简单的类和对象

class Led{ //建立一个类

public: //public表明以下的成员变量和成员函数为公有成员，在程序中可以任意调用

Led(); //构造函数:构造函数是在类被建立的时候执行的函数

~Led(); //析构函数(destructor) 与构造函数相反，当对象结束其生命周期，如对象所在的函数已调用完毕时，系统自动执行析构函数。

void on(); //类的成员函数

void off(); //类的成员函数

};

Led::Led() //构造函数

{

Serial.println("Led Object created");

pinMode(13,OUTPUT);

}

Led::~Led() //析构函数

{

Serial.println("Led Object Deleted");

}

void Led::on() //类的成员函数

{

digitalWrite(13,HIGH);

}

void Led::off() //类的成员函数

{

digitalWrite(13,LOW);

}

void setup()

{

Serial.begin(9600);

Led myLed; //建立led类的对象，同时调用构造函数Led(),生命周期结束后调用析构函数~Led()

Serial.println("Hello,this is from setup()");

for(int i=0;i<3;i++)

{

myLed.on();

delay(1000);

myLed.off();

delay(1000);

}

//对象生命周期结束,析构函数~Led()发挥作用，输出Led Object Deleted

}

void loop()

{

}

多构造函数类

class Led{ //建立一个类

public: //public表明以下的成员变量和成员函数为公有成员，在程序中可以任意调用

Led(); //构造函数:构造函数是在类被建立的时候执行的函数（无参数时调用）

Led(int userLedPin); //构造函数:构造函数是在类被建立的时候执行的函数（有参数时调用）

~Led(); //析构函数(destructor) 与构造函数相反，当对象结束其生命周期，如对象所在的函数已调用完毕时，系统自动执行析构函数。

void on(); //类的成员函数

void off(); //类的成员函数

int ledPin=13;//类的成员变量

};

Led::Led() //构造函数

{

Serial.println("Led Object created");

pinMode(13,OUTPUT);

}

Led::Led(int userLedPin)

{

Serial.println("Led Object created");

ledPin = userLedPin;

pinMode(ledPin,OUTPUT);

}

Led::~Led() //析构函数

{

Serial.println("Led Object Deleted");

}

void Led::on() //类的成员函数

{

digitalWrite(ledPin,HIGH);

}

void Led::off() //类的成员函数

{

digitalWrite(ledPin,LOW);

}

void setup()

{

Serial.begin(9600);

Led myLed; //建立led类的对象，同时调用构造函数Led(),生命周期结束后调用析构函数~Led()

Led myLed2(7);

Serial.println("Hello,this is from setup()");

for(int i=0;i<3;i++)

{

myLed.on();

myLed2.on();

delay(1000);

myLed.off();

myLed2.off();

delay(1000);

}

//对象生命周期结束,析构函数~Led()发挥作用，输出Led Object Deleted

}

void loop()

{

}

类的封装

class Led{ //建立一个类

public: //public表明以下的成员变量和成员函数为公有成员，在程序中可以任意调用

Led(); //构造函数:构造函数是在类被建立的时候执行的函数（无参数时调用）

Led(int userLedPin);//构造函数:构造函数是在类被建立的时候执行的函数（有参数时调用）

~Led(); //析构函数(destructor) 与构造函数相反，当对象结束其生命周期，如对象所在的函数已调用完毕时，系统自动执行析构函数。

void on();//类的成员函数

void off();//类的成员函数

int getLedPin(); //获取私有成员变量，以便类外的调用

void steLedPin(int userLedPin); //利用private里的ledPin改变pinMode

private: //私有(类以外的就不可以用，只能类之中的能用)

int ledPin=13;

};

Led::Led()//构造函数

{

Serial.println("Led Object created");

pinMode(13,OUTPUT);

}

Led::Led(int userLedPin)

{

Serial.println("Led Object created");

ledPin = userLedPin;

pinMode(ledPin,OUTPUT);

}

Led::~Led()//析构函数

{

Serial.println("Led Object Deleted");

}

void Led::on()//类的成员函数

{

digitalWrite(ledPin,HIGH);

}

void Led::off()//类的成员函数

{

digitalWrite(ledPin,LOW);

}

int Led::getLedPin() //只要类内部的成员函数才能调用private里的变量

{

return ledPin;

}

void Led::steLedPin(int userLedPin)

{

ledPin=userLedPin;

pinMode(ledPin,OUTPUT);

}

void setup()

{

Serial.begin(9600);

Led myLed; //建立led类的对象，同时调用构造函数Led(),生命周期结束后调用析构函数~Led()

myLed.steLedPin(3); //把Led::Led()中定义好的13引脚变成了3

int myLedPin=myLed.getLedPin();

Serial.print("int myLedPin=");

Serial.print(myLedPin);

Led myLed2(7);

int myLed2Pin = myLed2.getLedPin();//把myLed2对象中的getLedPin返回的值给myLed2Pin，并输出

Serial.print("myLed2Pin=");

Serial.print(myLed2Pin);

Serial.println("Hello,this is from setup()");

for(int i=0;i<3;i++)

{

myLed.on();

myLed2.on();

delay(1000);

myLed.off();

myLed2.off();

delay(1000);

}

//对象生命周期结束,析构函数~Led()发挥作用，输出Led Object Deleted

}

void loop()

{

}