A Design Pattern for Creating Master/Detail Screens

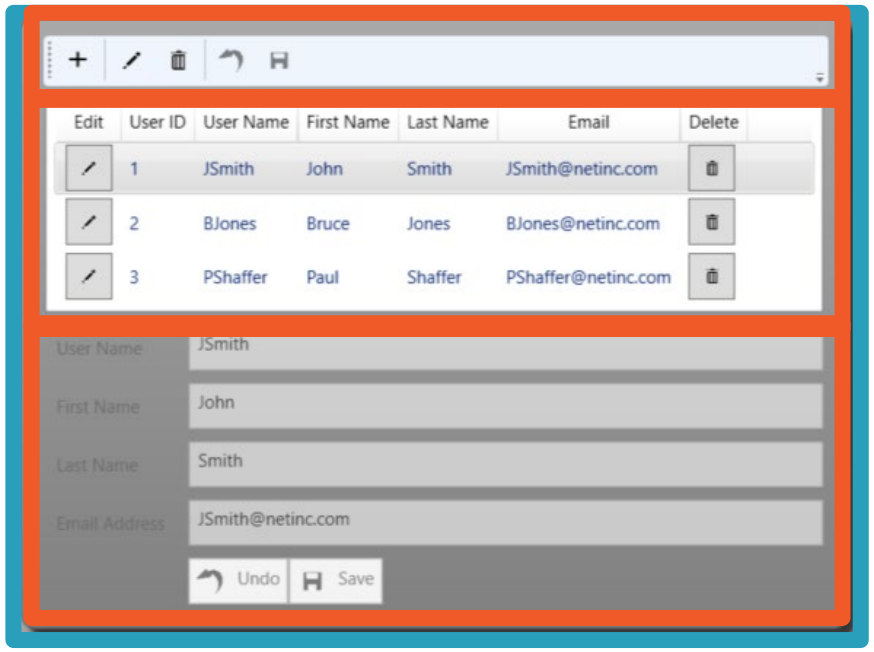
Paul D. Sheriff

BUSINESS SOLUTIONS ARCHITECT, FAIRWAY TECHNOLOGIES, INC.

www.fairwaytech.com psheriff@fairwaytech.com

Module Goals

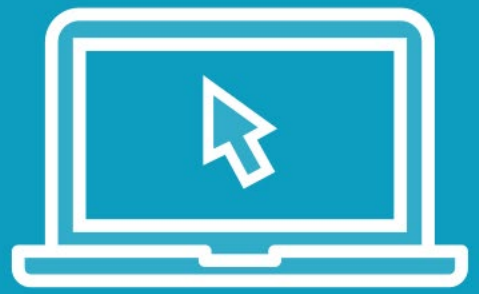
- Create master/detail screen
- Use control aggregation
- Use view model inheritance
- Handle screen state
- Add, edit, delete users



Aggregate three Controls

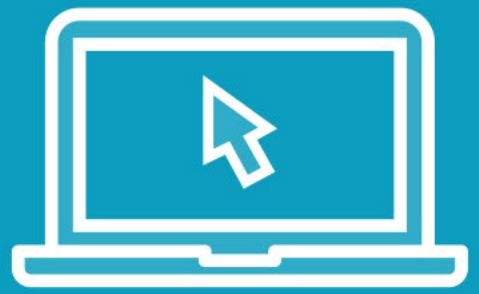
- User maintenance control
  - Contains the toolbar
  - Contains two other user controls
    - User list control
    - User detail control

Demo



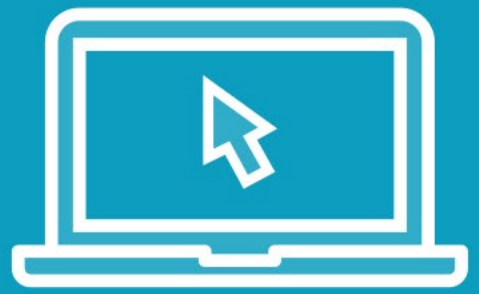
Create user control to list users

Demo



Create user control list view model

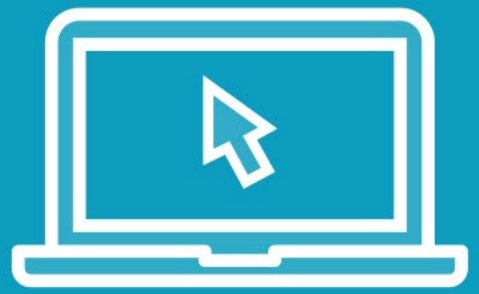
Demo



Create detail user control

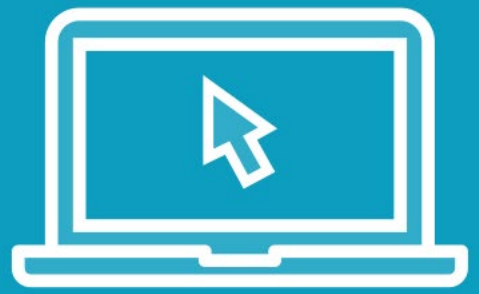
Create detail view model

Demo



Aggregate list and detail controls

Demo



Create toolbar to add, edit, delete, undo

Manage Screen "State"

Bind each controls' IsEnabled property

Change based on "state" of screen

Control State	Normal	Add/Edit
ListView	Enabled	Disabled
Detail User Control	Disabled	Enabled
Add Button	Enabled	Disabled
Edit Button	Enabled	Disabled
Delete Button	Enabled	Disabled
Undo Button	Disabled	Enabled
Save Button	Disabled	Enabled

```
public class ViewModelAddEditDeleteBase : ViewModelBase
{
  #region Private Variables
  private bool _IsListEnabled = true;
  private bool _IsDetailEnabled = false;
  private bool _IsAddMode = false;
  #endregion

  Public Properties

  BeginEdit Method

  CancelEdit Method

  Save Method

  Delete Method
}
```

