

更多原创作品尽在电路城: <http://www.cirmall.com/>

Arduino 入门教程(4)—互动交通信号灯

简单回顾下 Lesson 3 的内容：






- for 循环语句的使用
- x++是什么意思？

有没有试着做上面那个课后作业呢？做出来的话，说明你已经基本掌握上面所学的东西了，如果不会也没关系，我相信，看完这个章节，前面那个问题就不攻自破了！我们这回就基于上面这个交通灯来进行一个拓展，增加一种行人按键请求通过马路的功能。当按钮被按下时，Arduino 会自动反应，改变交通灯的状态，让车停下，允许行人通过。

这个项目中，我们开始要实现 Arduino 的互动了，也会在代码学习到如何创建自己的函数。这次的代码相对长一点，耐下心来，等看完这一章，相信你能收获不少！

我们之后在所需元件中将不再重复罗列以下三样，UNO、扩展板+面包板、跳线。但是！每次都还是需要用到的。

STEP1: 所需元件

2× 5mm LED 灯	
2× 5mm LED 灯	
1× 5mm LED 灯	
6× 220 欧电阻	
1× 按钮	

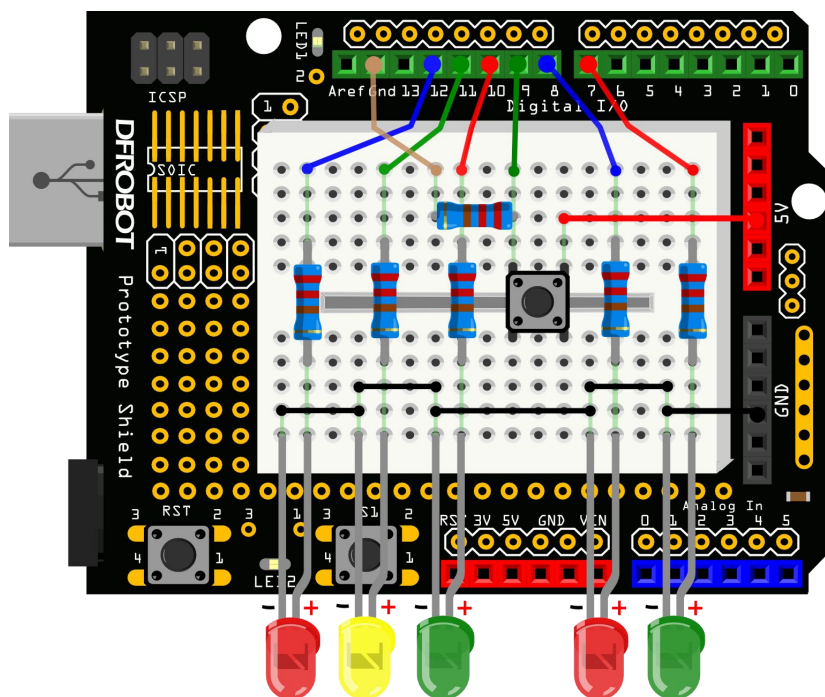
这里 5 个 LED 灯，为什么会用到了 6 个电阻呢？我们知道 5 个电阻是 LED 的限

更多原创作品尽在电路城：<http://www.cirmall.com/>

流电阻。还有一个电阻是给按钮的，它叫做下拉电阻。

STEP2：硬件连接

按下图的连线图连接你的电路。特别要注意的是，这次连线比较多，注意不要插错。下图中，面包板上标出淡绿色的不是跳线，只是为了说明纵向的孔导通，避免你插错。给 Arduino 上电前认真检查你的接线是否正确。在连线时，保持电源是断开的状态，也就是没有插 USB 线。



STEP 3：输入代码打开 Arduino IDE，输入下面这段代码。

1. `//项目三 -- 互动交通信号灯`
2. `int carRed = 12; //设置汽车灯`
3. `int carYellow = 11;`
4. `int carGreen = 10;`
5. `int button = 9; //按钮引脚`
6. `int pedRed = 8; //设置行人灯`
7. `int pedGreen = 7;`
8. `int crossTime = 5000; //允许行人通过的时间`
9. `unsigned long changeTime; //按钮按下后的时间`
- 10.

