Tori Lentz

Dave Retterer

Design Patterns

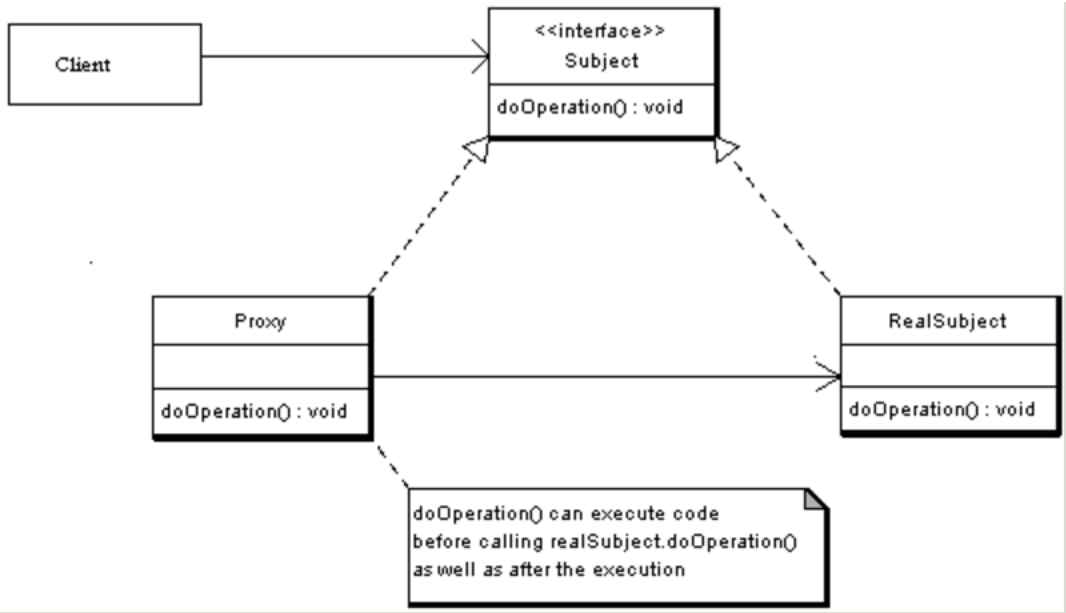
October 31, 2016

Proxy Pattern

Introduction

The purpose of this assignment is to implement the Proxy Pattern. The goal of this pattern is to be able to call upon a "proxy" that will take the place of a "real subject". By doing this, one can have a class substitute itself for another one.

UML Diagram



Subject	This class is the interface implemented by the RealSubject and Proxy classes. This class also represents the RealSubjects services.
Proxy	This class controls much of the body of this pattern. Within this class, there is a reference that allows the Proxy to have access to the RealSubject class. The same interface as the RealSubject is implemented so that the Proxy can be substituted as the RealSubject.
RealSubject	This class is the real object that the Proxy will take the place of.
http://www.oodesign.com/proxy-pattern.html	

Shown above is the UML diagram for the Proxy pattern. The Subject class is the interface for both the Proxy and RealSubject classes. Within this class there is a void doOperation method, which is in the Proxy and RealSubject classes as well. The RealSubject class is the actual object or objects of this whole

pattern. The Proxy class is what takes the place and represents the RealSubject. When first initialized it is the Proxy that the user will first see.

Code and Description

Subject

```
    public interface Subject
    {
        System.Drawing.Color
GetCol(System.Windows.Forms.ColorDialog ColDig);
        string GetA(System.Windows.Forms.ColorDialog ColDig);
        string GetR(System.Windows.Forms.ColorDialog ColDig);
        string GetG(System.Windows.Forms.ColorDialog ColDig);
        string GetB(System.Windows.Forms.ColorDialog ColDig);
    }
```

To the left is my Interface, which is also called the Subject. Within it, the operations to be performed by the RealSubject class are set up. My program uses colors so the operations I needed to initialize were to get the color through a color dialog box and then, to get the individual R, G, B, A values of that particular color.

Real Subject

```
    public class RealSubject : Subject
    {
        public System.Drawing.Color
GetCol(System.Windows.Forms.ColorDialog ColDig)
        {
            return ColDig.Color;
        }
        public string GetA(System.Windows.Forms.ColorDialog
ColDig)
        {
            return ColDig.Color.A.ToString();
        }
        public string GetR(System.Windows.Forms.ColorDialog ColDig)
        {
            return ColDig.Color.R.ToString();
        }
        public string GetG(System.Windows.Forms.ColorDialog ColDig)
        {
            return ColDig.Color.G.ToString();
        }
        public string GetB(System.Windows.Forms.ColorDialog ColDig)
        {
            return ColDig.Color.B.ToString();
        }

    }
```

Shown here, is my RealSubject class. This class contains the definitions for the methods that will be called upon by the Proxy class.

