**实 验 报 告 五**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **课 程** | 桌面应用程序开发 | **实验项目** | C#中的委托机制 | **成 绩** |  |
| **专业班级** | **软件工程一大班** | **实验日期** | **2019.11.2** | **批阅日期** |  |
| **姓 名** | **李毅** | **学 号** | 201731062133 | **指导教师** | 王世元 |
| **姓 名** | **李文毅** | **学 号** | 201731062208 |  |  |
| **姓 名** | **江镇岐** | **学 号** | 201731062210 |  |  |

**一【实验目的】**

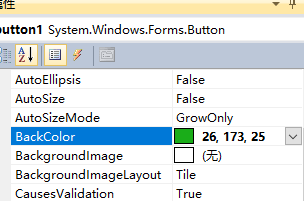
1. 掌握C#中委托的使用。掌握C#中装箱拆箱操作的使用。

**二【实验内容】**

**【实验1】仿微信PC端登录功能（请将所完成的软件截图及源代码附在题目后面）**

1. 微信APP是一款非常流行的及时通讯类应用软件，请利用所学的桌面应用开发知识，仿照完成一个类似微信登录功能的小软件，功能要求如下所示，最终效果见Demo文件夹：
2. 软件需求及编码提示：

* 该软件包含一个主窗体，窗体标题为“微信2019”，窗体的大小为280\*400。该窗体标题栏右边没有最小化，最大化，以及退出按钮。
* 窗体的右上角放置两个大小为24\*24的PictureBox控件，左侧控件放置“设置”图标，右侧放置“关闭”图标（相关图标资源在素材文件夹中）。
* 通过为上述PictureBox控件添加相关事件，完成如下效果。当鼠标进入上述控件时，设置图标背景变为LightGray颜色，关闭图标变为Red，当鼠标移出上述图标时，背景色都变为透明色，以此来友好的提醒用户相关操作。
* 在窗体合适位置添加一个大小为96\*96的PictureBox控件，用于显示用户头像（相关头像图片资源在素材文件夹中）。
* 在头像下方添加一个Label控件，用于显示微信用户昵称，文字为“微信小冰”，字体颜色为DimGray，字体为：微软雅黑，11号字体。
* 在昵称下方添加一个Button，按钮中的文字为“登录”，字体为：微软雅黑，11号字体，按钮初始背景颜色为RGB（26, 173, 25），按钮中的字体颜色为白色，通过修改相关属性去除按钮外围的边框。提示：可以通过直接在属性栏中输入具体数值的方式，设置用户自定义的颜色，如下图所示：



* 通过修改相关属性，或者编写相关事件的方式，实现鼠标移入移出“登录”按钮时的颜色变化。鼠标移入按钮后背景色变为RGB（18, 150, 17），移出后恢复为RGB（26, 173, 25）。
* 编写对应事件，当用户点击登录按钮后，登录按钮消失，在登录按钮所在位置出现“请在手机上确认登录”文本。该文本字体，颜色同微信昵称一致。**（4分）**
* 在登录按钮下方添加一个Label，文字为“切换账号”，字体为：微软雅黑，11号字体，颜色为RGB（67, 129, 199）。
* 为了友好的提示用户相关操作，请将“设置，关闭，头像，登录，以及切换账号”五个控件的光标设置为Hand。
* 请编写代码，当用户点击了关闭图标的时候，关闭窗体。
* 相关控件摆放整齐，垂直居中对齐，左右上下间距合理。
* 相关控件的命名，以及编码方式符合常见编码规范。

1. 最终实现效果如下图所示：

**代码：**

using System;

using System.Drawing;

using System.Windows.Forms;

namespace WeChatLogin

{

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

}

private void PicbSet\_MouseLeave(object sender, EventArgs e)

{

((PictureBox)sender).BackColor = Color.Transparent;

}

private void PicbClose\_MouseEnter(object sender, EventArgs e)

{

picbClose.BackColor = Color.Red;

