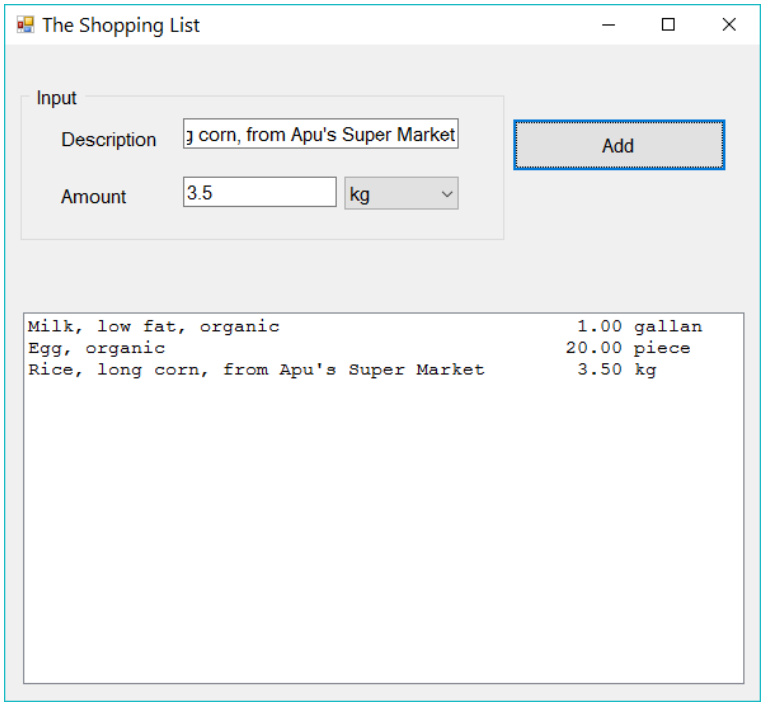
Assignment 5: Shopping List

1. Objectives

The main objectives are:

- Try some new common Windows Forms controls.
- Learn to use the List<T> collection for storing a list of objects.
- Exercise with Properties, constructors and chain-calling of constructors.



2. Description

Write a Windows Form Application that saves a list of items. Every item has the following data:

Description

Amount

Unit

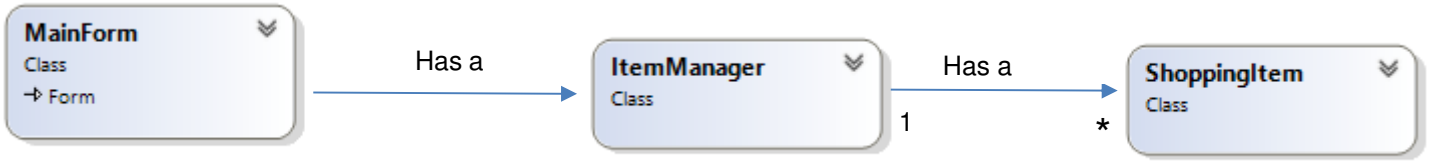
An example: “Buy milk 2 gallons”. “Buy milk” is saved as description, 2 as amount and gallons as unit

The basic requirement is that GUI should allow the user to add a new item, but as an optional (though recommended) part, the user should also be able to change or delete an existing item. The GUI should also present a list of all items saved in the registry. The list is to be updated after every change in the registry.

3. To Do

Start Visual Studio (VS), create a new project, Windows Forms Application to solve this assignment. Determine the input you may need from the user, and the output that your application should calculate and display to the user. Based on these, design your GUI using proper Windows Forms controls from the **ToolBox** in Visual Studio.

3.1 Write at least three classes, **MainForm** (GUI), **ItemManager** and **ShoppingItem**. Create also an **enum** in which you should encapsulate all common units, kg, lb, ft, etc.



The above class diagram gives a design of a solution to the given problem. **MainForm** uses an object of the **ItemMananger** class. This class is a container for a collection of **ShoppingItem** objects. The star character (*) means “many”; an object of the **ItemManager** class has one to many objects of the **ShoppingItem**. When a class contains objects of another class as its part, the association type is known as “aggregation” or a “has a” relation.