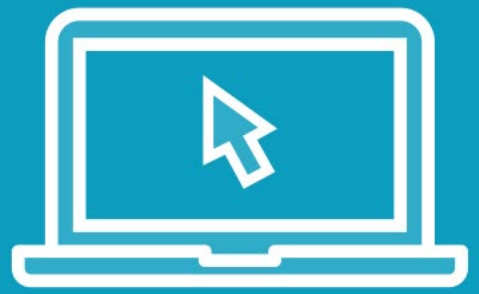
Three properties in the ViewModelAddEditDeleteBase class

- IsListEnabled
- IsDetailEnabled
- IsAddMode

Two methods for state management

- BeginEdit
- CancelEdit

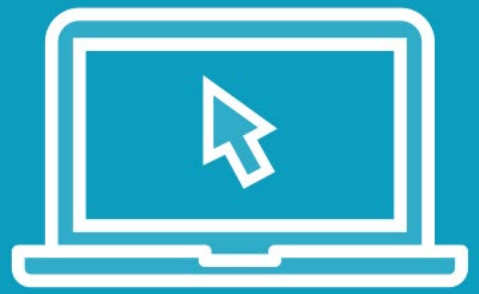
Demo



Modify UserMaintenanceListViewModel

- Inherit from ViewModelAddEditDeleteBase class
- Override Save() and Delete()

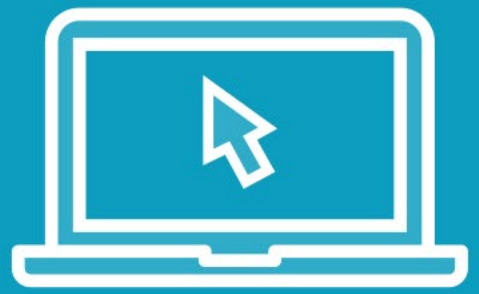
Demo



Bind state properties

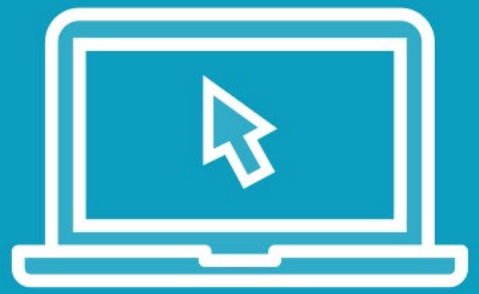
- On detail control
- On list control
- On toolbar

Demo



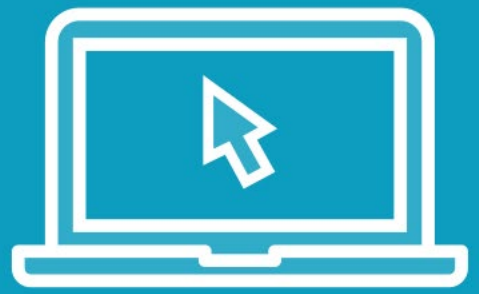
Add click events on detail control

Demo



Add click events on list control

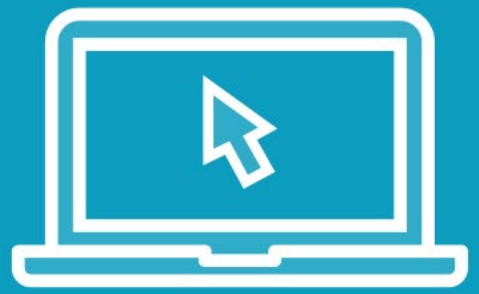
Demo



Add click events to toolbar

Write Code to Add, Edit and Delete

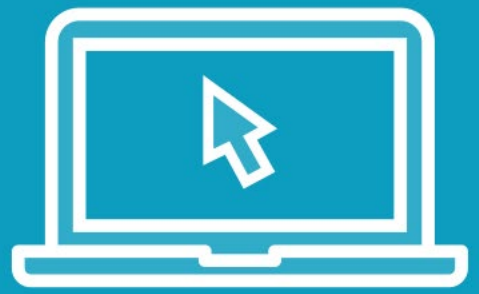
Demo



Begin and cancel editing

- Add property for original user data
- Override BeginEdit()
- Override CancelEdit()

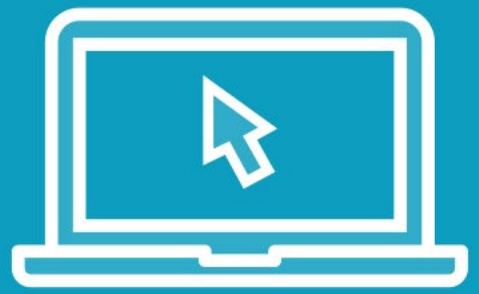
Demo



Add and update user

- Override Save()

Demo



Delete a user

- Override Delete()

Summary

State management is important

- Create a state table for your screen

Inherit from ViewModelAddEditDeleteBase class

- For list and edit modes
- Add any additional properties

Bind IsEnabled property

Write code to change state

Override methods to save/delete your data

Course Summary

- Always start with a solid architecture
- Create area to display messages
- Use the Dispatcher class for background processes
- Build user controls for each screen or partial screen
- Use control aggregation
- Use view model inheritance
- Manage the state for each screen using data binding

I hope you enjoyed this course!

Paul D. Sheriff

Business Solutions Architect, Fairway Technologies, Inc.

www.fairwaytech.com **psheriff@fairwaytech.com**