**武汉大学计算机学院**

**本科生实验报告**

专 业 名 称 ：计算机科学与技术

课 程 名 称 ：Windows原理与应用

指 导 教 师 ： 刘敏忠

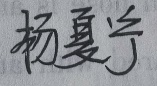
学 号 ：2022302111073

姓 名 ： 杨夏宁

二○二四年九月

**郑 重 声 明**

本人呈交的实验报告，是在指导老师的指导下，独立进行实验工作所取得的成果，所有数据、图片资料真实可靠。尽我所知，除文中已经注明引用的内容外，本实验报告不包含他人享有著作权的内容。对本实验报告做出贡献的其他个人和集体，均已在文中以明确的方式标明。本实验报告的知识产权归属于培养单位。



本人签名： 日期： 2024.09.23

摘 要

本实验旨在实现 Windows 操作系统中的事件处理机制和消息发送接收处理机制。通过创建一个包含四个项目的解决方案First：CopyDataStruct（类库）、SenderA（消息发送者）、ReceiverB（WinForm 消息接收者）和 ReceiverC（WPF 消息接收者），实现了跨进程消息传递。

在CopyDataStruct项目中定义了消息结构体 COPYDATASTRUCT，用于封装和传递消息数据。在SenderA项目中，通过引用CopyDataStruct项目，使用FindWindow和SendMessage函数实现消息发送功能。用户可以在文本框中输入目标窗口名称和消息内容，并通过按钮触发消息发送。

在ReceiverB项目中，通过重载DefWndProc方法处理接收到的消息，并在列表框中显示消息内容及接收时间。在ReceiverC项目中，通过在窗口加载事件中定义钩子函数处理消息，并在列表框中显示消息内容及接收时间。

Windows 的事件处理机制是基于消息的操作系统，用户的任何操作都被看作一个事件，操作系统会自动将该事件转换为相应的消息并投入应用程序的消息队列等待处理。消息发送和接收机制通过SendMessage和PostMessage函数实现。

实验结果表明，SenderA项目能够成功发送消息到ReceiverB和ReceiverC项目，并且接收者能够正确显示接收到的消息内容及时间。该实验验证了基于COPYDATASTRUCT结构体的跨进程消息传递的可行性和有效性。

**目录**

[课 程 名 称 ：Windows原理与应用 1](#_Toc178014400)

[摘 要 3](#_Toc178014401)

[1实验内容 5](#_Toc178014402)

[1.1实验具体内容 5](#_Toc178014403)

[1.2功能性要求 5](#_Toc178014404)

[1.3业务逻辑要求 6](#_Toc178014405)

[1.4上机实验所用平台 6](#_Toc178014406)

[2项目分析 7](#_Toc178014407)

[2.1CopyDataStruct 7](#_Toc178014408)

[2.2SenderA 8](#_Toc178014409)

[2.2.1声明User32.dll中的函数 8](#_Toc178014410)

[2.2.2定义窗体布局和控件元素 8](#_Toc178014411)

[2.2.3实现消息的封装及消息发送 9](#_Toc178014412)

[2.3ReceiverB 11](#_Toc178014413)

[2.3.1窗口设计 11](#_Toc178014414)

[2.3.2重载实现消息处理函数 12](#_Toc178014415)

[2.4ReceiverC 12](#_Toc178014416)

[2.4.1窗口控件 12](#_Toc178014417)

[2.4.2钩子函数的定义与实现 12](#_Toc178014418)

[3重要小操作 13](#_Toc178014419)

[4调试结果 14](#_Toc178014420)

[5实验总结 15](#_Toc178014421)

# 1实验内容

## 1.1实验具体内容

（1）准备工作：新建解决方案，在解决方案下新建四个项目.其中第一个项目为CopyDataStruct，其输出类型为类库;第二个项目为SenderA，其输出类型为windows窗体应用(.NET Framework)或WPF应用(.NET Framework),第三个项目为ReceiverB, 其输出类型为windows窗体应用(.NET Framework)，第四个项目为ReceiverC, 其输出类型为WPF窗体应用(.NET Framework)；

（2）定义消息类型和消息结构：在项目CopyDataStruct中定义消息类型，定义消息结构体；

（3）实现消息发送者程序A：引用CopyDataStruct，声明User32.dll中的函数FindWindow，声明User32.dll中的函数SendMessage，定义窗体布局和控件元素，实现消息的封装及消息发送；

