**实验6.2 并行编程**

实验相关材料位于：<ftp://192.168.40.100/>下的“**--- 梁其洋**”目录下（可以通过搜索名字找到）

作业提交到：[ftp://192.168.40.14/梁其洋/网络编程\*\*班/实验\*](ftp://192.168.40.14/梁其洋/网络编程**班/实验*)

要求：1. 提交时**将本文档重新命名**，文档命名规则：学号+姓名+实验\*，例如，**021300614400 张三 实验\*.docx （一定要学号在前姓名在后，方便自动排序！）**

2. 你们没有FTP文件的删除权限，**如果需要提交新版本，在姓名后加序号后提交即可，例如，021300614400张三（1）实验\*.docx**

实验6-3 Task的Wait、WaitAny、WaitAll方法

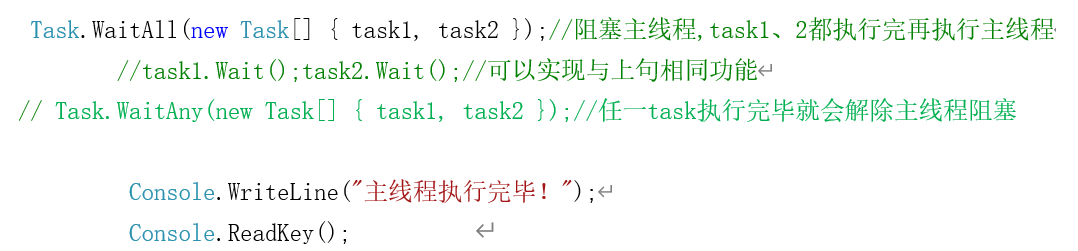
实验目的：如果想让所有的线程执行完毕(或者任一线程执行完毕)时，立即解除对主线程的阻塞，可以利用Task类的Wait、WaitAny、WaitAlltask方法控制线程阻塞，将异步执行的任务变为同步执行。

实验步骤：

（1）新建一个控制台应用程序，

static void Main(string[] args)

{

}

程序运行截图（保留两张截图，包括取消waitAny注释后的）：

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading;

using System.Threading.Tasks;

namespace sy6\_4

{

internal class Program

{

static void Main(string[] args)

{

Task task1 = new Task(() =>

{

Thread.Sleep(500);

Console.WriteLine("线程1执行完毕!");

});

task1.Start();

Task task2 = new Task(() =>

{

Thread.Sleep(1000);

Console.WriteLine("线程2执行完毕!");

});

task2.Start();

// Task.WaitAll(

// new Task[]

// {

// task1,

// task2

// });

Task.WaitAny(new Task[] { task1, task2 });

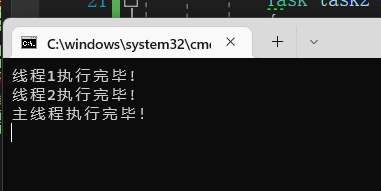
Console.WriteLine("主线程执行完毕！");

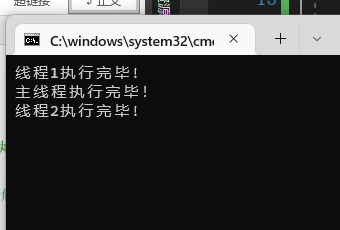
Console.ReadKey();

}

}

}





实验6-4 带并行选项的Parallel.For方法 （对应教材例6-2）

实验目的：验证并不是并行度越大，效率越高

实验步骤：

（1）新建一个WPF应用程序，在MainWindow.xaml中添加

<DockPanel>

<Border DockPanel.Dock="Top">

<TextBlock Text="Parallel.For方法自定义选项基本用法" />

</Border>

<Border DockPanel.Dock="Bottom">

<StackPanel Orientation="Horizontal" HorizontalAlignment="Center" Height="20">

<TextBlock Text="创建的对象个数："/>

<TextBox Name="textBox1" Text="100000" Width="80"/>

<Button Name="btnStart" Width="50" Margin="10 0 0 0" Content="执行" Click="btnStart\_Click" Height="20" VerticalAlignment="Top"/>