}

private void PicbSet\_MouseEnter(object sender, EventArgs e)

{

picbSet.BackColor = Color.LightGray;

}

private void BtnLogin\_MouseEnter(object sender, EventArgs e)

{

btnLogin.BackColor = Color.FromArgb(18, 150, 17);

}

private void BtnLogin\_MouseLeave(object sender, EventArgs e)

{

btnLogin.BackColor = Color.FromArgb(26, 173, 25);

}

private void BtnLogin\_Click(object sender, EventArgs e)

{

btnLogin.Visible = false;

lblTip.Visible = true;

}

private void PicbClose\_Click(object sender, EventArgs e)

{

this.Close();

}

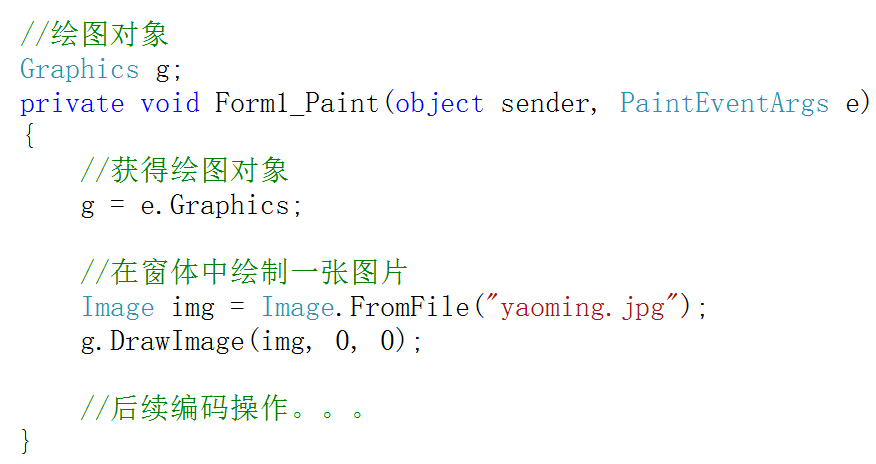
}

}

**【实验2】仿IOS 12测距仪功能（请将所完成的软件截图及源代码附在题目后面）**

1. 苹果IOS 12推出的测距仪功能，在没有尺子的情况下，能够粗略的利用手机获得距离长度，获得用户的好评。请利用所学的桌面应用开发知识，仿照完成一个类似测距仪功能的小软件，功能要求如下所示，最终效果见Demo文件夹：**（本题共30分）**
2. 软件需求及编码提示：

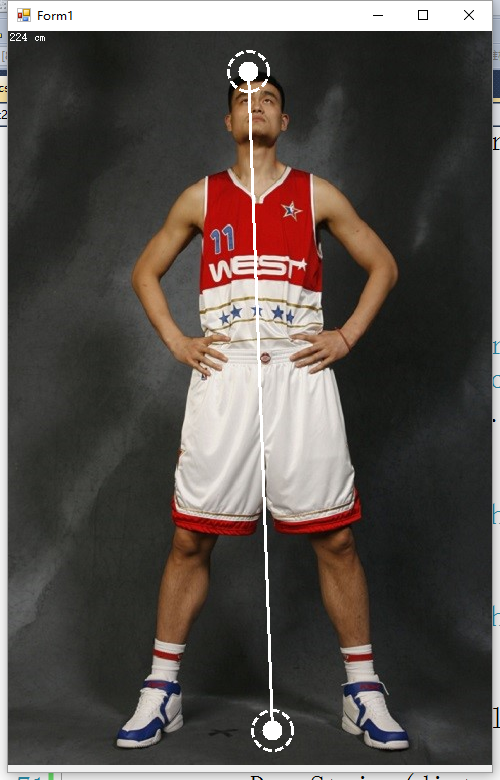
* 该软件只需要一个窗体，该窗体名称为FrmDistanceMeasure， 该窗体标题为“测距仪”， 窗体大小为500\*780。
* 首先利用Graphics对象的DrawImage( )方法，将所需测量距离的图片显示出来， 相关代码如下：测试上述编码之前，请确认yaoming.jpg文件已保存到项目所在的bin文件夹下。运行上述代码，成功后，你会看到窗体中显示了一张姚明的图片。



