## Exercise 3 – Data Binding

* What you'll learn
  + How to use data binding to tie UI controls to properties in a ViewModel
* Stuff to Know
  + Your ViewModel should inherit from ViewModelBase, in the skeleton project
  + Use the Binding keyword to set up the binding
  + The default UpdateSourceTrigger for a TextBox is LostFocus
* Best Practices
  + Property setter for bindable properties in ViewModel must invoke PropertyChanged event

### The Exercise

* Start from existing project: Exercise\_3\_DataBinding
  + The project includes a ViewModelBase class that you can use
* Create a ViewModel
  + Create a Person ViewModel, with properties for first and last name
  + Your ViewModel should inherit from ViewModelBase, to leverage the SetValue method
* Set DataContext of main window
  + In the code-behind for MainWindow, after initializing components, create an instance of your ViewModel and set the window's data context to the instance
  + When creating the instance, give it default values for the first and last name
* Create controls in a grid
  + Create a 4 row, 3 column Grid. You can set the Height of the first three rows to Auto
  + Use column widths of 1\*, 2\*, 5\*
  + Add first and last name labels in the first column
  + Add first and last name textboxes in the second column
  + Use data binding to wire up the textboxes to the first and last name
  + Run / test. Use trace output to verify that ViewModel properties are being updated
* Add other data bound labels
  + Add a horizontal StackPanel to the 3rd row
  + Add two labels in this StackPanel and bind them to first and last name
  + Run / test. Demonstrate--when you enter new values, the labels should update
* Extra credit
  + When does binding occur? Can you make a change so that it updates as you type?