# Managing Data Presentation with Model-View-ViewModel

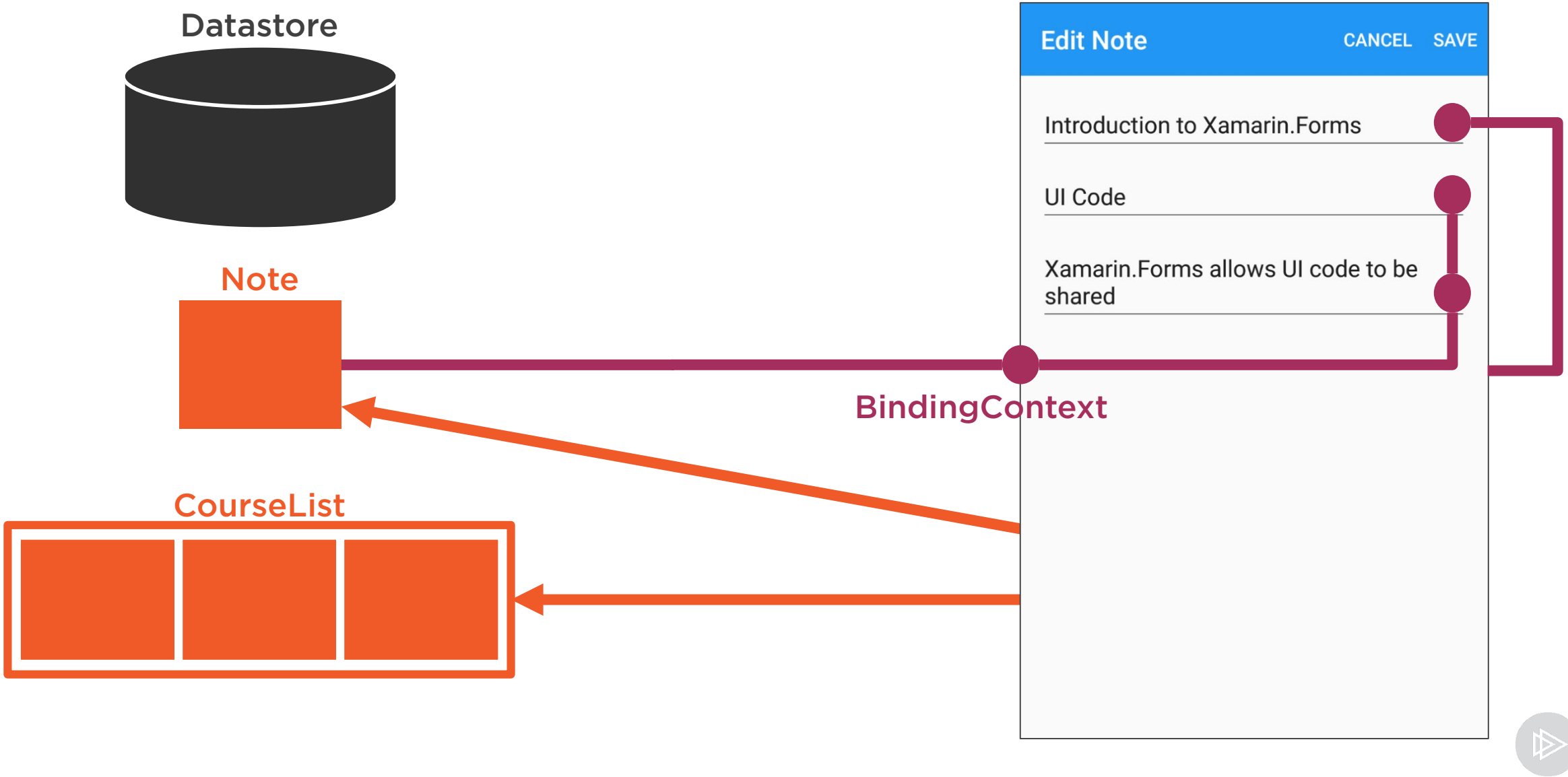
**Jim Wilson**
MOBILE SOLUTIONS DEVELOPER & ARCHITECT