（4）实现winform窗体应用程序B：引用CopyDataStruct，重载实现winform窗体消息处理函数DefWndProc；

（5）实现wpf窗体应用程序C：引用CopyDataStruct，在wpf窗体加载事件中定义钩子函数，并实现该钩子函数。

## 1.2功能性要求

（1）窗体应用程序A通过文本输入框接收用户输入，通过文本框输入目标窗体应用B或C，并通过按钮触发消息发送给相应窗体应用程序B或C；

（2）窗体应用程序B为winform应用程序，在收到消息之后，在列表框中显示消息内容及接收时间；

（3）窗体应用程序C为wpf应用程序，在收到消息之后，在列表框中显示消息内容及接收时间。

## 1.3业务逻辑要求

（1）按下“触发消息1”按钮后，发送消息，窗体winform程序接收消息并显示内容；

（2）按下“触发消息2”按钮后，发送消息，窗体wpf程序接收消息并显示内容。

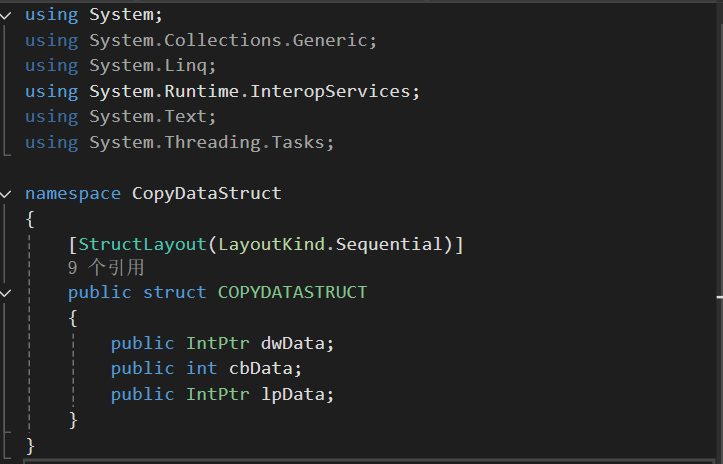
## 1.4上机实验所用平台

实验在 Windows 11操作系统上进行，使用 Visual Studio 2022作为开发环境，编程语言为 C#，目标框架为 .NET Framework 4.7.2。

硬件环境：CPU Intel i5-9300H ；显卡 NIVIDIA GeForce GTX 1650 4GB ；内存24G(2667 MHz)。

# 2项目分析

## 2.1CopyDataStruct

CopyDataStruc项目是一个类库项目，主要用于定义消息结构体COPYDATASTRUCT。该结构体用于封装和传递消息数据。

在每一个项目中都声明了常数private const int WM\_COPYDATA = 0x004A;

