The Memento Design Pattern is useful when we want to save data in a temporary location and depending on the user's needs we can retrieve the old data.  
  
So, let's think about the scenario with a form ( simple Windows Forms form with a few controls) and in the form load event data will be loaded. Now the user may update the loaded data and allowed to save it. And after saving the data the user can restore it   
  
Now, the problem is, once s/he updates the form, how will we restore the old data? In this case,the Memento Design Pattern will help us.

At first we will design our original class and then we will implement the mechanism to keep a copy of the original object.  
  
The original person class is:

public class Person

{

    public String Name { get; set; }

    public String Surname { get; set; }

    MomentoPerson objMPerson = null;

    public Person()

    {

        Name = "Sourav";

        Surname = "Kayal";

        objMPerson = new MomentoPerson(Name,Surname);

    }

    public void Update(String name, string Surname)

    {

        this.Name = name;

        this.Surname = Surname;

    }

    public void Revert()

    {

        Name = objMPerson.Name;

        Surname = objMPerson.Surname;

    }

}

Here Person() is the constructor and when the object is created it will initialize the Name and surname properties.

And here is our MomentoPerson class, which is nothing but a mirror of the original class:  
  
public class MomentoPerson

{

    public String Name { get; set; }

    public string Surname { get; set; }

    public MomentoPerson(String Name, String Surname)

    {

        this.Name = Name;

        this.Surname = Surname;

    }

}

Let's now build the user interface. It's nothing but a simple form containing two TextBoxes.

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Windows.Forms;

using DesignPattern;

namespace DesignPattern

{

    public partial class Form1 : Form

    {

        public Form1()

        {

            InitializeComponent();

        }

        Person objP = new Person();

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

        {

            objP.Update(this.txtName.Text, this.txtSurname.Text);

        }

        public void DisplayCustomer()

        {

            this.txtName.Text = objP.Name;

            this.txtSurname.Text = objP.Surname;

        }

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

        {

            objP.Revert();

            DisplayCustomer();

        }

        private void Form1\_Load(object sender, EventArgs e)

        {

            DisplayCustomer();

        }

    }

}