4. MainForm

4.1 VS creates a starting form and gives it the default name Form1. Rename **Form1** to **MainForm**. This class should handle all the interactions (input/output) between the user and the application, and it should not contain logics that is (or can be) a part of the **ShoppingItem** class or the **ItemManager** class.

4.2 The MainForm class should declare and create an instance of the **ItemManager** class (itemManager)

```
ItemManager itemManager = new ItemManager ( );
```

4.3 When the Add button is clicked, the **MainForm** should read input values (description amount and unit) validate the values and then save them in an object of the **ShoppingItem**. The object is then to be passed to the **ItemManager** object for saving in its list (registry of items).

4.4 When invalid values are provided, **MainForm** should message the user and not continue processing results.

5. ShoppingItem

5.1 Write a class, **ShoppingItem**, with instance variables (fields) for input data, Properties and three constructors:

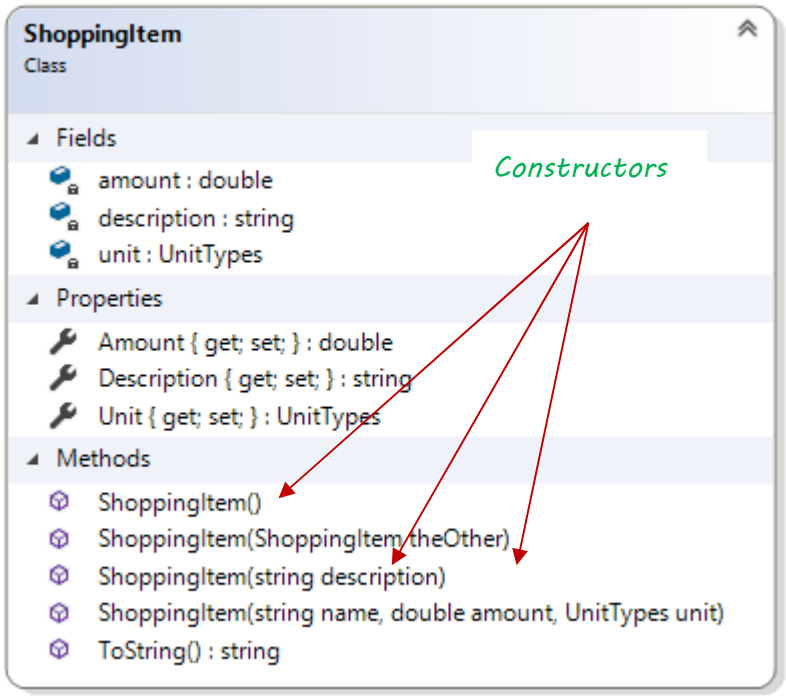
5.1.1 One default constructor.

5.1.2 One constructor with 1 parameters (for initializing description).

5.1.3 One constructor with 3 parameters (description, amount and unit).

5.2 **Chain calling**: The constructor with 1 parameter should call the one with 3 parameters (using the keyword this). This way of methods calling each other (to avoid rewriting same code) is called “chain calling”.

5.3 Override the **ToString** method so the method returns a string formatted with values saved in the **ShoppingItem** object.