更多原创作品尽在电路城: <http://www.cirmall.com/>

```
11. void setup() {
12.     //所有 LED 设置为输出模式
13.     pinMode(carRed, OUTPUT);
14.     pinMode(carYellow, OUTPUT);
15.     pinMode(carGreen, OUTPUT);
16.     pinMode(pedRed, OUTPUT);
17.     pinMode(pedGreen, OUTPUT);
18.     pinMode(button, INPUT); //按钮设置为输入模式
19.     digitalWrite(carGreen, HIGH); //开始时, 汽车灯绿灯
20.     digitalWrite(pedRed, HIGH); //行人灯为红灯
21. }
22.
23. void loop() {
24.     int state = digitalRead(button);
25.     //检测按钮是否被按下, 并且是否距上次按下后有 5 秒的等待时间
26.     if(state == HIGH && (millis() - changeTime) > 5000){
27.         //调用变灯函数
28.         changeLights();
29.     }
30. }
31.
32. void changeLights() {
33.     digitalWrite(carGreen, LOW); //汽车绿灯灭
34.     digitalWrite(carYellow, HIGH); //汽车黄灯亮
35.     delay(2000); //等待 2 秒
36.
37.     digitalWrite(carYellow, LOW); //汽车黄灯灭
38.     digitalWrite(carRed, HIGH); //汽车红灯亮
39.     delay(1000); //为安全考虑等待 1 秒
40.
41.     digitalWrite(pedRed, LOW); //行人红灯灭
42.     digitalWrite(pedGreen, HIGH); //行人绿灯亮
43.
44.     delay(crossTime); //等待一个通过时间
45.
46.     //闪烁行人灯绿灯, 提示可过马路时间快到
```

更多原创作品尽在电路城: <http://www.cirmall.com/>

```
47.     for (int x=0; x<10; x++) {
48.         digitalWrite(pedGreen, HIGH);
49.         delay(250);
50.         digitalWrite(pedGreen, LOW);
51.         delay(250);
52.     }
53.     digitalWrite(pedRed, HIGH); //行人红灯亮
54.     delay(500);
55.
56.     digitalWrite(carRed, LOW); //汽车红灯灭
57.     digitalWrite(carYellow, HIGH); //汽车黄灯亮
58.     delay(1000);
59.     digitalWrite(carYellow, LOW); //汽车黄灯灭
60.     digitalWrite(carGreen, HIGH); //汽车绿灯亮
61.
62.     changeTime = millis(); //记录自上一次灯变化的时间
63.     //返回到主函数循环中
64. }
```

下载完成后,可以尝试按下按钮。看看是个什么的效果?我们可以看到整个变化过程是这样的——开始时,汽车灯为绿灯,行人灯为红灯,代表车行人停。一旦行人,也就是你,按下按钮,请求过马路,那么行人灯就开始由红变绿,汽车灯由绿变黄,变红。在行人通行的过程中,设置了一个过马路的时间 `crossTime`,一旦到点,行人绿灯开始闪烁,提醒行人快速过马路。闪烁完毕,最终,又回到了开始的状态,汽车灯为绿灯,行人灯为红灯。

整段代码看起来很复杂,其实理清一下思路并不难。如果你还是没有办法理不清里面变化关系的话,可以试着画一个示意图,像 Lesson 3 的课后作业那样,这样一来可能会方便你理解程序。

更多原创作品尽在电路城: <http://www.cirmall.com/>

STEP 4 : 代码回顾

通过前面两个项目,你应该能够理解这个代码的大部分内容。代码开始是一串的变量的声明,在声明中,出现了一个新名词。这里就解释一下这个新名词:

```
1. unsigned long changeTime;
```

这是一个新的变量类型。我们之前,只创建过 int 整型变量。这次要创建的是一个 long 的变量类型,它可以存放更大的数。而 unsigned long 表示不存储负数。如果我们使用一个 int 型的话,信号灯状态变化的时间,它只能存储最大 32 秒 (int 决定的),一旦出现变量溢出就会造成程序运行出现错误,所以,为了避免这样的情况,要选用能存储更大数的一个变量,并且不为负,我们就可以考虑使用 unsigned long 型。算了下,这个变量最大能存储的数累计时间可达 49 天。

怎么理解这个变量呢?我们做个这样的比方,变量好比一个盒子,盒子的空间用来存放东西的,想要放的东西一定要比盒子小,那样才放的下,否则会溢出。变量也是一样,你存储的数据一定要在变量的范围内,否则会出现溢出。所以也就是为什么我们这里要用 unsigned long,而不用 int 的原因。变量开始如果没有赋初值的话,默认从 0 开始。(对变量类型如果感兴趣可以查看该网页中的 Data Types 部分)

