**实验5异步编程**

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，**517300614400 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，517300614400张三（1）实验\*.docx**

**5.1 Lambda表达式**

实验步骤：

（1）新建一个名为example1的WPF工程，在MainWindow.xaml文件中添加以下代码：

<DockPanel>

<Label DockPanel.Dock="Top" Content="Lambda表达式基本用法" />

<Border DockPanel.Dock="Bottom" >

<Button Name="btn" Width="60" Content="运行" Click="btn\_Click"/>

</Border>

<ScrollViewer>

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

<TextBlock x:Name="textBlock1" Margin="0 10 0 0" TextWrapping="Wrap"/>

</StackPanel>

</ScrollViewer>

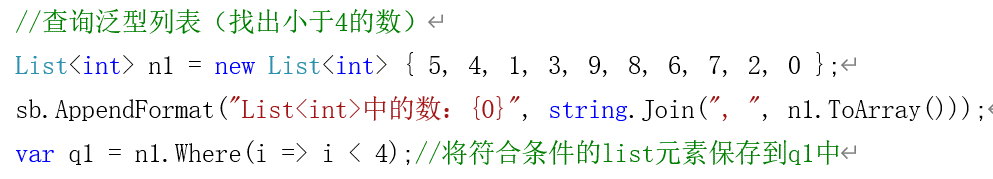
</DockPanel>

（2）在btn的click事件中添加以下代码：

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

{

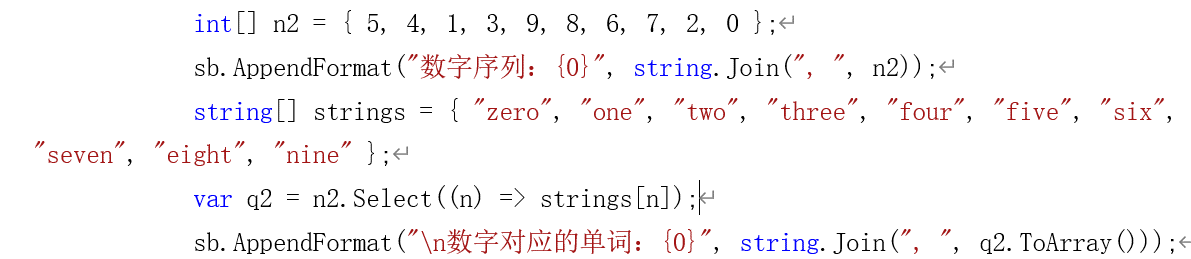
StringBuilder sb = new StringBuilder();



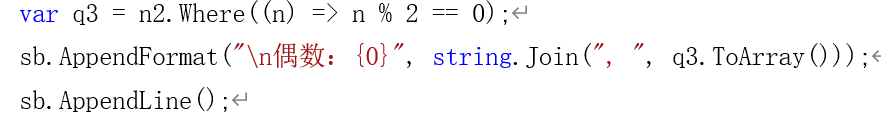
sb.AppendFormat("\n小于4的数：{0}", string.Join(", ", q1.ToArray()));

sb.AppendLine("\n");

//查询数组（找出每个数字对应的英文单词）



//查询数组（找出所有偶数）



textBlock1.Text = sb.ToString();

}

程序截图：

<Window x:Class="sy5\_1.MainWindow"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:local="clr-namespace:sy5\_1"

mc:Ignorable="d"

Title="MainWindow" Height="450" Width="800">

<DockPanel Height="433" VerticalAlignment="Top">

<Label DockPanel.Dock="Top" Content="Lambda表达式基本语法"></Label>

<Border DockPanel.Dock="Bottom">

<Button Name="btn" Width="60" Content="运行" Click="btn\_Click"></Button>

</Border>

<ScrollViewer>

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

<TextBlock x:Name="textBlock1" TextWrapping="Wrap" Margin="0 0 0 0"></TextBlock>

</StackPanel>

</ScrollViewer>

</DockPanel>

</Window>

using System;

using System.Collections.Generic;

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 sy5\_1

{

/// <summary>

/// Interaction logic for MainWindow.xaml

/// </summary>

public partial class MainWindow : Window

{

public MainWindow()

{

InitializeComponent();

}

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

{

StringBuilder sb = new StringBuilder();

// 查询泛型列表

List<int> n1 = new List<int>{5,4,1,3,9,8,6,7,2,0};

sb.AppendFormat("List<int>中的数：{0}", string.Join(",", n1.ToArray()));

var q1 = n1.Where(i => i < 4);

sb.AppendFormat("小于4的数：{0}", string.Join(",", q1));

sb.AppendLine("\n");