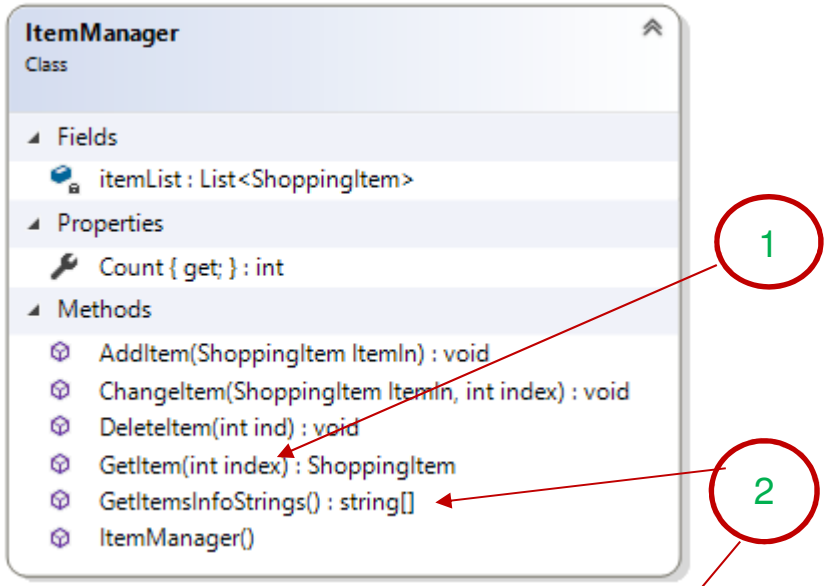


6. ItemManager

6.1 Write a class **ItemManager** and create an instance of the List<T> class, where T is to be replaced by **ShoppingItem**. . Every item that the user adds via the GUI should be first saved in this list and then the **ListBox** on the **MainForm** be updated.

6.2 The list should be private and access should not be given to the whole list, i.e. write neither a get nor a set property connected to the **List** object. Write a standard method (setter) to give access to one element at a time (1).

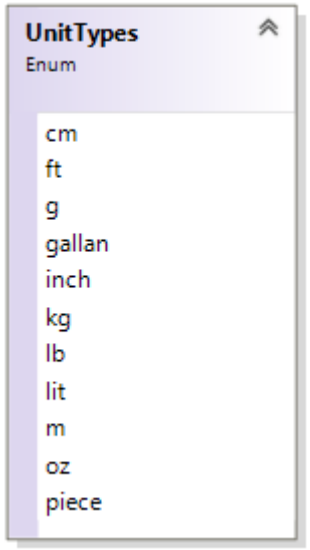
6.3 The **ItemManager** should be responsible for adding, deleting, changing and all other operations on the item-list. The **MainForm** should not do any manipulations for processing the list. It should request the manager object to provide the required services (like creating a list of info strings (2) about the items registered in the **ItemManager** object).



7. UnitTypes

7.1 Write an enum, **UnitTypes** and define the common units we use with products we buy, e.g. kg, lb, cm, in, etc. A suggested list of frequently used units is shown in this imgage. Save this enum in a separate file UnitTypes.cs.

7.2 **MainForm** uses an instance of this enum as a field.



This method returns an array of strings made up of info for every element saved in the List. For item, the method calls the ShoppingItem's ToString() method.

8. Requirements for a pass grade (G)

8.1 Declare and create an instance of the **ItemManager** and do not use any other instance variable in the **MainForm** class.

8.2 Declare and create an instance of the **List<ShoppingList>** and do not use any other instance variables in the **ItemManager** class.

8.3 The **ShoppingItem** class should not have other variables than those required to save intput values (as in class diagram given earlier). All output data should be made available through method returns.

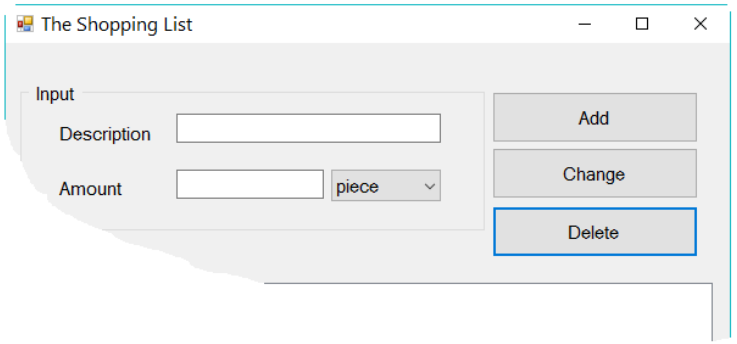
8.4 When the user clicks the **Add** button, the item should be saved in the list in the **itemManager** object that **MainForm** uses. **ManinForm** should then use the Listbox to display the list of items in the ListBox.

9. Advanced features and requirements (Optional)

If you would like to have a little challenge, the following features can be implemented.

9.1 The user should be able to change an existing item. The change should be done if the user has selected an item in the ListBox. The new values should be taken from the input controls as when adding a new item and saved in **itemManager** object.

9.2 The user should be able to delete an existing item. The delete action should be done if the user has selected an item in the ListBox..



10. Help and Guidance

A detailed guidance will be available in a separate document.

11. Submission

Submit your assignment in the same way as the previous one.

Good Luck.

Farid Naisan
Course Coordinator and teacher