week12

November 28, 2018

1 Dictionaries

- Stores a key **and** value pair
  - You cannot store one without the other
- Values are retreived by keys, e.g.:
  - Student record can be fetched by student id
  - Product price can be fetched by product name or id

2 Dictionaries Cont.

- Values can be changed, but keys cannot
- Keys can be number sequences like lists, e.g.: 1,2,3
  - But if you delete an entry, the other entries will not be shifted

3 Creating dictionaries in VB

```
'Empty Dictionary
Dim myDict As New Dictionary(Of String, Integer)

'Adding/Replacing elements are the same
myDict("Mustang") = 7000
myDict.item("F150") = 5000
myDict.Add("Focus", 3000)
myDict("Mustang") = 9000 'Replace value, why?
myDict.Add("Focus", 4000) ' this one will fail, why?
```

4 Dictionaries in VB

```
'Counting elements
myDict.count

'Accessing elements
myDict.item("Mustang") 'returns 9000
myDict("Mustang") ' also returns 9000
```

5 Dictionaries in VB

```
'Checking for element (always by key)
myDict.containsKey("mustang")  'returns False
myDict.containsKey("Mustang")  'returns True

'Delete element (always by key, associated value is removed)
myDict.remove("Mustang")
```

6 Initializing Dictionaries With Values

```
Dim myDict As New Dictionary(Of String, Integer) From {
    {"Mustang", 8000},
    {"F150", 7000},
    {"Focus", 4000}
}
```

7 Looping Over Dictionary Items

Notice how the loop variable **x** has a `x.Key` and `x.Value` properties that allow you to get to the key and value

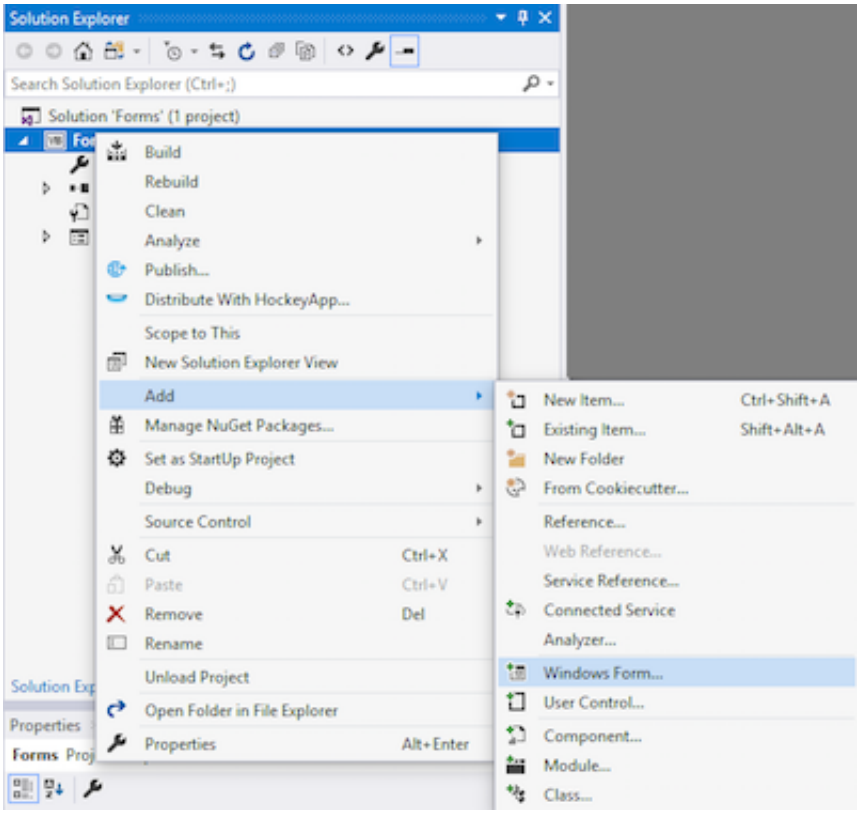
```
For Each x In myDict
    MsgBox(x.Key & ": " & x.Value)
Next
```

8 Dictionaries and Lists

- Don't forget to read the vb.net documentation and discover how you can use them
- When you need to maintain order of records, use **lists**
- When you do not need order, and you have information that identifies a record (e.g. student id), you can use **dictionaries**.
- See http://www.dotnetperls.com for useful hints

9 Shopping List Exercise

- Create a shopping list that you add items, their quantity, and unit price to the list
- Only display the items in a list, the quantity and unit price should be stored in separate dictionaries in memmory
  - Hint: use shopping item name as key
- When you select an item, display the quantity and the unit price for that item in the same textboxes used to enter the data
- User should be able to remove items from the list
- Show the total price for the shopping list, total number of items, and average item price
- Allow user to update the values for selected entries.
- Allow user to sort the list



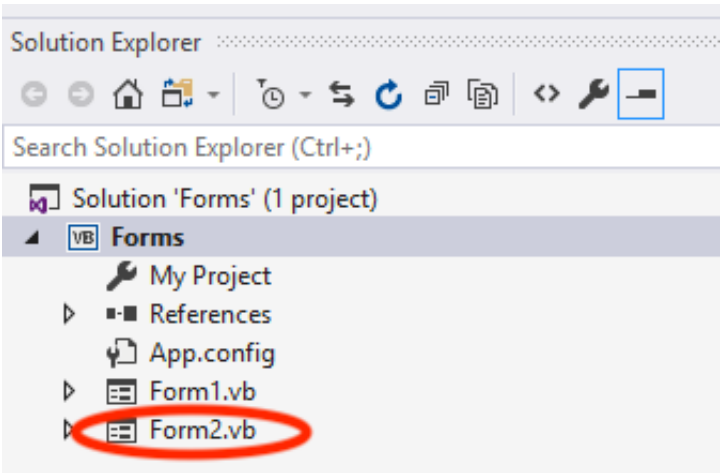
10 More advanced shopping list

- Create an application with two different types of users:
  - Sales manager
  - Shopper
- Sales manager can add items and unit prices
- Sales manager can update/delete items from list of products
- Shopper can select from available items to add to his/her shopping list
- Shopper can select quantity of each item
- Shopper can view items, the quntity, their unit prices, and the item total prices each as a list
- Shopper can view total cost
- Shopper can remove items from shopping list

11 Adding a new form to your project

12 Designing The Form

Select the form and design it as you did with previous forms, Notice its name is **Form2**



13 To show Form2

- Use the command `Form2.Show()` inside Form1
  - Add it to a button subroutine
  - To open Form2 from Form1
- Remember, you can rename the forms, they don't have to be Form1 and Form2
  - Right click on the form you want to rename in the solution explorer

14 To Hide Form2

- Use the command `Me.Close()` inside Form2 itself
  - Add it to a button subroutine