// 查询数组

int[] n2 = {5, 4, 1, 3, 9, 8, 6, 7, 2, 0};

sb.AppendFormat("数字序列:{0}", string.Join(",", n2));

string[] strings = {"zero", "one", "two", "three", "four", "five", "six", "seven", "eight", "nine"};

var q2 = n2.Select(n => strings[n]);

sb.AppendFormat("数字对应的单词：{0}", string.Join(",", q2));

// 查询数组

var q3 = n2.Where(n => n % 2 == 0);

sb.AppendFormat("偶数：{0}", string.Join(",", q3));

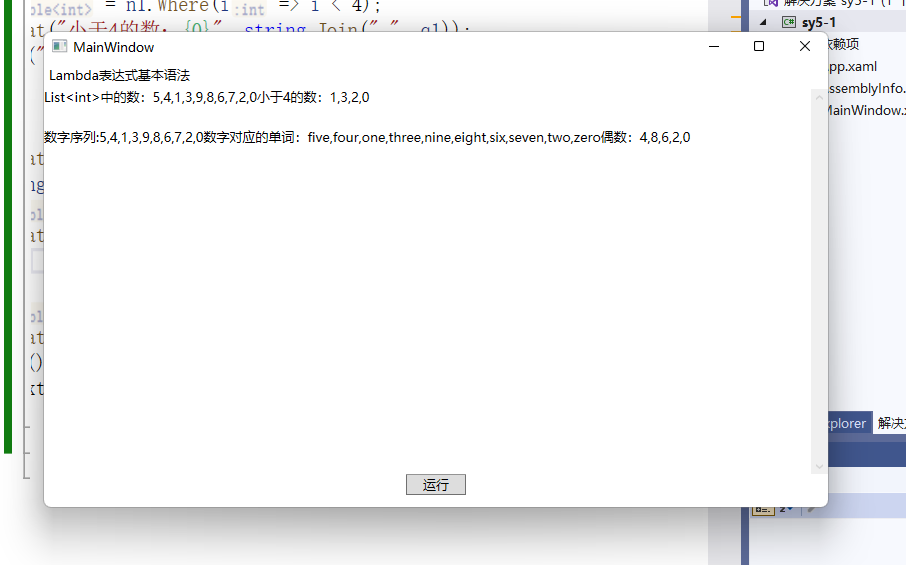
sb.AppendLine();

textBlock1.Text = sb.ToString();

}

}

}



**5.2 Action和Func委托**

实验步骤：

（1）替换xaml文件里的标签内容：

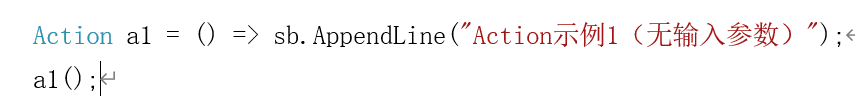
<Label DockPanel.Dock="Top" Content="Action和Func基本用法"/>

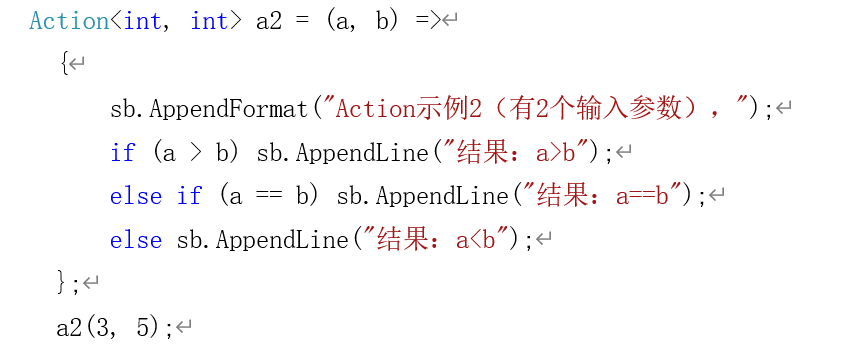
（2）在btn的click事件中添加以下代码：

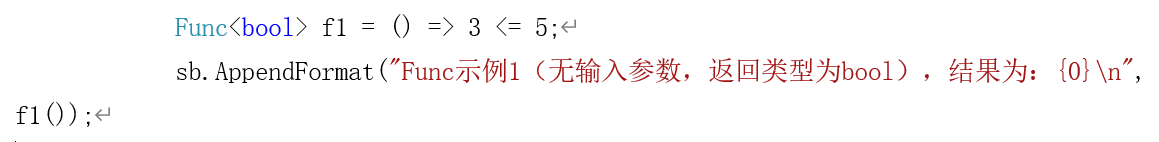
private void btn\_Click(object sender, RoutedEventArgs e)