随即进入 setup()函数,对 LED 和按钮进行一些设置,在设置时,需要注意到的是:

```
1. pinMode(button, INPUT);
```

pinMode()函数我们已经很熟悉了,在 LESSON 1 的时候就介绍过,只是和 LED 有所不同的是,按钮要设置为 INPUT。

简单说下,我理解的 INPUT 和 OUTPUT。

我是这么理解的,INPUT 是输入的信号,是外部往控制器输入信号的。可以理解为像键盘,鼠

更多原创作品尽在电路城：<http://www.cirmall.com/>

标，他们都是给电脑（控制器）输入信号的。比如按钮，它就是典型的 INPUT 模式，它需要我们在按下按键后，控制器才能接收到外部给它的指令。

而 OUTPUT 是往外输出信号的。可以理解为电脑的显示屏，它是电脑往外输出信号的。这就是为什么 LED 使用 OUTPUT,它闪烁的过程就是向外部发出信号的过程。我们之后会用到的蜂鸣器（一个会发出声音的玩意儿），也是典型的 OUTPUT。

在 setup()函数中，先给定行人灯和汽车灯的一个初始状态：

```
1. digitalWrite(carGreen, HIGH); //开始时，汽车灯绿灯
2. digitalWrite(pedRed, LOW);    //行人灯为红灯
```

进入到的主程序中的第一句，就是来检测 button（引脚 9）的状态的：

```
1. int state = digitalRead(button);
```

此时，一个新函数出现了——digitalRead()！

digitalRead(pin)函数

Arduino 官方：[digitalRead\(\)](#)

pinMode()与 digitalWrite()、digitalRead()的关系

一开始总是在什么时候用 digitalWrite(),什么时候又该用 digitalRead()中迷惑。后来我自己把自己给说服了,我是这么理解它们的关系,前面说了INPUT和OUTPUT的区别。digitalWrite和digitalRead是基于这个的基础上说的。有了前面设置的INPUT,才能有后面的ditialRead。有了外部数据的输入,才存在控制器对输入数据的读取。同样,OUTPUT对应的后面要用digitalWrite。既然需要向外部输出信号,那么首先控制器要给它先写入信号啊——digitalWrite!我不知道有没有把他们之间的关系讲清楚,但愿说清楚了。。

我列了这么个表便于初学者参考吧！

比如：LED、蜂鸣器	比如：按键控制
pinMode(pin, OUTPUT)	pinMode(pin, INPUT)
digitalWrite(pin, HIGH/LOW)	digitalRead(pin)

更多原创作品尽在电路城：<http://www.cirmall.com/>

好了，digitalRead 后就是对按键的值进行判断了。

```
1.  if(state == HIGH && (millis() - changeTime)> 5000) {  
2.      //调用变灯函数  
3.      changeLights();  
4.  }  
5.
```

这里涉及新的语句-- if 语句。**if 语句是一种条件判断的语句，判断是否满足括号内的条件，如满足则执行花括号内的语句，如不满足则跳出 if 语句。**

if 语句格式如下：

if(表达式){

语句；

}

表达式是指我们的判断条件，通常为一些关系式或逻辑式，也可是直接表示某一数值。如果 if 表达式条件为真，则执行 if 中的语句。表达式条件为假，则跳出 if 语句。

我们代码中，第一个条件是 state 变量为 HIGH。如果按键被按下，state 就会变为 HIGH。第二个条件是 millis()的值减 changeTime 的值大于 5000。这两个条件之间有个“&&”符号。这是一种逻辑运算符，表示的含义是两者同时满足。

常用的一些逻辑运算符有：

- && —— 逻辑与（两者同时满足）
- || —— 逻辑或（两者其中一个满足）
- ! —— 逻辑非（取反，相反的情况）

```
1.  (millis() - changeTime)> 5000)
```

更多原创作品尽在电路城: <http://www.cirmall.com/>

millis()函数用法

millis()是一个函数,该函数是 Arduino 语言自有的函数,它返回值是一个时间,Arduino 开始运行到执行到当前的时间,也称之为机器时间,就像一个隐形时钟,从控制器开始运行的那一刻起开始计时,以毫秒为单位。

这里是,通过 millis()函数不断记录时间,判断两次按键之间的时间是不是大于 5 秒,如果在 5 秒之内不予反应。这样做的目的是,防止重复按键而导致的运行错误。

if 语句内只有一个函数:

```
1.  changeLights();
```

这是一个函数调用的例子。该函数单独写在了 loop()函数之外。我们需要使用的时,直接写出函数名就可以实现调用了。运行完之后,再跳回主函数。**需要特别注意的:函数调用时,函数名后面的括号不能省,要和所写的函数保持一致。**

