. boot.py  开发板启动时将执行这个该脚本，它设置了开发板的多个选项参数  
. main.py  包含你的 python 程序的主要脚本，在 boot.py 运行后被执行  
. README.txt  包含开启开发板的必要基础信息  
. pycdc.inf  确认 USB 设备连接的 Windows 配置文件，之后的教程中有更多的介绍。  
  
****2.3 编辑mian.py****  
现在我们可以开始编写自己的 python 程序了。用文本编辑器打开 main.py 文件。Windows 环境下可以使用记事本或者其他编辑器。Mac 和 linux 下使用你喜欢的文本编辑器即可。  
打开文件后你将看到如下的一行：

# main.py -- put your code here!

该行以 # 字符开始，意味着只是一个注释。这样的命令行不会被执行，仅为代码提供信息用在这个 main.py 加多两行，如下所示：

# main.py -- put your code here!

import pyb

pyb.LED(4).on()

第一行表明使用 pyb 模块，这个模块包含了控制开发板的所有函数和类。  
第二行打开了蓝色的 LED：先是在 pyb 模块中使用了 LED 类，创建了 LED 4 的实例，然后将其点亮  
  
****2.4  重置开发板****  
为运行这个小小的脚本，我们需要保存并关闭 main.py文件，然后在 USB 设备中退出（或者卸载），就像退出移动磁盘一样。  
当设备安全退出或解除挂载后就来到了实现功能的地方：按下板上的复位键将重置开发板并运行写好的程序。黑色的复位按键位于右边，USB 接口下。  
当按下复位键后绿色的 LED 将快速闪烁，然后蓝色的 LED 保持长亮。  
至此编写和运行第一个 MicroPython 程序就完成了，加油！You Can Do It !

1. ****点亮LEDS及python 简要概念****

ttpyboard 板上最容易实现的事情莫过于点亮板上附带的小灯。连接开发板，按照上篇教程中提到的登录方法，就可与在解释器中开始点亮 LED 灯了，代码如下：

myled = pyb.LED(1)

myled.on()

myled.off()  
这些命令将控制 LED 的亮和灭。  
这种方式不错，不过我们将尝试让其更智能化。在你擅长的文本编辑器里打开 ttpyboard 里边的 MAIN.PY 文件，写入或粘贴如下的代码。如果你是 python 新手，那么希望从此开始你能对 python 有正确的认识印象。

led = pyb.LED(2)

while True:

 led.toggle()

    pyb.delay(1000)

当你保存了文件后，tpyboard 上的红色 LED 将在约一秒后亮起。为运行脚本程序，先以 CTRL -D 进行软件复位。tpyboard 将被重启且能够看到绿色的 LED 持续闪烁。至此先恭喜你在“the army of evil robot”的路途上迈出了重要的一步！当需要关闭烦人的闪灯时，直接在终端界面按下 CLRT -C 即可。  
到目前为止代码做了什么事情呢？首先我们需要引用一些术语。Python 是一门面向对象语言（object-oriented），pyhon 中一切几乎都是类（class）和通过声明得到的类的对象（object）。通过方法（methods，也称为成员函数）我们连接类，并和对象相交互或者控制它。  
程序的第一行我们通过实例化了LED对象并命名为led。当这个对象被创建时，它唯一的参数必须在1和4之间，与开发板上四颗LED相呼应。pyb.LED这个类有三个我们使用的重要成员函数：on( ), off( ) 以及 toggle( )。另一个使用到的函数pyb.delay( ) 仅是一个简单的毫秒级别的延时。一旦我们创建了 LED 对象，while True 这个 声明将创建一个无限循环等待一秒时间的 led亮灭翻转。

****3.1 在 tpyboard 板上跳迪斯科（disco）****  
到目前我们使用了板上的单颗 LED 灯而实际上总共有四颗可供使用。我们可以为每颗 LED 灯创建一个对象并分别控制它们。我们将声明一个便于理解的列表（list）形式：

leds = [pyb.LED(i) for i in range(1,5)]

如果没有用 1，2，3，4 的数字作为 pyb.LED( ) 的形参，我们将会得到错误的信息。接下来我们将添加每个 LED 亮灭的无限循环：

n = 0

while True:

n = (n + 1) % 4

leds[n].toggle()

pyb.delay(50)

在这里，n 代表了当前的 LED 且每次循环执行后我们可以得到下一个 n 的值（求余符号%保证了 n 的值在0和3之间）。然后我们就可以控制第 n 颗 led 灯的翻转亮灭了。执行该程序将可见一排的 led 同时亮和灭。  
你可能会发现一旦停下当前脚本的运行并重新启动，开发板上的 LED 灯将从之前运行的状态突然进入到我们精心设计的迪斯科。可以通过在脚本初始化时关闭所有的 LED 灯并使用  try/finally 块的方式解决这个问题。当打入 CTRL-C，MicroPython 将产生一个 VCP 中断异常。异常通常意味着某些不对劲的东西，所以你可以通过 try:command 指令“抓取”一个异常。这种情况属于用户打断了脚本的运行，所以我们不需要抓取错误而是简单告诉 MicroPython 当我们退出时要做些什么。最终的程序块如下所示，且我们确保了所有的 LED灯 熄灭。完整的代码如下所示：

leds = [pyb.LED(i) for i in range(1,5)]

for l in leds:

l.off()

n = 0try:

while True:

n = (n + 1) % 4

leds[n].toggle()

pyb.delay(50)

finally:

for l in leds:

l.off()

****3.2 特殊的第四颗灯****  
蓝色的LED 灯比较特别。可以在让其亮灭的同时通过 iniensity( ) 的方法控制其亮度。其亮度值在 0 到 255 的值间决定。以下的脚本实现了蓝色的LED循环渐亮然后熄灭的功能。

led = pyb.LED(4)

intensity = 0

while True:

intensity = (intensity + 1) % 255

led.intensity(intensity)

pyb.delay(20)  
你可以对其他LED灯调用 instensity( ) 的方法不过其只能被熄灭或被点亮。0 值将使之熄灭而最多到达255的其他值只能点亮该LED。