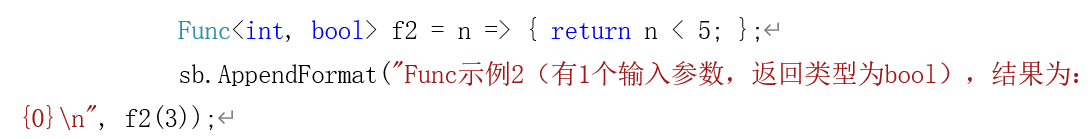
{

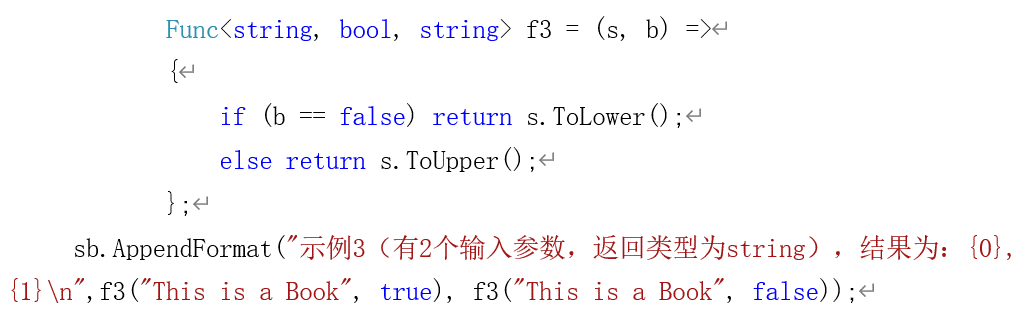
StringBuilder sb = new StringBuilder();













textBlock1.Text = sb.ToString();

}

}

程序截图：

<Window x:Class="sy5\_2.MainWindow"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:local="clr-namespace:sy5\_2"

mc:Ignorable="d"

Title="MainWindow" Height="450" Width="800">

<DockPanel Height="430" VerticalAlignment="Top">

<Label DockPanel.Dock="Top" Content="Action和Func基本用法"/>

<Border DockPanel.Dock="Bottom">

<Button Name="btn" Width="60" Content="运行" Click="btn\_Click"></Button>

</Border>

<ScrollViewer>

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

<TextBlock x:Name="textBlock1" TextWrapping="Wrap" Margin="0 0 0 0"></TextBlock>

</StackPanel>

</ScrollViewer>

</DockPanel>

</Window>

using System;

using System.Collections.Generic;

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 sy5\_2

{

/// <summary>

/// Interaction logic for MainWindow.xaml

/// </summary>

public partial class MainWindow : Window

{

public MainWindow()

{

InitializeComponent();

}

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

{

StringBuilder sb = new StringBuilder();

Action a1 = () => sb.AppendLine("Action示例1（无输入参数）");

a1();

Action<int, int> a2 = (a, b) =>

{

sb.AppendFormat("Action示例2(有2个输入参数)");

if (a > b)

{

sb.AppendLine("结果 a>b");

}

else if (a == b)

{

sb.AppendLine("结果：a==b");

}

else

{

sb.AppendLine("结果：a<b");

}

};

a2(3, 5);

Func<bool> f1 = () => 3 <= 5;

sb.AppendFormat("Func示例1(无序输入参数，返回类型bool)，结果为：{0}", f1());

Func<int, bool> f2 = n =>

{

return n < 5;

};

sb.AppendFormat("Func示例2（有一个输入参数，返回类型bool）结果为：{0}", f2(3));

Func<string, bool, string> f3 = (s, b) =>

{

if (b == false)

{

return s.ToLower();

}

else

{

return s.ToUpper();

}

};

sb.AppendFormat("示例3（有2个输入参数，返回类型string）,结果为：{1}\n",f3("This is a Book",true),f3("This is a Book",false));

string[] words = { "orange","apple","Article" };

var q = words.Select(a=>a.ToUpper());