Proxy

```
    public class Proxy : Subject
    {
        private RealSubject _strftn = new RealSubject();

        public System.Drawing.Color
GetCol(System.Windows.Forms.ColorDialog ColDig)
        {
            return _strftn.GetCol(ColDig);
```

The Proxy class here calls the RealSubject class. Then, it returns the values that were specified in the operations.

```
        }
        public string GetA(System.Windows.Forms.ColorDialog ColDig)
        {
            return _strftn.GetA(ColDig);
        }
        public string GetR(System.Windows.Forms.ColorDialog ColDig)
        {
            return _strftn.GetR(ColDig);
        }
        public string GetG(System.Windows.Forms.ColorDialog ColDig)
        {
            return _strftn.GetG(ColDig);
        }
        public string GetB(System.Windows.Forms.ColorDialog ColDig)
        {
            return _strftn.GetB(ColDig);
        }

    }
```

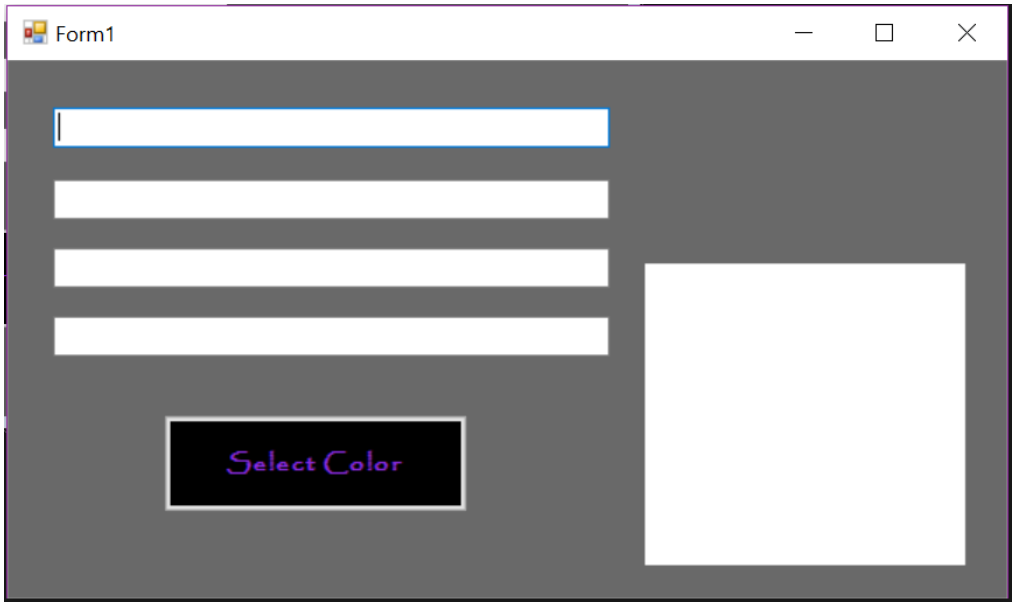
Form

Within the form I call my Proxy class and then, from there I can call upon the different operations that it can perform.