@hedgehogjim  blog.jwhh.com

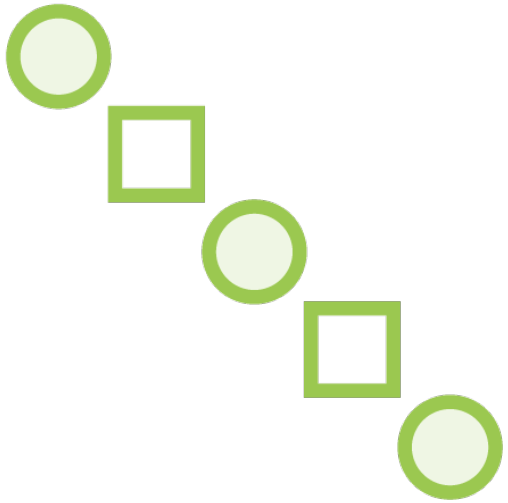
# What to Expect from This Module

- Challenges in our current implementation
- Understanding Model-View-ViewModel
- Migrating to Model-View-ViewModel
- BaseViewModel class
- Property change notifications

# Our App's Current Implementation



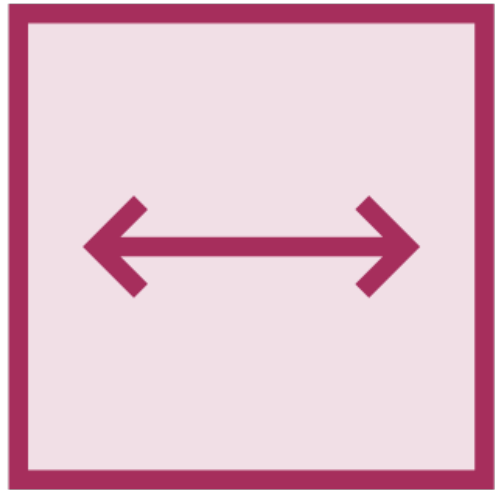
# Model-View-ViewModel (MVVM)



