Design Patterns
Prof. Retterer
Adapter Pattern

Jeremy Martin
10/17/16

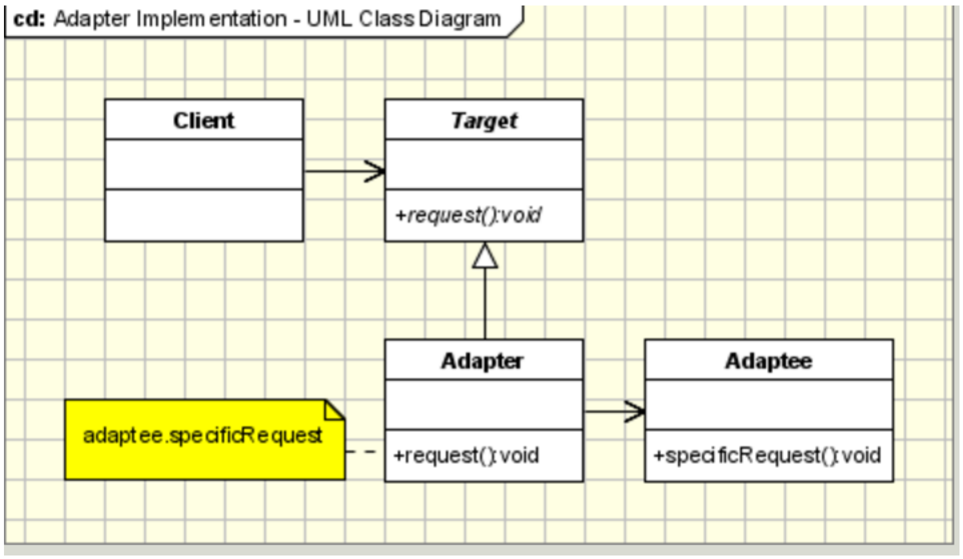
Introduction

For this project we were tasked with making a program that displayed the Adapter pattern. I chose to make a program that would show the number of US dollars that you had as another type of currency.

Adapter UML

To the right you will see the UML for the Adapter pattern.

Below I will show you the pieces of code that were used to make this program run.



The first piece is the Target class.

```
public abstract class Target
    {
        public abstract double request(double dollar);
    }
```

The next piece is the Adapter class.

```
public class Adapter : Target
    {
        Adaptee adaptee = new Adaptee();

        public override double request(double dollar)
        {
            return adaptee.specificRequest(dollar);
        }
    }
```

The next piece is the Adaptee class.

```
public class Adaptee
    {
        public double specificRequest(double dollar)
        {
            return (dollar * .91);
        }
    }
```

The last piece is the code for Form1.

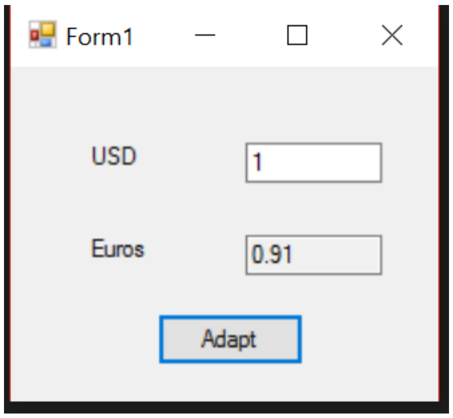
```
public partial class Form1 : Form
    {
        Adapter adapter;

        public Form1()
        {
            InitializeComponent();
            adapter = new Adapter();
        }

        private void m_btnAdapt_Click(object sender, EventArgs e)
        {
            double x = Convert.ToDouble(m_tbUSD.Text);

            m_tbEuro.Text = adapter.request(x).ToString();
        }
    }
```

This is a picture of the program working.



Conclusion

Overall this was an easy program to put together, but trying to come up with a good real life application of this pattern was the hard part. Most of my ideas all seemed to be a converter rather than an Adapter. This pattern will help me later when trying to put together code that I don't want to try to make a constructor in the main form.