Communicating between Views and ViewModels in WPF

Brian Noyes
CTO AND CO-FOUNDER, SOLLIANCE INC

@briannoyes www.briannoyes.com

Introduction

Commands

Attached properties and behaviors

PropertyChanged notifications

Commands

Based on classic command design pattern

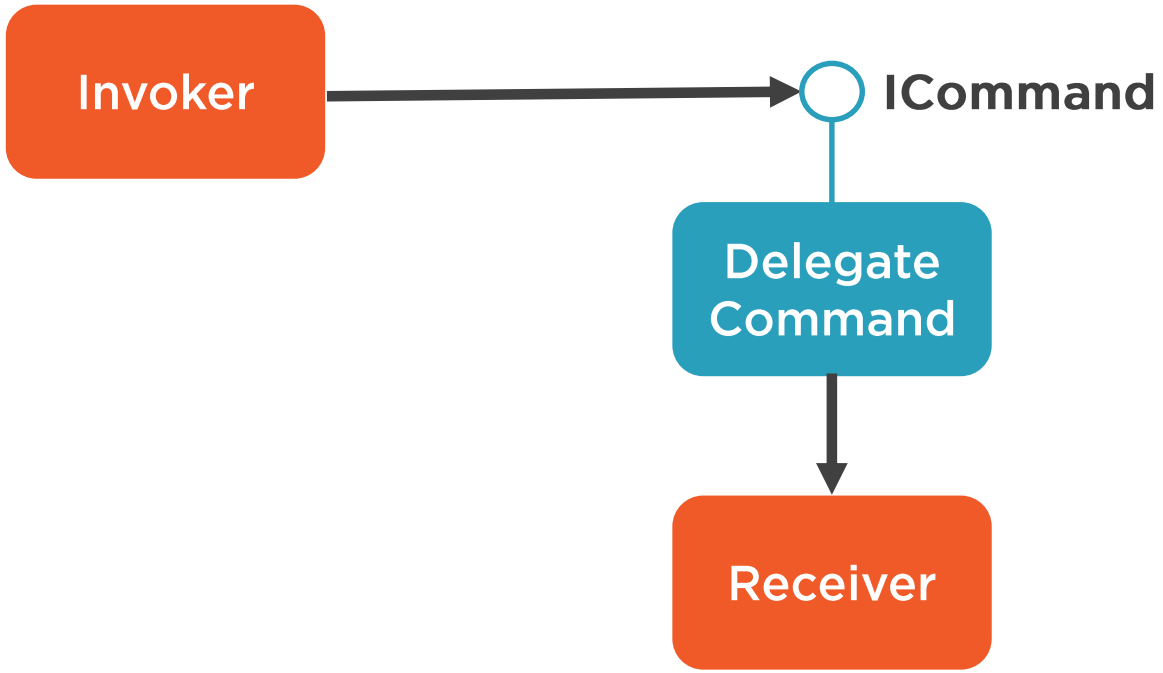
Invoker: View control

Receiver: ViewModel

Use delegating command implementation

Supports decoupled command handling invocation

Supports enabling/disabling associated control



Attached Properties/Behaviors

Attached Properties are DependencyProperties and a fundamental part of XAML

Attached Properties can be defined on any class, and can be applied to any object that derives from DependencyObject

Attached properties form the basis for behaviors

Can implement behaviors as Attached Properties, or using the XAML Behaviors for WPF open source library

Using XAML Behaviors for WPF is the recommended approach

Using XAML Behaviors for WPF

Behaviors still get hooked up to controls through Attached Properties

Behaviors can expose their own DependencyProperties, so they can be data bound in XAML

Behaviors can subscribe to events or property change notifications on the control they are attached to

Behaviors can dispatch calls into the ViewModel, through Commands, setting data bound properties, or directly invoking a method on the ViewModel

Behaviors can trigger on changes in the ViewModel to modify the control they are attached to, or other controls in the same visual tree

Property Changed Notifications

Need to trigger bindings to refresh as property values change

Two options: Dependency Properties or INotifyPropertyChanged

INotifyPropertyChanged is more appropriate for MVVM

Summary

Commands are the primary form of View to ViewModel communication

Behaviors bridge the gap between View events and ViewModel logic

PropertyChanged notifications keep your Views in sync with the properties of your Models or ViewModels