</StackPanel>

</Border>

<ScrollViewer>

<StackPanel Background="White" TextBlock.LineHeight="20">

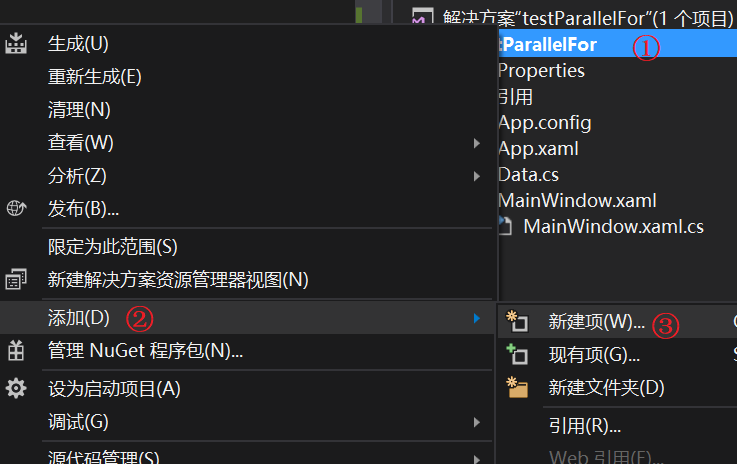
<TextBlock x:Name="textBlock1" Margin="5" TextWrapping="Wrap"/>

</StackPanel>

</ScrollViewer>

</DockPanel>

（2）在工程里添加一个新类Data.cs



在Data类里添加两个属性：

*class Data*

*{*

public string Name { get; set; }

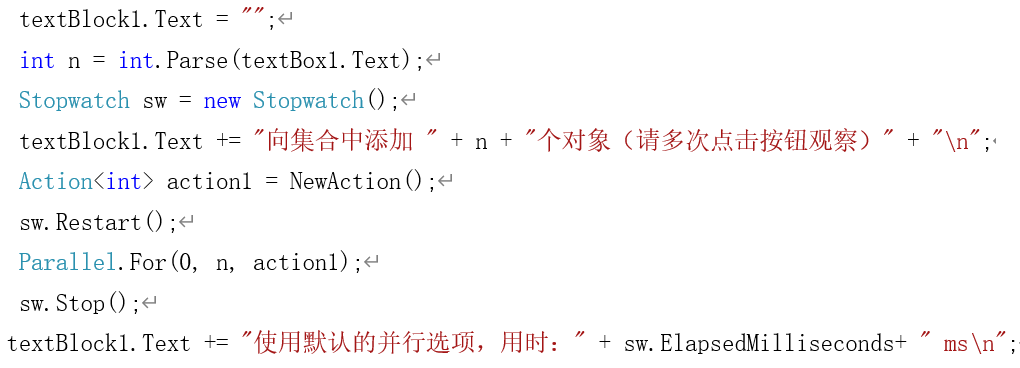
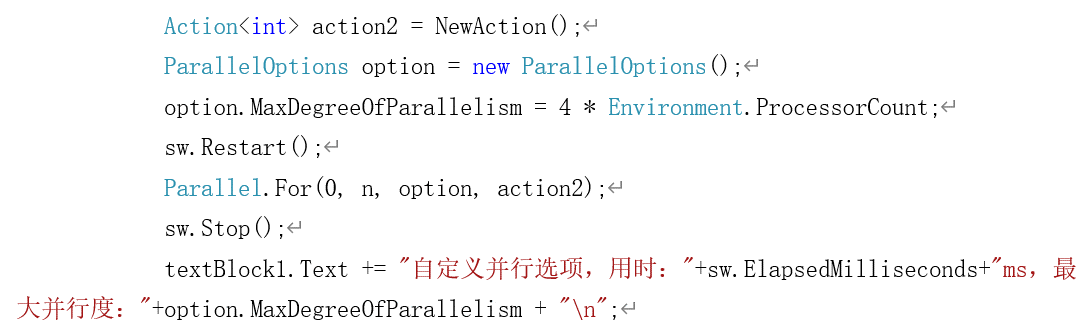
public int Number { get; set; }