## Architectural pattern

## Decouples data from presentation

Presentation and data unaware of the other's details

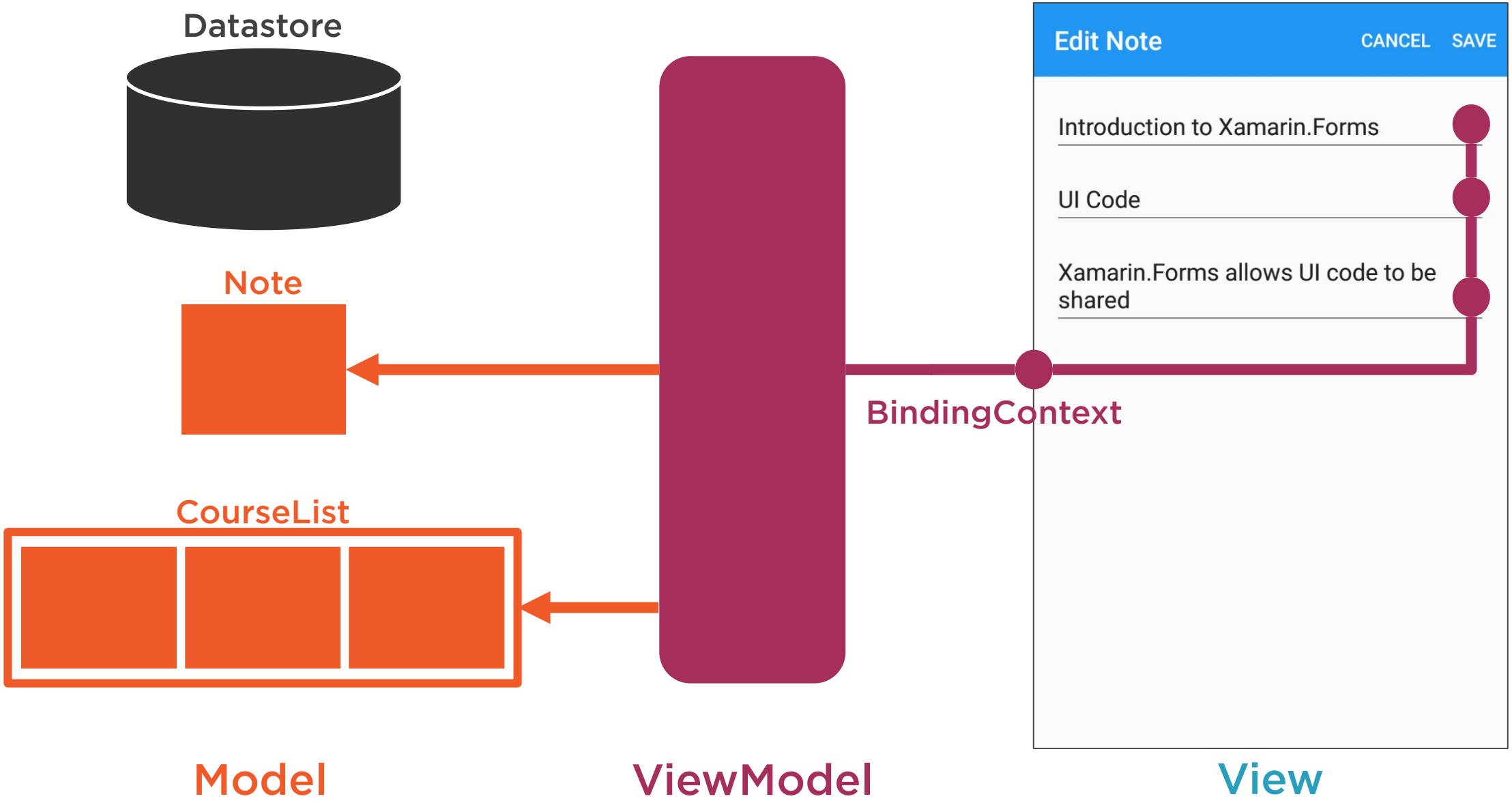


## Adds an intermediary

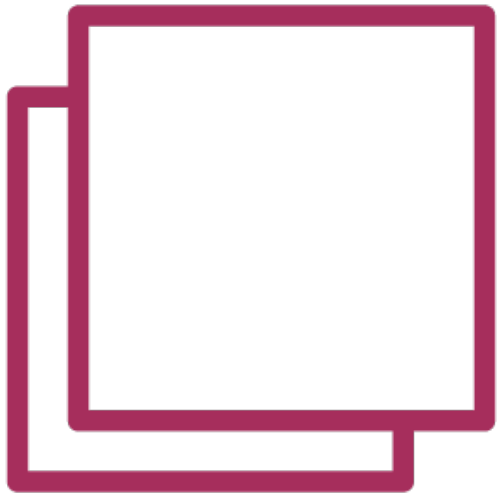
Presentation-friendly layer over data

Known as the ViewModel

# Using Model-View-ViewModel

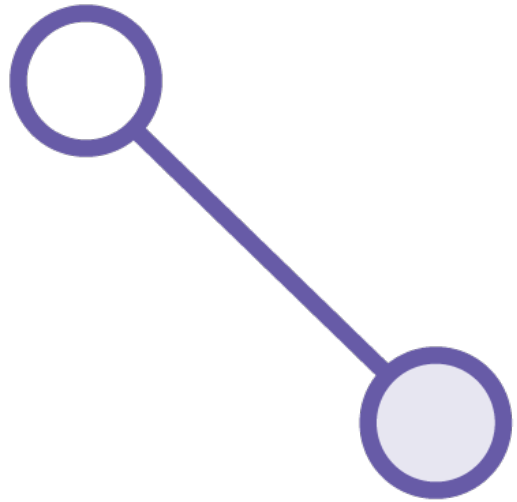


# MVVM with Xamarin.Forms



## ViewModel classes

Provide presentation-friendly layer over data



## BaseViewModel class

Serves as base to ViewModel classes

Includes code to simplify common MVVM tasks

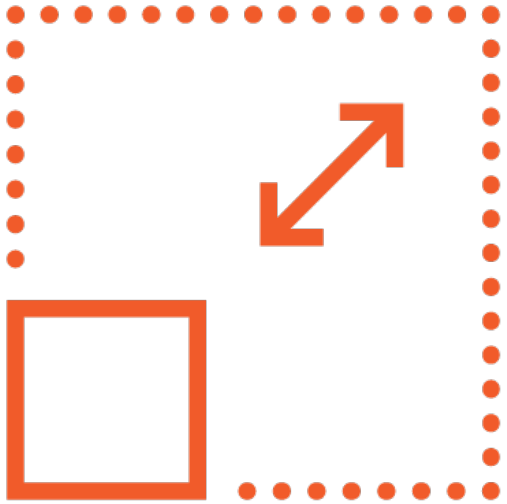
