**int every(long period, callback)**   
每“period”个毫秒调用“callback”方法。返回定时事件的ID。

**int every(long period, callback, int repeatCount**)  
每“period”个毫秒调用“callback”方法，共“repeatCount”次。返回定时事件的ID。

**int after(long duration, callback)**  
在“period”个毫秒后调用“callback”方法一次。返回定时事件的ID。

**int oscillate(int pin, long period, int startingValue)**  
每“period”个毫秒切换数字输出引脚“pin”的状态。引脚的初始值由“startingValue”定义，应为HIGH或LOW。返回定时事件的ID。

**int oscillate(int pin, long period, int startingValue, int repeatCount)**  
每“period”个毫秒切换数字输出引脚“pin”的状态，共“repeatCount”次。引脚的初始值由“startingValue”定义，应为HIGH或LOW。返回定时事件的ID。

**int pulse(int pin, long period, int startingValue)**  
在“period”个毫秒后切换数字输出引脚“pin”的状态一次。引脚的初始值由“startingValue”定义，应为HIGH或LOW。返回定时事件的ID。

**int stop(int id)**  
停止正在运行的定时事件。返回定时事件的ID。

**int update()**  
必须在“loop”中调用。Must be called from 'loop'. 它会服务所有与定时器相关的事件。

void setup() 执行一次

循环执行void loop()

#include "Timer.h"

Timer t;

int ledEvent;

void setup()

{

Serial.begin(9600);

int tickEvent = t.every(2000, doSomething);//本语句执行以后2000毫秒执行回调函数

Serial.print("2 second tick started id=");

Serial.println(tickEvent);

pinMode(13, OUTPUT);

ledEvent = t.oscillate(13, 50, HIGH);

Serial.print("LED event started id=");

Serial.println(ledEvent);

int afterEvent = t.after(10000, doAfter);

Serial.print("After event started id=");

Serial.println(afterEvent);

}

void loop()

{

t.update();

}

void doSomething()

{

Serial.print("2 second tick: millis()=");

Serial.println(millis());

}

void doAfter()

{

Serial.println("stop the led event");

t.stop(ledEvent);

t.oscillate(13, 500, HIGH, 5);

}

你最多可以在一个定时器上附加10个事件。