任务：任务一旦完成就不能重新启动它。因此，除非重新创建任务，否则没有方法重新运行任务。

任务ID：Task的属性Id。Id为只读属性，是唯一的，无序的。

CurrentId属性获取当前执行的任务的ID。

例：程序task\_test2

// Copyright 2016.刘珅珅

// author：刘珅珅

// 任务ID

using *System*;

using *System*.*Collections*.*Generic*;

using *System*.*Linq*;

using *System*.*Text*;

using *System*.*Threading*;

using *System*.*Threading*.*Tasks*;

namespace task\_test2

{

class TaskTest

{

// 任务函数

static void MyTask()

{

*Console*.*WriteLine*("MyTask() #" + *Task*.*CurrentId* + " starting.");

for (int i = 0; i < 10; ++i)

{

*Thread*.*Sleep*(500);

*Console*.*WriteLine*("In MyTask() #" + *Task*.*CurrentId* + ", count is " + i);

}

*Console*.*WriteLine*("MyTask #" + *Task*.*CurrentId* + " terminating");

}

static void Main(string[] args)

{

*Console*.*WriteLine*("Main thread starting.");

*Task* task1 = new *Task*(MyTask);

*Task* task2 = new *Task*(MyTask);

task1.*Start*();

task2.*Start*();

*Console*.*WriteLine*("Task ID for task1 is " + task1.*Id*);

*Console*.*WriteLine*("Task ID for task2 is " + task2.*Id*);

// 任务执行期间，保持Main()运行不退出

for (int i = 0; i < 60; ++i)

{

*Console*.*Write*(".");

*Thread*.*Sleep*(100);

}

*Console*.*WriteLine*("Main thread ending.");

}

}

}

输出结果：

MyTask() #2 starting.

.....In MyTask() #2, count is 0

In MyTask() #1, count is 0

....In MyTask() #2, count is 1

.In MyTask() #1, count is 1

....In MyTask() #2, count is 2

.In MyTask() #1, count is 2

....In MyTask() #2, count is 3

.In MyTask() #1, count is 3

....In MyTask() #2, count is 4

.In MyTask() #1, count is 4

....In MyTask() #2, count is 5

In MyTask() #1, count is 5

.....In MyTask() #2, count is 6

In MyTask() #1, count is 6

.....In MyTask() #2, count is 7

In MyTask() #1, count is 7

.....In MyTask() #2, count is 8

In MyTask() #1, count is 8

.....In MyTask() #1, count is 9

MyTask #1 terminating

.In MyTask() #2, count is 9

MyTask #2 terminating

.........Main thread ending.

从结果中可以看出，主线程，task1和task2是并行执行，互不干扰。