Windows Forms Projects

1. Introduction:

So far, we have been working with Console applications to learn and test the basic syntax and structures of the C# language. It is now time to move over to cases that are more practical, and create applications having graphical user interface (GUI). From now on, all our applications, examples, exercises and assignments will be GUI-based. Using Visual Studio (VS), we are going to create a Windows Forms Application.

In our Console applications, we had classes, which took care of both calculations and communications with the user, because we were not in a position to separate these two when we had just began learning to learn the syntax of the language and basics of programming. A more object-oriented way is to let a class, a user interface class (UI) be responsible for all user interactions and let other classes to do other jobs. However, the time is mature now and as we begin creating a form object (**MainForm**) that would make our main user-interface, we will decouple all other classes from interacting with the user, and let the Form class be solely responsible for that.

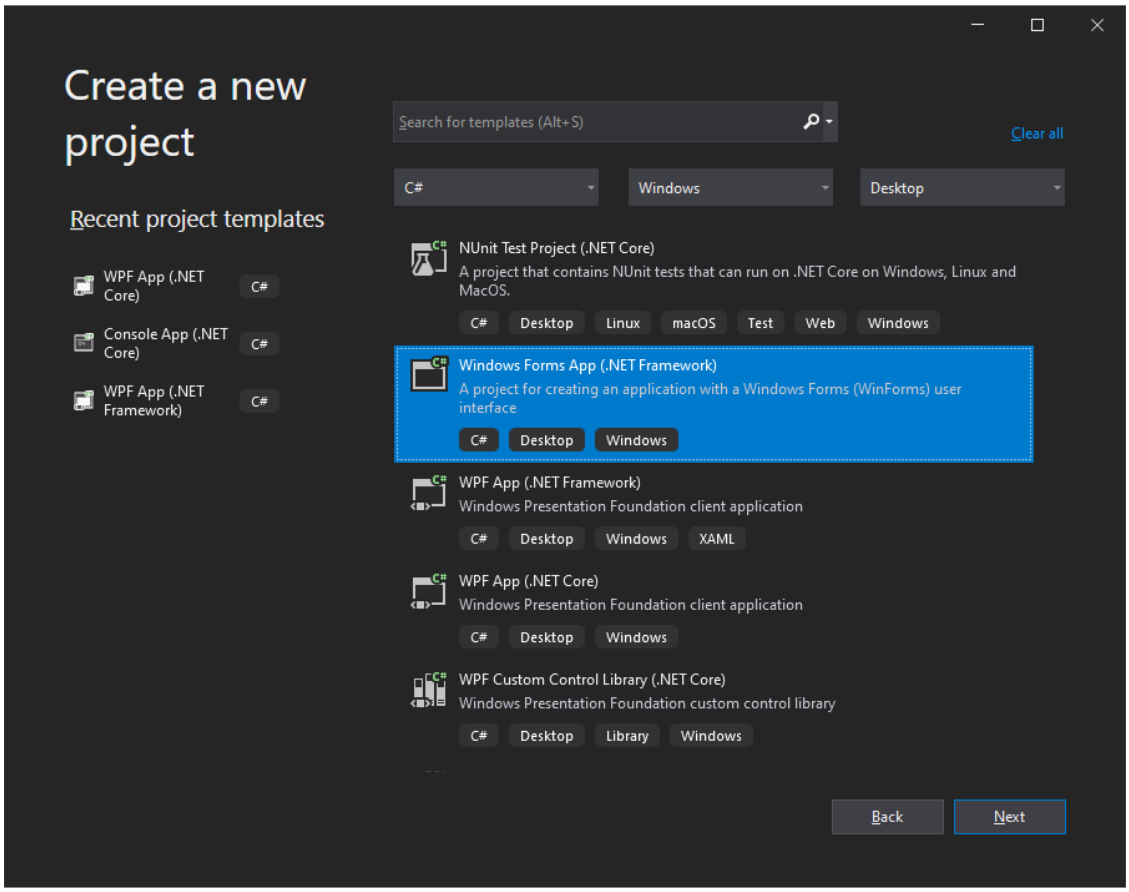
Before starting with this assignment, it is important to note the following:

a. Analogous to routines in an organization where not every staff-member is (or should be) having dialogs with customers, we let only one or more Form objects (GUI) to act as a link between the user (one who runs the program) and the rest of the program (objects of other classes). Classes other than Forms should be designed such that their methods serve the GUI. The communication between them should take place through methods. In this assignment, we will be using at least one Form (**MainForm**) class as both the starting object and the user-interface to handle all interactions with the user.

b. Remember that a class should normally not know its clients, i.e. other classes that want to use the services (methods) of that class. As an example, if you have a class **Product** with data and methods, this class should be programmed such that it should be independent of the classes that may be using it. It should able to be used by **MainForm**, as well as by another class (**ProductManager**, if you had one such class) which are in need of services that the Product class makes available. The Product class in turn can make use of other classes to accomplish its tasks.

c. In our Console applications, we had a class containing the Main method that started the program at run time. This is still true for Windows Form applications. However, working in VS, you normally do not need to do any programming in this class. VS generates this class and the programming code in this method, necessary to start the program You can begin your work directly with the start-up Form which VS creates (Form1) for you every time you create a new Windows Form Application.

d. To create a Windows Forms application, select Windows Forms .NET Framework Desktop Application (instead of Console).

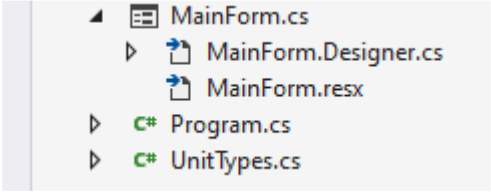


e. VS prepares the project and it creates a start-up form (Form1.cs). Change the name of this file to MainForm.cs (right-click on the file name in Project Explorer in

f. VS and select **Rename**). VS asks you if you wish to rename also the related class (Form1); accept the offer and begin working with this file. VS creates also a file named Program.cs. It is will be this file in which VS codes the Main method. Normally you should not need to change this class.

g. Every form has a visual part and a code part. You use the visual part to design your GUI, using the **Toolbox**, the **Properties** box and other graphical tools available in VS, and the related code file to program for the GUI.

h. VS creates a file xxx.Designer.cs, where xxx is the name of the related Form (MainForm.Designer.cs). VS uses this file to do its initializations and write (generate) all code behind the visual design that you do on the form.



Good Luck!

Programming is fun. Never give up. Ask for help!

Farid Naisan, Course Responsible and Instructor