## Xamarin.Forms services

Framework features that handle common management and interaction tasks

# Providing Complete Data Binding Support

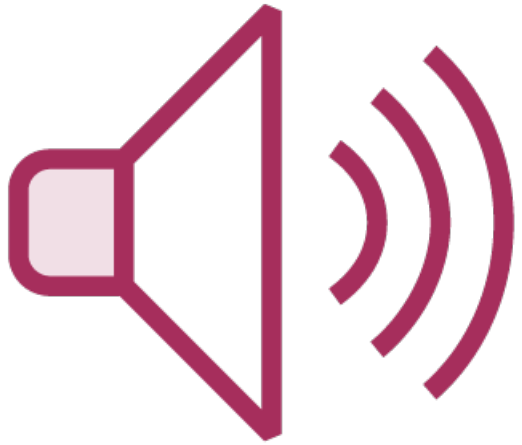
## Initial display

Target reads value from source



## Value changes

Target does not detect a change in the source



## Change notification

Source must notify target of change

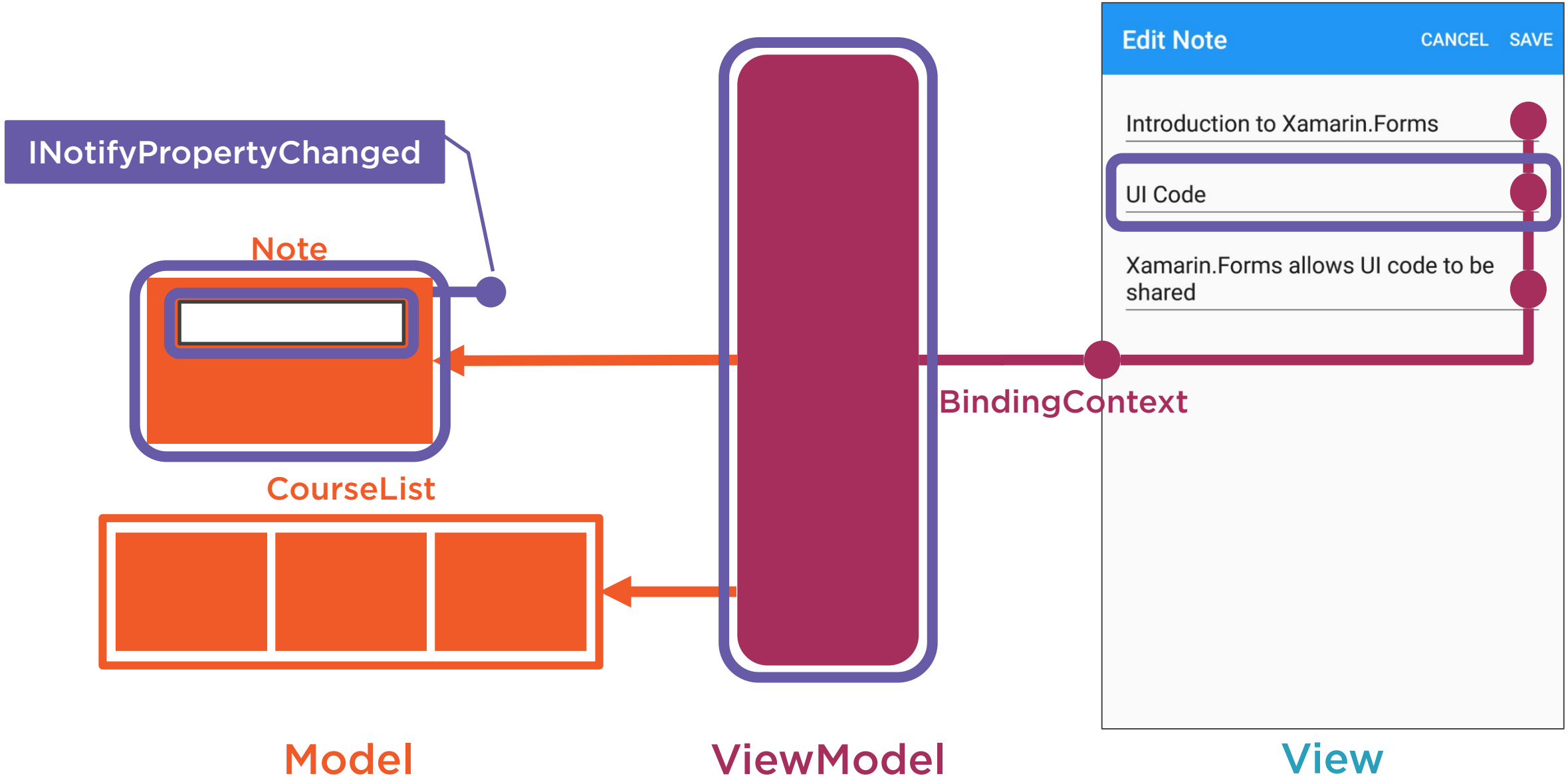
Target can then reread value

# Change Notification

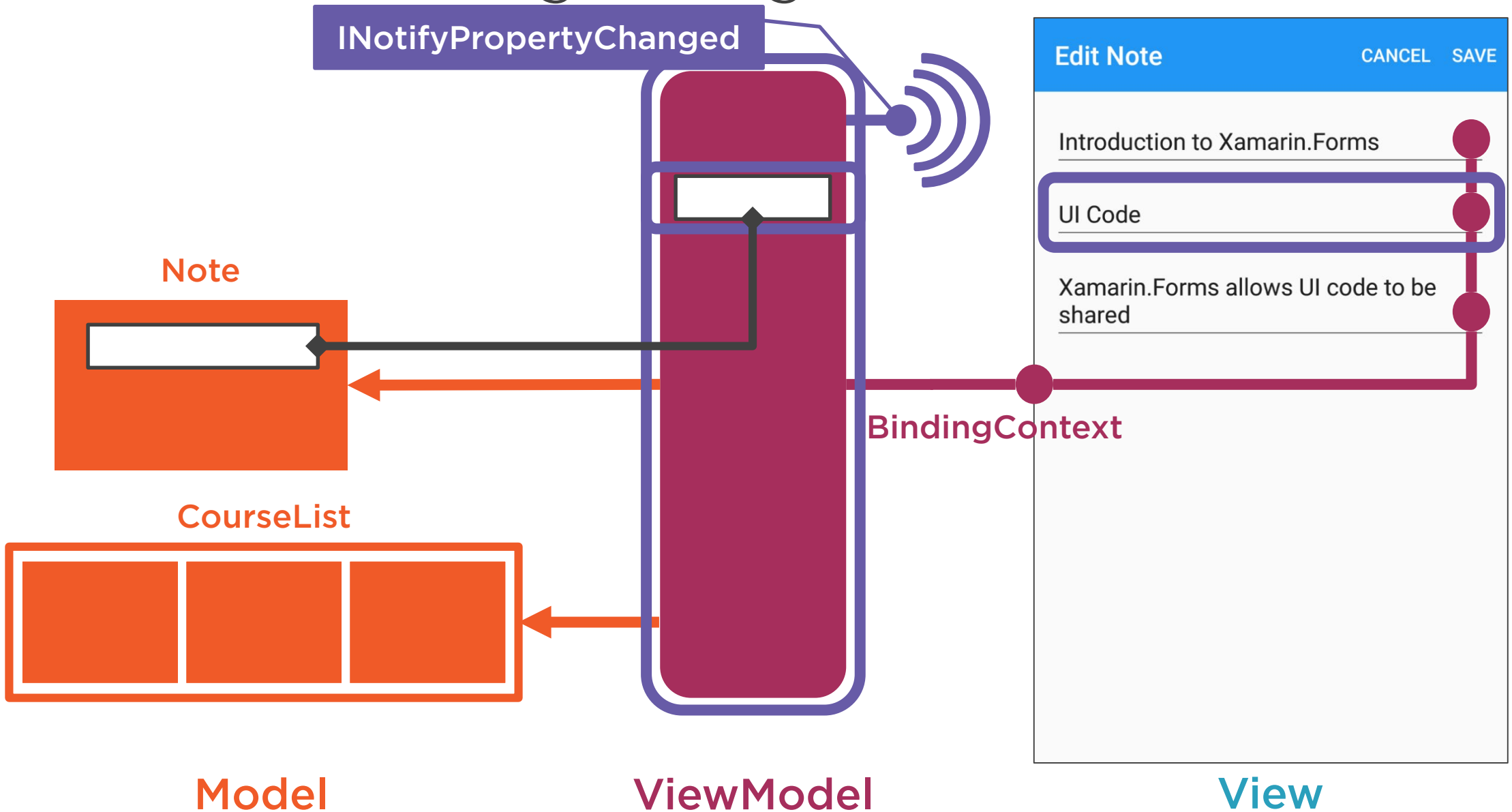
**Providing value change notification**

- Implement INotifyPropertyChanged
- Fire PropertyChanged event when value changed

# Providing Change Notification



# Providing Change Notification



# Summary

**Model-View-ViewModel**

- Architectural pattern
- Decouples data from presentation

# Summary

**View**

- User presentation

**Model**

- Data representation

**ViewModel**

- Intermediary between view and model
- Provides presentation-friendly layer over the data representation

# Summary

**ViewModel properties**

- Provide data binding support
- Provide value change notifications

**BaseViewModel class**

- Generated by Visual Studio
- Base class of ViewModel classes
- Implements INotifyPropertyChanged
- Includes code to simplify change notifications