

Jackson Spencer

The Proxy Pattern

November 2, 2016

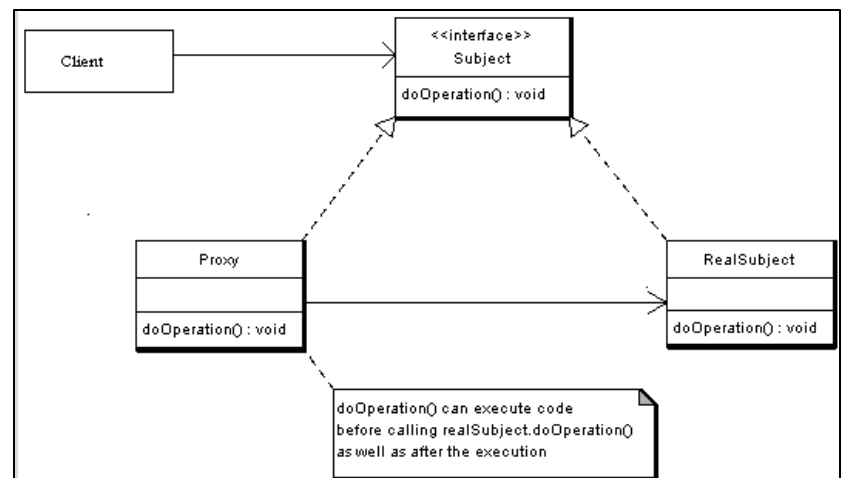
ECCS 2411: Design Patterns

Introduction

The intent of the Proxy pattern is to provide a placeholder for an object to control references to it. I am using a voting system to demonstrate this pattern. The user can create a new candidate, name how large a loss of votes can be, then add and remove votes for each created candidate. The application contains an interface that is inherited by a proxy and a real subject.

UML Diagram

The UML Diagram for the Proxy pattern is shown here. Each contains a method `doOperation()`. As mentioned before, the Proxy and the RealSubject classes inherit from the interface. The interface in my application, `Poll_Sub`,



is responsible for the methods needed in the proxy and real subject. The proxy, `Poll_Proxy`, prevents a loss of votes that is too large. It contains the controls for creation and deletion. The RealSubject, `RealPoll`, contains the methods that do the work of adding and removing votes from candidates, because it is the real object that the proxy represents.

Classes

The Poll subject interface contains the methods needed by the child classes.

```
public interface Poll_Sub      // The Subject Interface
{
    double getVotes();
    void add(double votes);
    double remove(double votes);
}
```

The RealPoll class contains the methods of keeping the vote count, how many votes are added to a candidate, and how they are removed. The getVotes() method returns the number of votes. The add() method adds to the number of votes, and the remove() method takes votes away.

```
public class RealPoll: Poll_Sub
{
    private double votes = 0;

    public double getVotes()
    {
        return votes;
    }

    public void add(double number)
    {
        votes += number;
    }

    public double remove(double number)
    {
        votes -= number;
        return number;
    }
}
```

The Proxy class inherits from the interface and stores the maximum loss of votes. It also knows which candidate has how many votes. The getVotes() returns a candidate's votes. The getName() returns the name of the candidate. The add() method adds the number of votes specified by the user. The method remove() determines if the candidate can lose the specified amount, and

displays a message depending on if the number is acceptable. Finally, the ToString() method returns the name of the candidate.

```
public class Poll_Proxy : Poll_Sub // The Proxy Class inherits from the subject
interface
{
    private Poll_Sub poll;
    private double maxVoteLoss;
    private string name;

    // Constructor
    public Poll_Proxy(double startVotes, string name)
    {
        this.maxVoteLoss = startVotes;
        this.name = name;
        this.poll = new RealPoll();
    }

    // Get the number of candidate's votes
    public double getVotes()
    {
        return poll.getVotes();
    }

    // Get the candidate's name
    public string getName()
    {
        return name;
    }

    // Add votes to candidate
    public void add(double number)
    {
        poll.add(number);
    }

    // Remove votes from candidate
    public double remove(double number)
    {
        if (number <= poll.getVotes())
        {
            if (number <= maxVoteLoss)
            {
                return poll.remove(number);
            }
            MessageBox.Show("The candidate cannot lose that many votes at once.");
            return 0;
        }
        MessageBox.Show("The candidate does not have that many votes to lose.");
        return 0;
    }

    // Override the name in the box
    public override string ToString()
    {