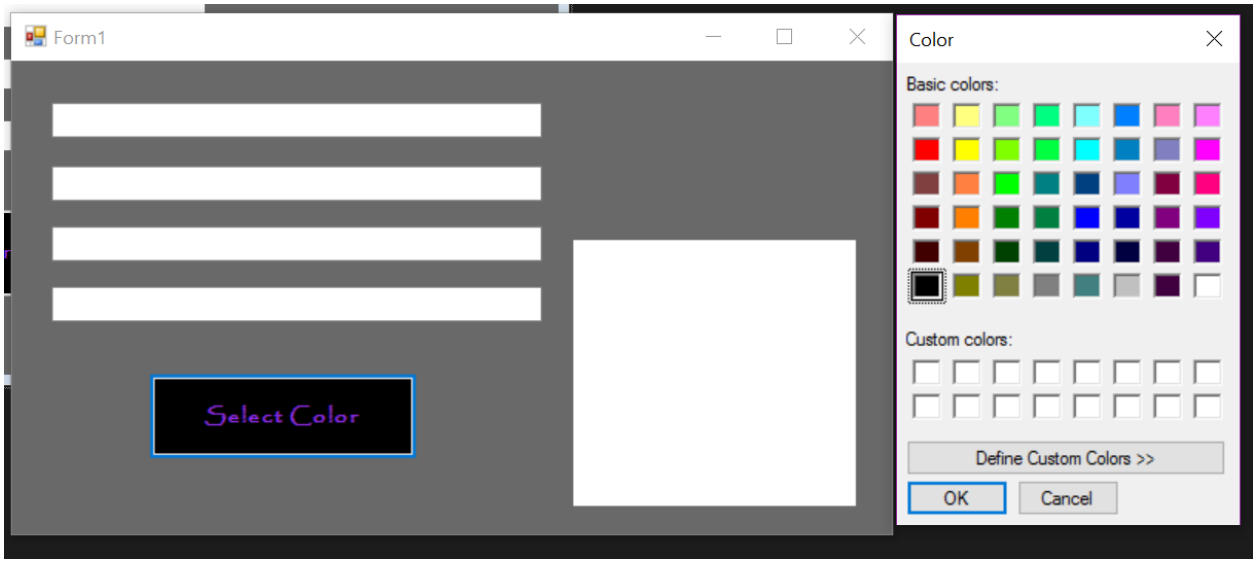
```
    public partial class Form1 : Form
    {
        Proxy proxy = new Proxy();
        ColorDialog MyDialog = new ColorDialog();

        public Form1()
        {
            InitializeComponent();
        }

        private void btn_ShowDialog_Click(object sender, EventArgs e)
        {
            if (MyDialog.ShowDialog() == DialogResult.OK)
                pnl_MyColor.BackColor = proxy.GetCol(MyDialog);
            tb_1.Text = "This is the Alpha Component of your color: " +
proxy.GetA(MyDialog);
            tb_2.Text = "This is the Red Component of your color: " +
proxy.GetR(MyDialog);
            tb_3.Text = "This is the Blue Component of your color: " +
proxy.GetG(MyDialog);
            tb_4.Text = "This is the Green Component of your color: " +
proxy.GetB(MyDialog);
        }
    }
```

Screen Shots

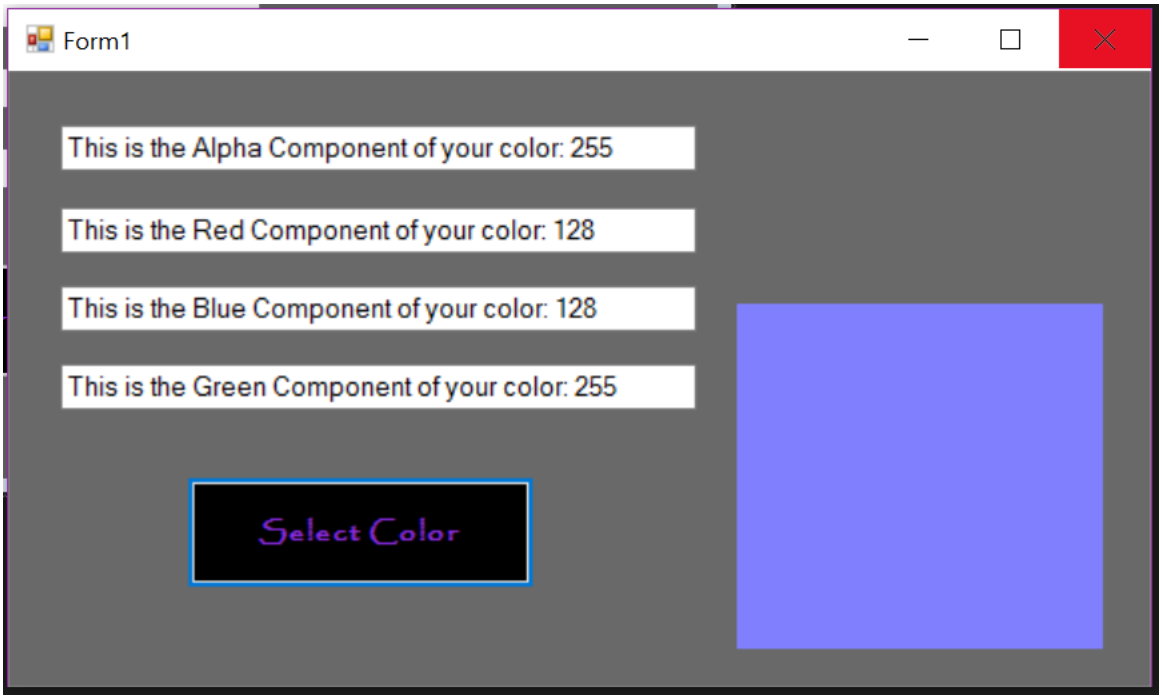
Upon opening of the form:



After clicking the button:



After selecting a color:



Observations

Overall, I thought this assignment went very well. I keep wanting to know more about the color class and I feel like this assignment really helped me to understand how I can use it more in the future. Color and color dialogs are very interesting parts of C# and there are so many things that people can do with them. I'm curious to see what I can figure out about them in the future.