# Data Persistence v1

We have spent a huge amount of time building applications of various form and function. However, we have only built “one-off” applications. These are applications where the data they use and/or produce die when the application closes. Being able to persist data across multiple lifetimes of the application is a vital skill. The good news is there are several ways to make this happen in .NET.

In this iteration of the application, we’ll be using what is called a flat file to store our data. In order to do this, we will leverage the power of binary serialization. So, without further ado, let’s build a contact manager application with data persistence!

## Requirements

You will create a GUI contact tracker. The GUI will have a form for displaying/editing/creating a contact AND an ItemsControl displaying a collection of contacts. The ItemsControl MUST NOT be a DataGrid. Selecting a contact in the collection displays it in the form area.

A contact consists of a name (first and last), at least one number, the type of number it is (work, home, cell), at least one email and its type (work, personal), and the "group" to which the contact belongs. You should have at least three groups (family, friends, coworkers), but may have more group possibilities if you so wish.

This will serve as our base project for the next couple of labs. While the requirements for the base project are minimal, the following should be fully functional:

1) Selecting and displaying a contact works

2) Adding a contact through the form works and immediately updates the collection in the view

3) Editing works and immediately updates the collection in the view

4) There should be some mechanism in the GUI to delete a contact

Once the base project is complete, you will add the following features:

* A File menu with Save As, Save, and Open options
  + Save As opens a file dialog box that allows the user to specify a location and file name for saving the current contact collection data.
  + Save persists the current collection of contacts to the previously designated save location. If a save location is not currently set, then Save will function the same as Save As.
  + Open displays a file dialog box to the user, allowing him/her to select the file to be opened.
    - Opening a file replaces the current collection in the application with the collection in the opened file.
* Your application will use a custom file extension for its saved files. When the file dialog opens, it will only allow files of that extension to be opened or saved.
  + This extension must be different from those considered common (txt, jpg, png, exe, dll, etc.)
* You will use binary serialization to save the application’s current contact collection.
* You will use binary deserialization to load saved data from a previously serialized file.

# Rubric

**Automatic Zero:** You hardcode ANY contacts into your collection (even a “starting set”; just don’t do it), use a DataGrid to display the collection of contacts, or fail to use binary serialization.

(15 points) GUI has a contact form with all the appropriate fields and labels

(10 points) Selecting a contact in the contact collection populates the form with that contact

(30 points) User can add/edit/delete a contact through the GUI

(10 points) All changes to the contacts collection are immediately reflected in the contact collection view portion of the GUI

(20 points) Application uses binary serialization for data persistence and retrieval

(15 points) Application has a menu with the required functions