```

```

        return name;
    }
}

```

The Form adds functionality to the controls. It contains a list of candidates. The add button calls the add method, then displays the specified candidate's votes. The remove button does the same. The displayResult() method displays the results.

```

public partial class Form1 : Form
{
    public Form1()
    {
        InitializeComponent();

        private List<Poll_Sub> candidate = new List<Poll_Sub>();

        private void displayResult(Poll_Proxy temp)
        {
            lbl_Result.Text = temp.ToString() + " currently has " + temp.getVotes() + "
votes.";
        }

        private void btn_Candidate_Click(object sender, EventArgs e)
        {
            Poll_Proxy newCandidate = new Poll_Proxy(Double.Parse(txt_Init_Votes.Text),
txt_Candidate.Text);
            candidate.Add(newCandidate);
            cbx_Canditates.Items.Add(newCandidate);
        }

        private void btn_VoteAdd_Click(object sender, EventArgs e)
        {
            Poll_Proxy temp = (Poll_Proxy)cbx_Canditates.SelectedItem;
            temp.add(Double.Parse(txt_Votes.Text));
            displayResult(temp);
        }

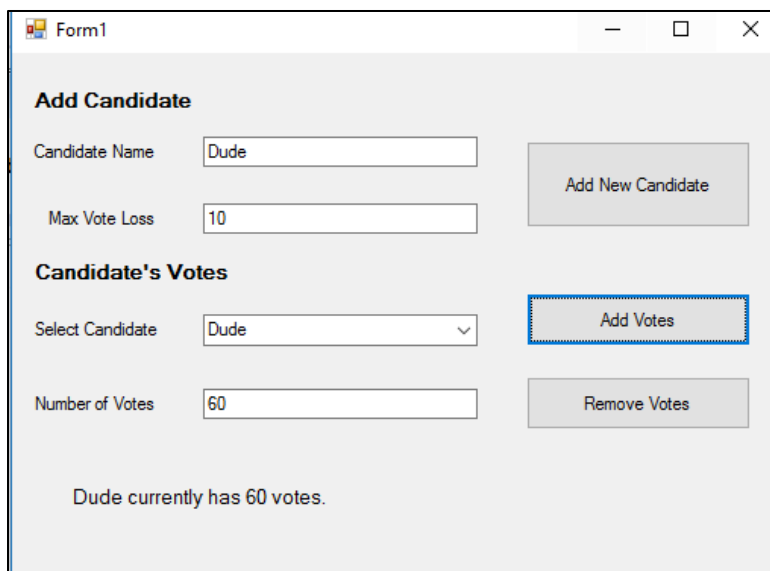
        private void btn_VoteRem_Click(object sender, EventArgs e)
        {
            Poll_Proxy temp = (Poll_Proxy)cbx_Canditates.SelectedItem;
            temp.remove(Double.Parse(txt_Votes.Text));
            displayResult(temp);
        }
}

```

Visual

In the images below, I created a candidate named Dude and added 60 votes. I then removed 10 votes from Dude. When I tried removing 12 votes, I got a warning message. Finally, I added a second candidate, Eric. Then I tested the removal from no votes. As expected, I got the error message.

1



Form1

Add Candidate

Candidate Name

Max Vote Loss

Add New Candidate

Candidate's Votes

Select Candidate

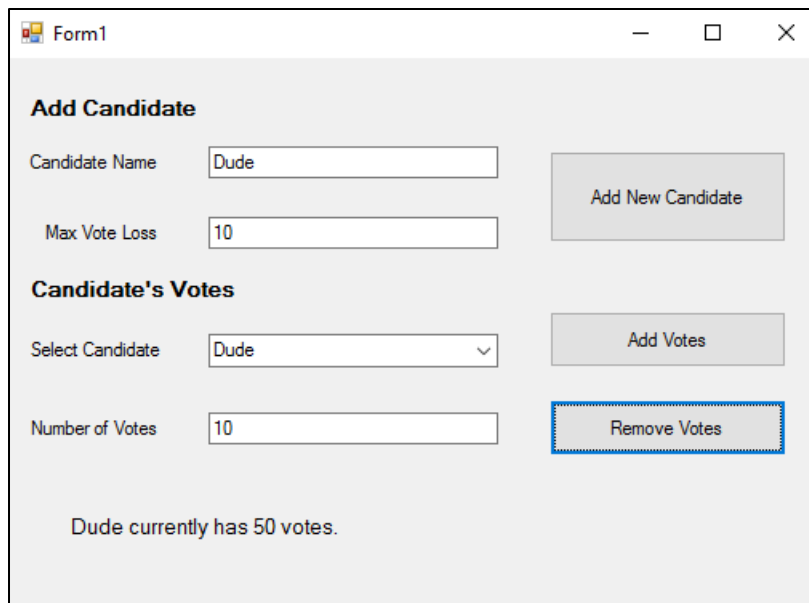
Number of Votes

Add Votes

Remove Votes

Dude currently has 60 votes.