* 通过为该窗体添加相关事件，结合GDI+中的Graphics对象，实现测距以及距离展示功能。具体方法如下：
* 当用户鼠标按下的时候，以点击点位置为中心绘制一个填充颜色为白色，半径为10个像素的填充圆；并以点击点位置为中心绘制一个颜色为白色，虚线类型为划线，半径为20的轮廓圆。
* 当用户鼠标移动的时候，以点击点位置为中心绘制一个填充颜色为白色，半径为10个像素的填充圆；并以点击点位置为中心绘制一个颜色为白色，虚线类型为划线，半径为20的轮廓圆。并且在两个点之间绘制一条颜色为白色，宽度为3的直线。
* 定义两个Point对象，clickPoint, movePoint，用于分别记录鼠标点击下去的点的坐标，以及鼠标移动到的点的坐标。
* 定义一个double类型的distance变量，用于计算出真实的物理距离。利用两点之间的距离公式，可以计算出点击点与移动点的像素距离。上述像素距离\*0.34， 就能换算出实际的物理长度，以cm为单位。具体公式如下所示：
* 得到上述距离信息后， 在Paint事件中， 利用Graphics对象的DrawString( )方法，将上述距离信息，显示在窗体的左上角，具体代码如下：（其中 “f0”，表示保留0位小数）



1. 最终实现效果如下图所示：



**代码：**

using System;

using System.Drawing;

using System.Windows.Forms;

namespace IOS12TestDistance

{

public partial class FrmDistanceMeasure : Form

{

public FrmDistanceMeasure()

{

InitializeComponent();

}

Graphics g;

private void FrmDistanceMeasure\_Paint(object sender, PaintEventArgs e)

{

g =e.Graphics ;

Image img = Image.FromFile(@"..\..\..\素材\yaoming.jpg");

g.DrawImage(img, 0, 0);

g.DrawString(distance.ToString("f0") + "cm", this.Font, Brushes.White, 0, 0);

}

public static Point pMouseDown;

public static Point pMove;

int radius=10;

bool isDown;

double distance;

public void drawMyGraphics(Point p)

{

g = this.CreateGraphics();

Pen pen = new Pen(Color.White, 1);

pen.DashStyle = System.Drawing.Drawing2D.DashStyle.Dash;

g.FillEllipse(Brushes.White, new Rectangle(p.X - radius, p.Y - radius,

radius \* 2, radius \* 2));

g.DrawEllipse(pen, new Rectangle(p.X - radius \* 2, p.Y - radius \* 2, radius \* 4,

radius \* 4));

}

private void FrmDistanceMeasure\_MouseDown(object sender, MouseEventArgs e)

{

if (e.Button == System.Windows.Forms.MouseButtons.Left)

isDown = true;

pMouseDown = e.Location;

this.Refresh();

}

private void FrmDistanceMeasure\_MouseMove(object sender, MouseEventArgs e)

{

if (isDown)

{

this.Refresh();

drawMyGraphics(pMouseDown);

g = this.CreateGraphics();

Pen pThead = new Pen(Color.White, 3);

pMove = e.Location;

g.DrawLine(pThead, pMouseDown,pMove);

// pStart = e.Location;//将起点重置为当前鼠标所在位置

distance = Math.Sqrt(Math.Pow((pMouseDown.X - pMove.X),2 )+Math.Pow( (pMouseDown.Y - pMove.Y),2)) \* 0.34f;

// g.Clear(Color.Transparent);

}

}

private void FrmDistanceMeasure\_MouseUp(object sender, MouseEventArgs e)

{

drawMyGraphics(pMove);

isDown = false;

}

}

}

**实验报告要求**

实验过程中的心得体会，写在下面的空白中。