这一行代码定义了一个常量WM\_COPYDATA，其值为0x004A。在 WM\_COPYDATA是一个消息常量，用于在进程间传递数据。

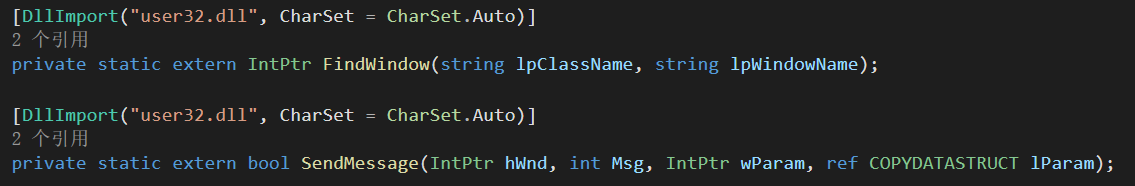
## 2.2SenderA

SenderA项目是一个Windows窗体应用程序，负责发送消息。用户可以在文本框中输入目标窗口名称和消息内容，并通过按钮触发消息1或触发消息2发送。

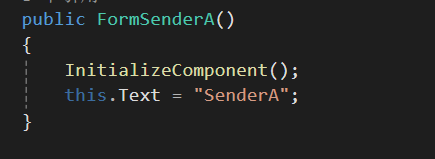
主要语句如下：

#### 2.2.1声明User32.dll中的函数

声明User32.dll中的函数FindWindow，声明User32.dll中的函数SendMessage。



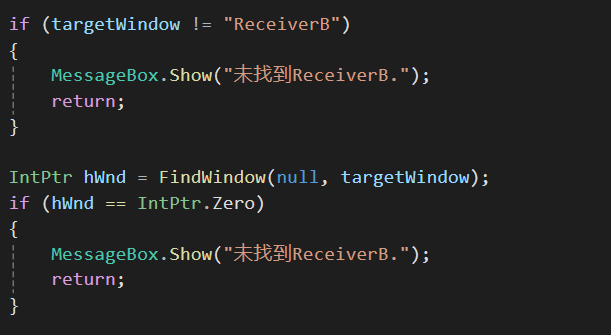
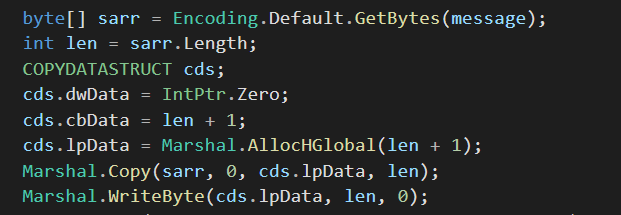
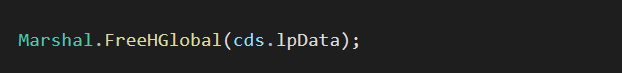
#### 2.2.2定义窗体布局和控件元素

通过FormSenderA()函数初始化窗口，并设置窗口名称为SenderA。在设计图中使用两个TextBox分别命名为txtTargetWindow与txtMessage，并使用两个Label分别标识为“发送窗口”与“发送内容”，之后使用两个Button分别标识为“触发消息1”与“触发消息2”，命名为btnSend1与btnSend2，分别负责给WinForm与WPF发送消息。



#### 2.2.3实现消息的封装及消息发送

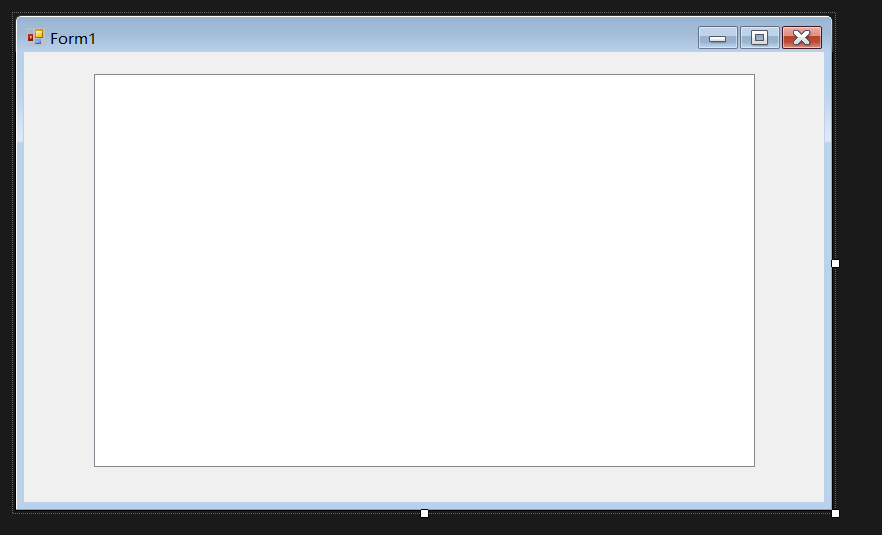
这部分内容主要在btnSend1\_Click()与btnSend2\_Click()中实现，其中的道理相同，故不重复赘述，此处以btnSend1\_Click()为例。

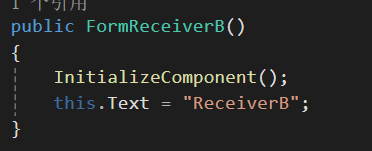
1. 获取目标窗口名称和消息内容
2. 查找目标窗口的句柄
3. 将消息内容转换为字节数组
4. 封装消息数据
5. 发送消息
6. 释放分配的内存