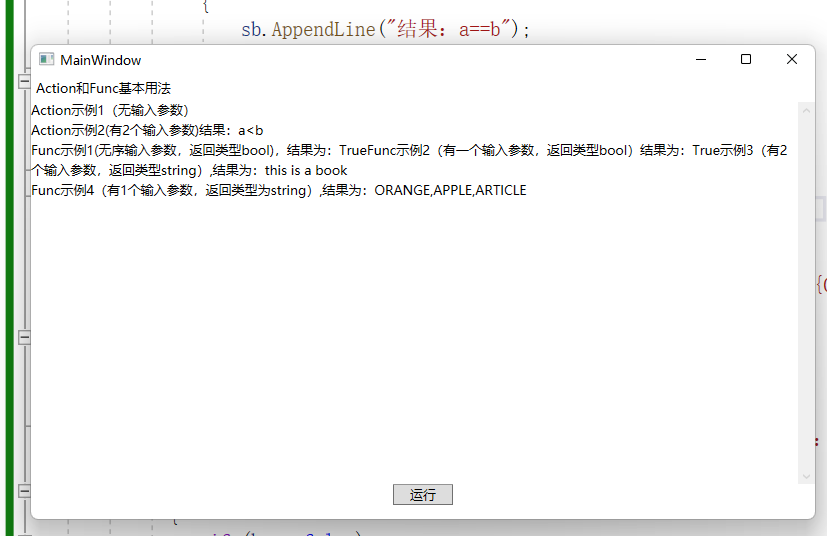
sb.AppendFormat("Func示例4（有1个输入参数，返回类型为string）,结果为：{0}",string.Join(",",q));

textBlock1.Text = sb.ToString();

}

}

}



**5.3 async和await**

实验步骤：

（1）替换xaml文件里DockPanel的内容：

<DockPanel>

<Label DockPanel.Dock="Top" Content="async和await基本用法" />

<Border DockPanel.Dock="Bottom" >

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

<Button Name="btnStart" Width="70" Content="启动任务" Click="btnStart\_Click"/>

<Button Name="btnStop" Width="70" Margin="20 0 0 0" Content="终止任务" Click="btnStop\_Click"/>

</StackPanel>

</Border>

<ScrollViewer>

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

<TextBlock x:Name="textBlock1" Margin="0 10 0 0" TextWrapping="Wrap"/>

</StackPanel>

</ScrollViewer>

</DockPanel>

（2）在窗体初始化函数之前添加一行代码，创建取消源：

private System.Threading.CancellationTokenSource cts;

*public MainWindow()*

*{*

*InitializeComponent();*

*}*

（3）将btnStart按钮的click事件改为异步事件处理程序，并添加以下代码：

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

{

cts = new System.Threading.CancellationTokenSource();

textBlock1.Text = "开始执行任务......";

try{



textBlock1.Text += "\n任务1执行完毕";



textBlock1.Text += "\n任务2（计算1到1000的和）结果为：" + sum;



textBlock1.Text += string.Format("\n任务3（求39除以8的商和余数）结果

为：{0},{1}\n", a.Item1, a.Item2);

图形用户界面, 文本, 应用程序

描述已自动生成

}

catch (OperationCanceledException)

{

textBlock1.Text += "\n任务被取消";

}

}

（4）在btnStop按钮的click事件中添加以下代码：

cts.Cancel();

（5）添加几个方法：

图形用户界面, 文本

描述已自动生成

图形用户界面, 文本, 应用程序

描述已自动生成

图形用户界面, 文本, 应用程序

描述已自动生成

图形用户界面, 文本, 应用程序, 电子邮件

描述已自动生成

程序截图：

using System;

using System.Collections.Generic;

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 sy5\_3

{

/// <summary>

/// Interaction logic for MainWindow.xaml

/// </summary>

public partial class MainWindow : Window

{

private System.Threading.CancellationTokenSource cts;

public MainWindow()

{

InitializeComponent();

}

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

{

cts = new System.Threading.CancellationTokenSource();

textBlock1.Text = "开始执行任务.......";

try

{

await Task.Run(()=>Method1(),cts.Token);

textBlock1.Text += "\n任务1执行完毕";

var sum = await Task.Run(() => Method2(), cts.Token);

textBlock1.Text += "\n任务2(计算1到1000的和)结果为：" + sum;

var a = await Task.Run(()=>Method3(39,8),cts.Token);

textBlock1.Text += string.Format("\n任务3(求39除以8的商和余数)的结果为：{0},{1}\n",a.Item1,a.Item2);

while (true)

{

textBlock1.Text += await Task.Run(() => Method1("a"), cts.Token);

textBlock1.Text += await Task.Run(() => Method1("b"), cts.Token);

}

}

catch (OperationCanceledException)

{

textBlock1.Text += "\n任务被取消";

}

}

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

{

cts.Cancel();

}

public void Method1()

{

System.Threading.Thread.Sleep(100);

}

public string Method1(string s)

{

System.Threading.Thread.Sleep(100);

return s;

}

public int Method2()

{

var range = Enumerable.Range(1, 1000);

int n = range.Sum();

return n;

}

public Tuple<int,int> Method3(int n1,int n2)

{

var result = Tuple.Create(n1 / n2,n1%n2);

System.Threading.Thread.Sleep(100);

return result;

}

}

}