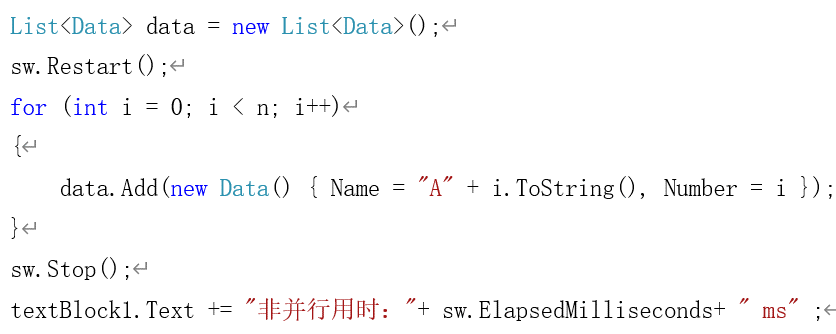
*}*

（3）在MainWindow.xaml.cs文件中实现btnStart按钮的click事件：

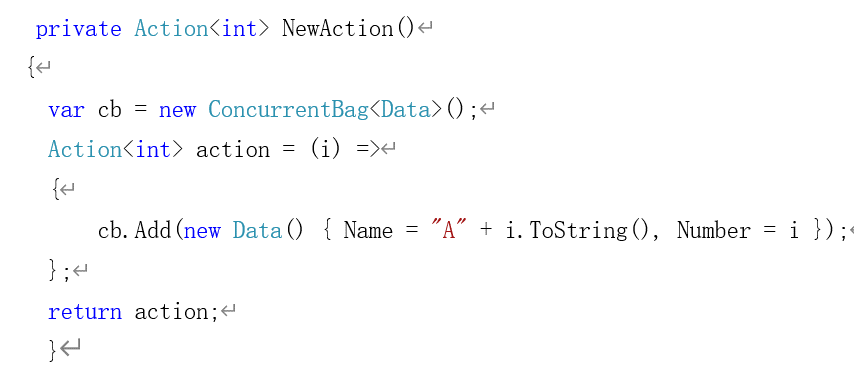
private void btnStart\_Click(object sender, RoutedEventArgs e)

{



（4）添加一个委托NewAction



程序运行截图：

using System;

using System.Collections.Concurrent;

using System.Collections.Generic;

using System.Diagnostics;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Navigation;

using System.Windows.Shapes;

namespace sy6\_3

{

/// <summary>

/// Interaction logic for MainWindow.xaml

/// </summary>

public partial class MainWindow : Window

{

public MainWindow()

{

InitializeComponent();

}

private void btnStart\_Click(object sender, RoutedEventArgs e)

{

textBlock1.Text = "";

int n = int.Parse(textBox1.Text);

Stopwatch sw = new Stopwatch();

textBlock1.Text += "向集合中添加 " + n + " 个对象" + "\n";

Action<int> action1 = NewAction();

sw.Restart();

Parallel.For(0, n, action1);

sw.Stop();

textBlock1.Text += "使用默认的并行选项，用时：" + sw.ElapsedMilliseconds + " ms\n";

Action<int> action2 = NewAction();

ParallelOptions options = new ParallelOptions();

options.MaxDegreeOfParallelism = 4 \* Environment.ProcessorCount;

sw.Restart();

Parallel.For(0, n, action2);

sw.Stop();

textBlock1.Text += "自定义并行选项，用时：" + sw.ElapsedMilliseconds + " ms，最大并行度：" + options.MaxDegreeOfParallelism + "\n";

List<Data> datas = new List<Data>();

sw.Restart();

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

{

datas.Add(new Data()

{

Name = "A"+i.ToString(),

Number = i

});

}

sw.Stop();

textBlock1.Text += "非并行用时：" + sw.ElapsedMilliseconds + " ms";

}

private Action<int> NewAction()

{

ConcurrentBag<Data> cb = new ConcurrentBag<Data>();

Action<int> action = (i) =>

{

cb.Add(new Data()

{

Name = "A" + i.ToString(),

Number = i

}

);

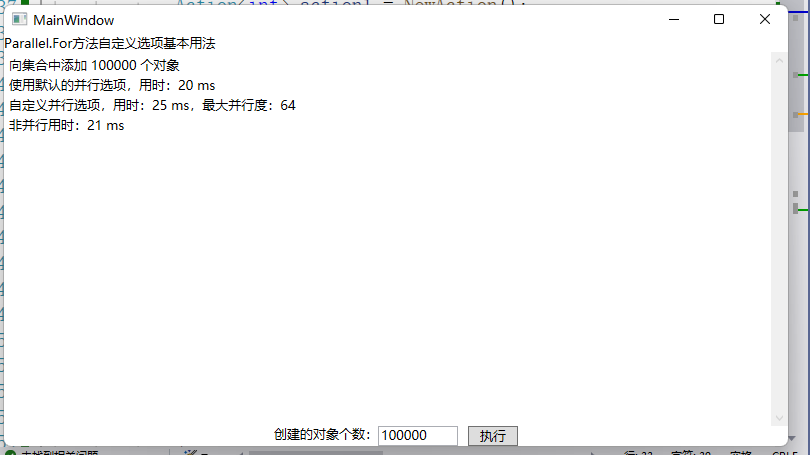
};

return action;

}

}

}



**6.1 补充作图的结果**