2



Form1

Add Candidate

Candidate Name

Max Vote Loss

Add New Candidate

Candidate's Votes

Select Candidate

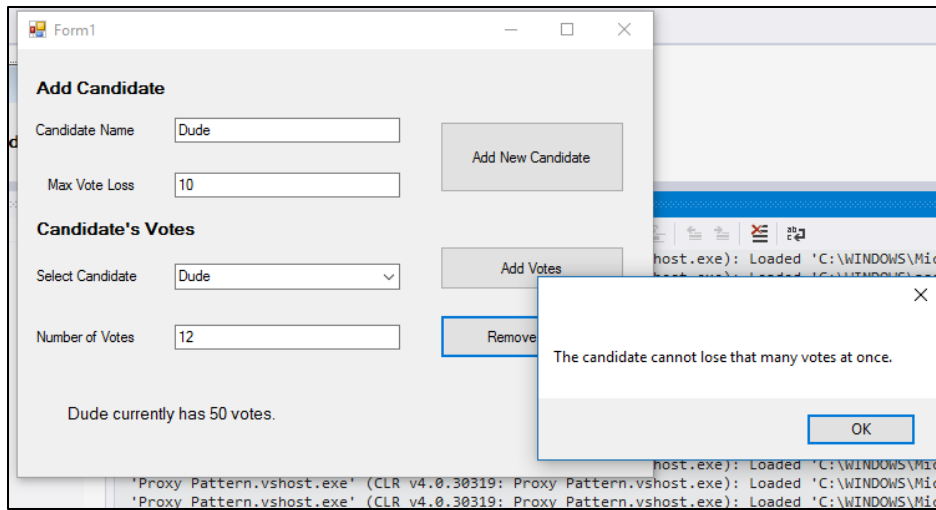
Number of Votes

Add Votes

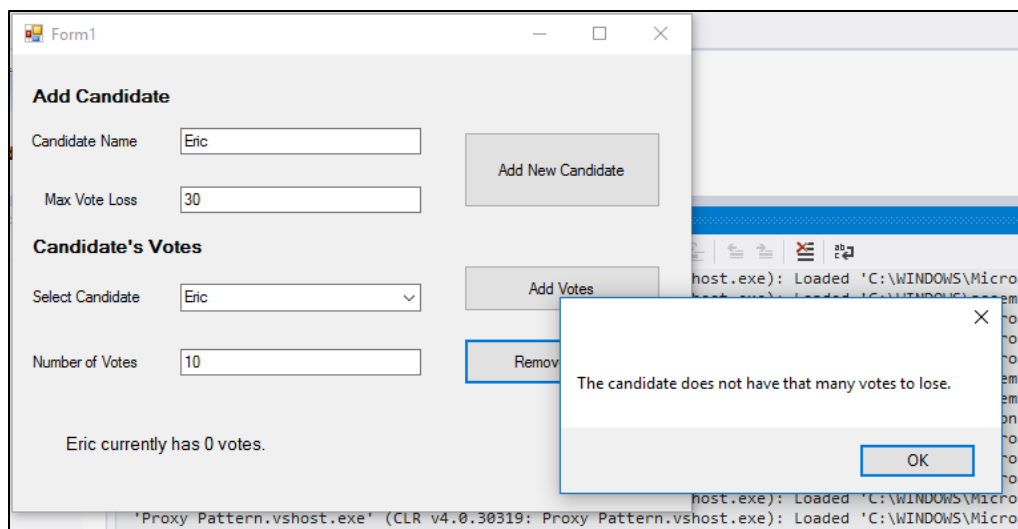
Remove Votes

Dude currently has 50 votes.

3



4



Observations

When the idea of a proxy was explained in class, it became a lot clearer on how this pattern was supposed to work. The voting example made sense, because of the examples Prof. Retterer gave in class about having someone vote in one's place. I had some confusion when I got to the Form, because when I made the methods for parsing to the textboxes, I was getting errors. This is because I used the integer type instead of the double. The system accepts integers, thus I had to

change all three classes that I created integers, and changed them to doubles. Overall, I feel as though I have gained a better understanding of this design pattern.