## 2.3ReceiverB

ReceiverB项目是一个Windows窗体应用程序，负责接收消息并在列表框中显示消息内容及接收时间。通过重载DefWndProc方法处理接收到的消息。

#### 2.3.1窗口设计

在设计图中使用一个ListView控件并命名为lstMessages，显示模式为List。

同时在对应的代码中，通过初始化将窗口名称命名为ReceiverB。

#### 2.3.2重载实现消息处理函数

## 2.4ReceiverC

ReceiverC项目是一个 WPF 窗体应用程序，负责接收消息并在列表框中显示消息内容及接收时间。通过在窗口加载事件中定义钩子函数处理消息。

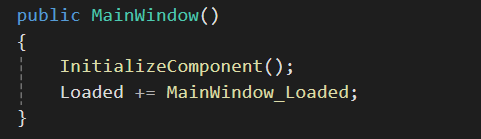
#### 2.4.1窗口控件

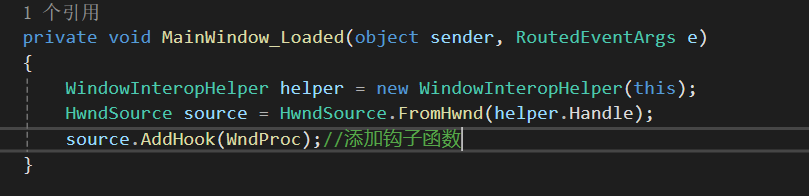
在xmal里将窗口的Title命名为ReceiverC并且使用ListBox控件，命名为lstMessages来接受信息并显示。

