```
//using is a directive
//System is a name space
//name space is a collection of features that our needs to run
using System;
using System.IO;
using System.Runtime.Serialization.Formatters.Binary;
using System.Diagnostics; //for notepad
//public means accessible anywhere
//partial means this class is split over multiple files
//class is a keyword and think of it as the outermost level of grouping
//:System.Web.UI.Page means our page inherits the features of a Page
[Serializable()]
public class Person  //make class serializable
{
    public string Name { get; set; } //define name property
    public decimal Salary { get; set; } //define Salary property
    public override string ToString() //override ToString() from object class
    {
        return $"{Name} makes {Salary:C} per year."; //return pretty string to describe each person
    }
}
public partial class _Default : System.Web.UI.Page
{
    protected void Button1_Click(object sender, EventArgs e)
    {
        string file = @"c:\data\person.bin"; //define path where file will be saved
        Person per = new Person() { Name = "John Smith", Salary = 78999 };//build an object
        using (FileStream str = File.Create(file)) //enclose FileStream in a using because of low level access
        {
            BinaryFormatter binFormatter = new BinaryFormatter(); //make a formatter
            binFormatter.Serialize(str, per);  //this is the step that saves the information
        }
        Process.Start("notepad.exe", file); //start notepad and display file
    }

    protected void Button2_Click(object sender, EventArgs e)
    {
        Person personRebuilt;//person object to hold the rebuilt person from the disk
        string file = @"c:\data\person.bin"; //path
        if(File.Exists(file)) //first confirm file exists
        {
            using (FileStream personStream = File.OpenRead(file)) //enclose FileStream in a using
            {
                BinaryFormatter binReader = new BinaryFormatter(); //make a formatter
                personRebuilt = (Person)binReader.Deserialize(personStream);//reconstruct person using a cast
                sampLabel.Text = personRebuilt.ToString();//invoke to string on the person
            }
        }
    }
}



```