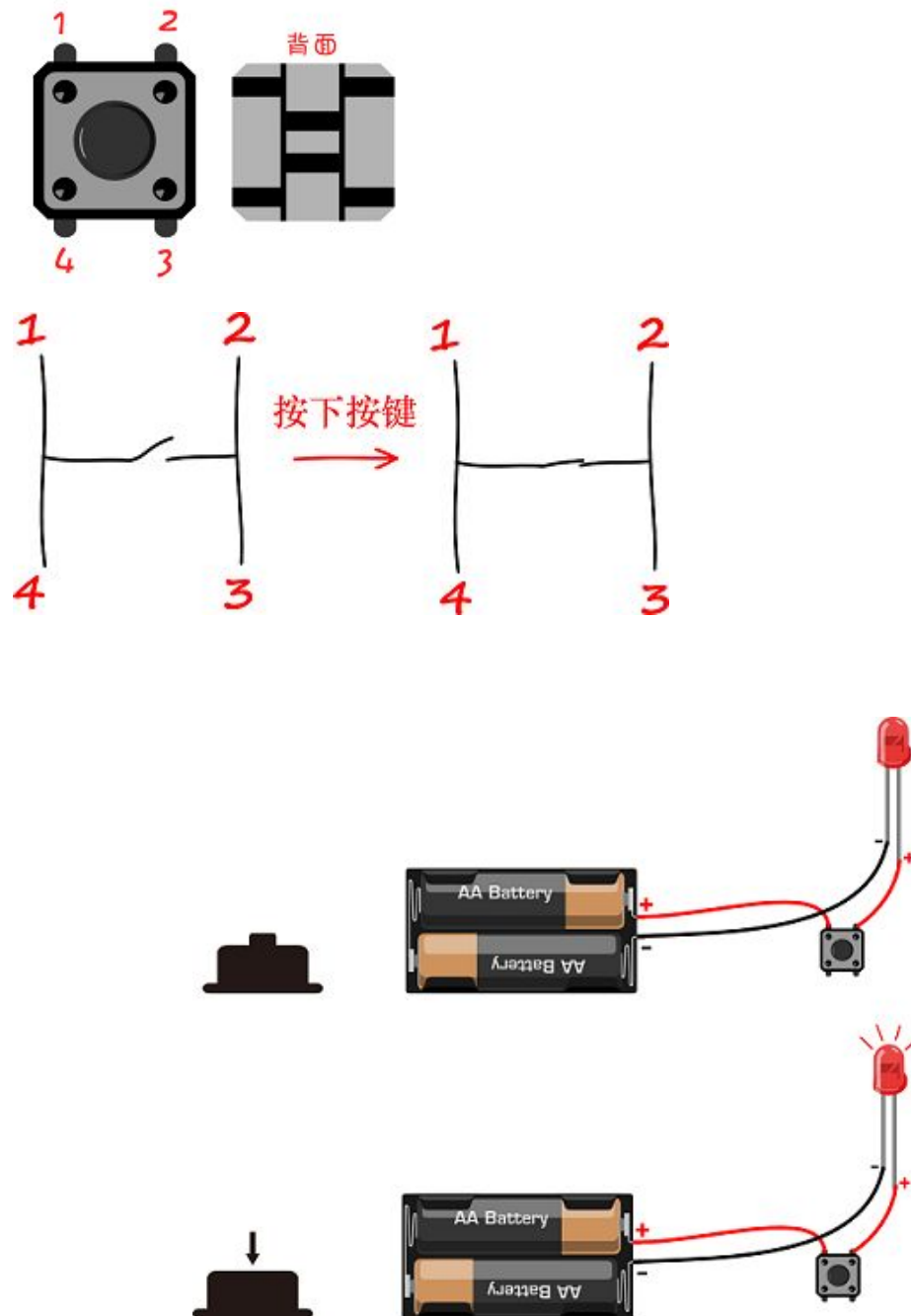
changeLights() 函数内部就不做说明了。

STEP5: 硬件回顾

按键开关

按键一共有 4 个引脚,下面分别显示了正面与背面。而再下面一张则说明了按键的工作原理。一旦按下后,左右两侧就被导通了,而上下两端始终导通。

更多原创作品尽在电路城: <http://www.cirmall.com/>



上图传达的意思是，按钮就是起到一个通断的作用。在我们这个，按钮控制数字引脚是否接高（接 5V）。按下的话，数字引脚 9 就能检测到为高电平。否则就是保持一个低电平的状态（接 GND）。