#### 2.4.2钩子函数的定义与实现

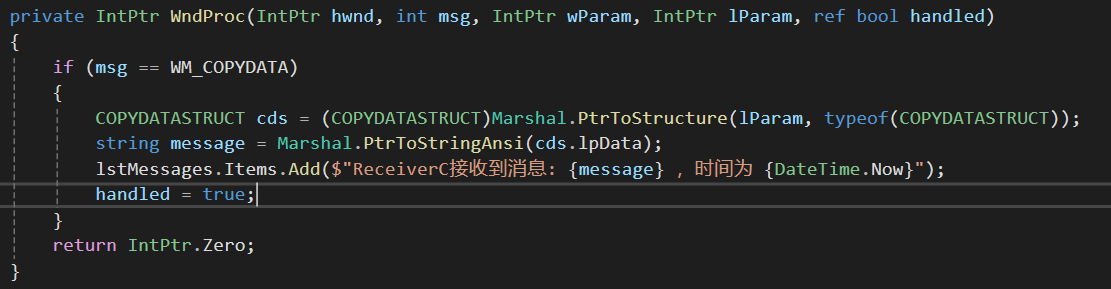
①订阅Loaded事件

在MainWindow构造函数中，订阅了Loaded事件，以便在窗口加载完成后执行特定操作。

②添加钩子函数

在MainWindow\_Loaded方法中，获取窗口的句柄，并通过HwndSource添加钩子函数WndProc。

③实现钩子函数

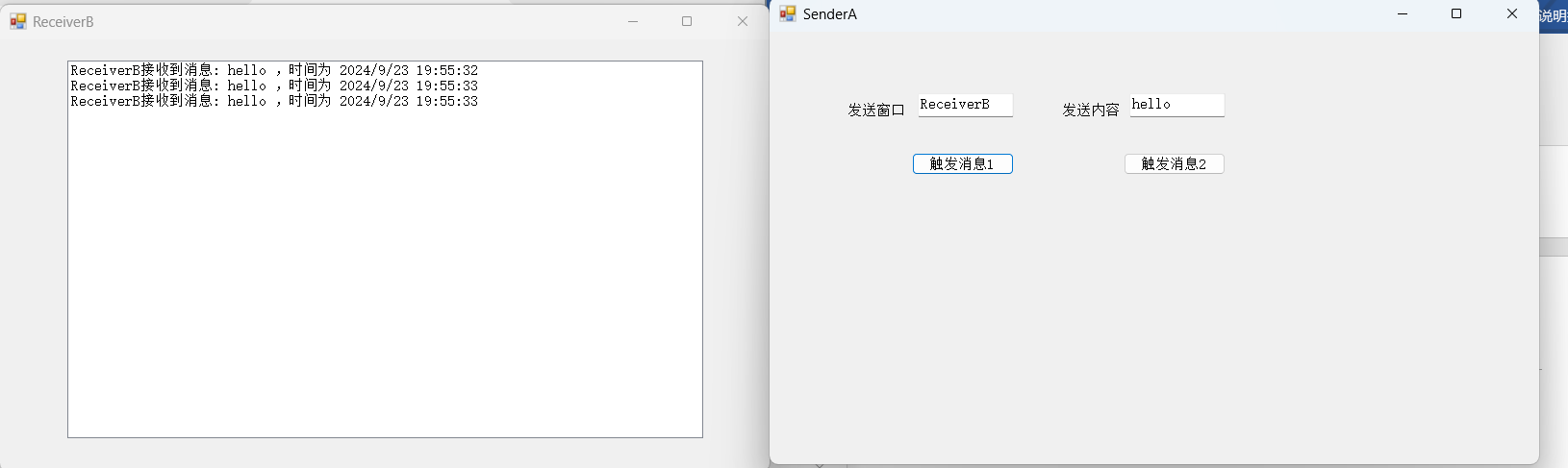
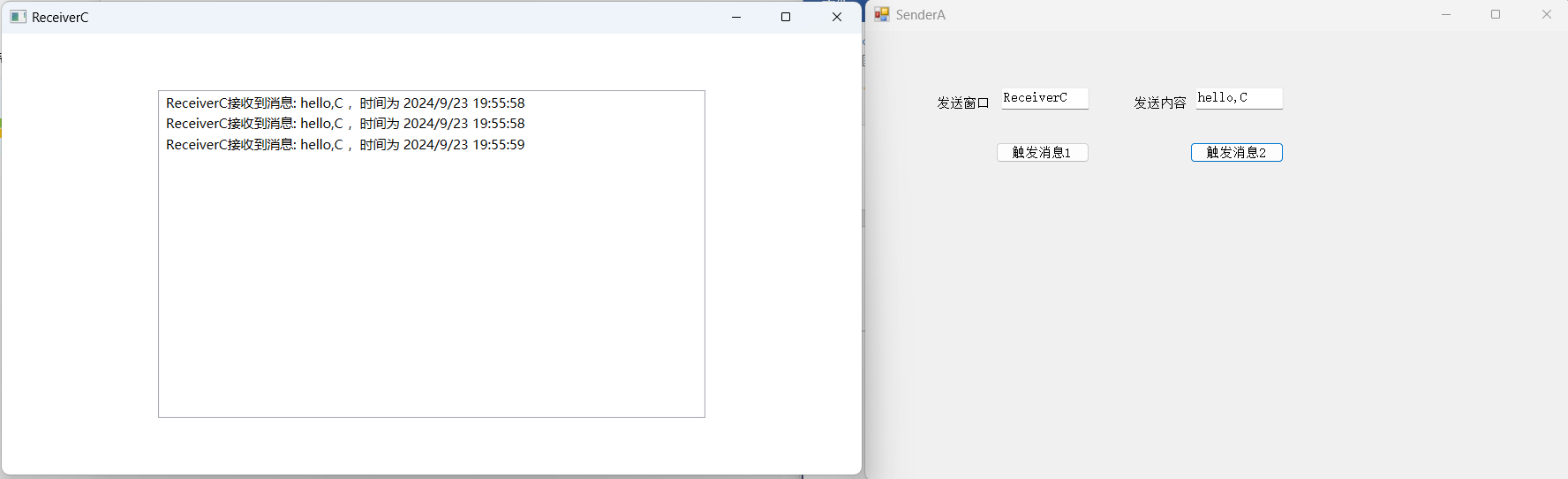
钩子函数WndProc用于处理特定的WM\_COPYDATA消息，并将接收到的消息内容显示在lstMessages列表框中。

# 3重要小操作

在这个实验过程中最重要的中间操作是在三个窗口项目文件中添加对CopyDataStruct的引用，在此描述一下具体操作。

首先编写完成CopyDataStruct.cs,之后生成.dll文件，一般保存在./bin/Debug/中，右键SenderA项目，添加->引用->浏览->选择CopyDataStruct.dll ，最后在cs文件中添加using语句即可完成对信息结构的引用。

# 4调试结果

在调试过程中SenderA项目能够成功发送消息到ReceiverB和ReceiverC项目。ReceiverB和ReceiverC项目能够正确接收消息，并在列表框中显示消息内容及接收时间。以下是调试结果的截图：

# 5实验总结

本实验通过实现基于Windows消息机制的跨进程通信，验证了COPYDATASTRUCT结构体在消息传递中的可行性和有效性。实验过程中SenderA项目成功发送消息到ReceiverB和ReceiverC项目，接收者能够正确显示接收到的消息内容及时间。

教师评语评分

评语：

评分：

评阅人：

年 月 日

（备注：对该实验报告给予优点和不足的评价，并给出百分之评分。）