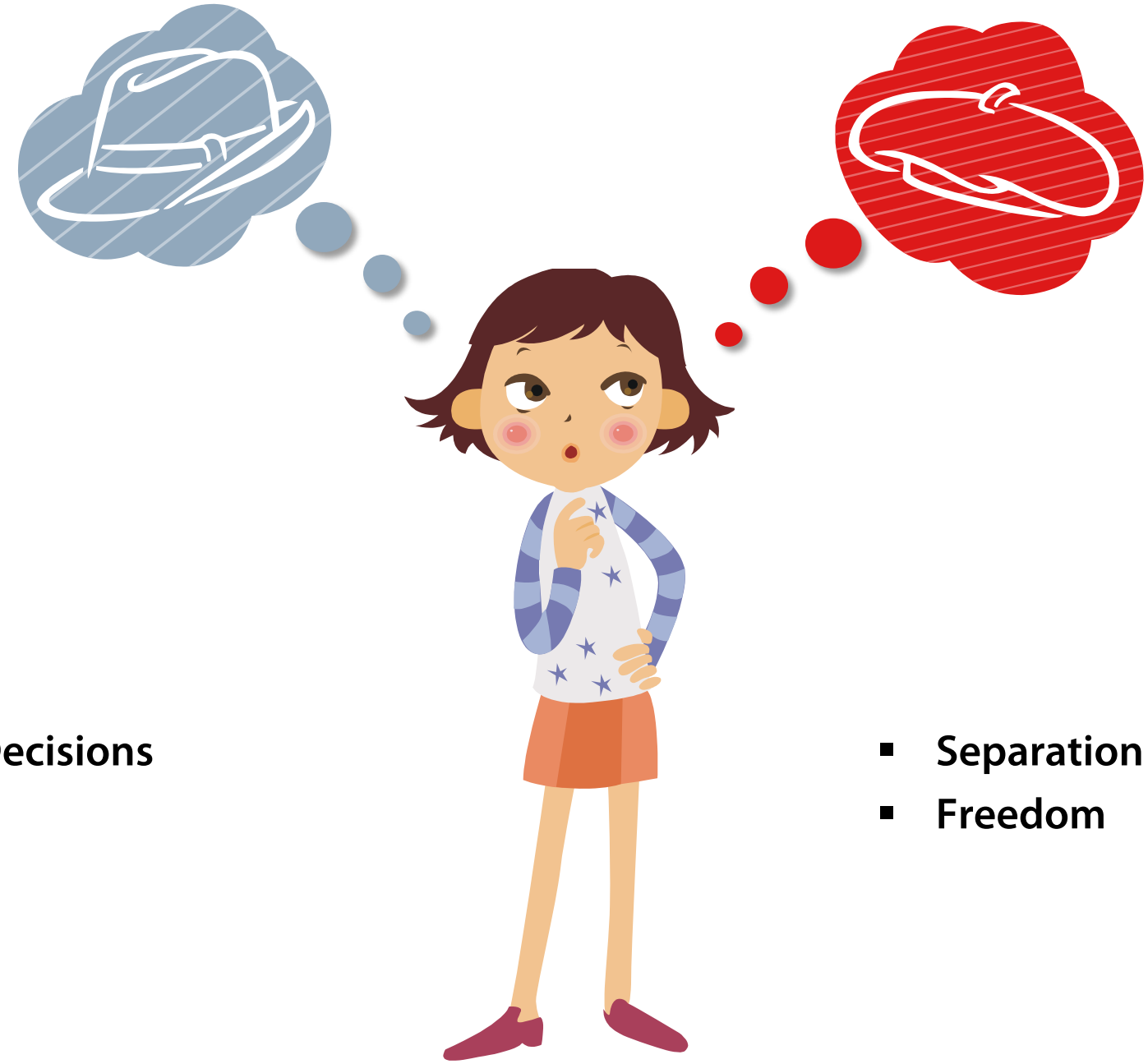
Behavioral Patterns

Michael L Perry
Michael@qedcode.com

pluralsight
hardcore developer training



Decisions

- **Separation**
- **Freedom**

Patterns

- **Separation - Commands**
  - Stateful → Relay Command
  - Stateless → Dependent Command
  - Reactive → Reactive Command
- **Freedom - Behaviors**
  - Attached Behaviors
  - Blend Behaviors
  - Trigger Actions
  - Targeted Trigger Actions

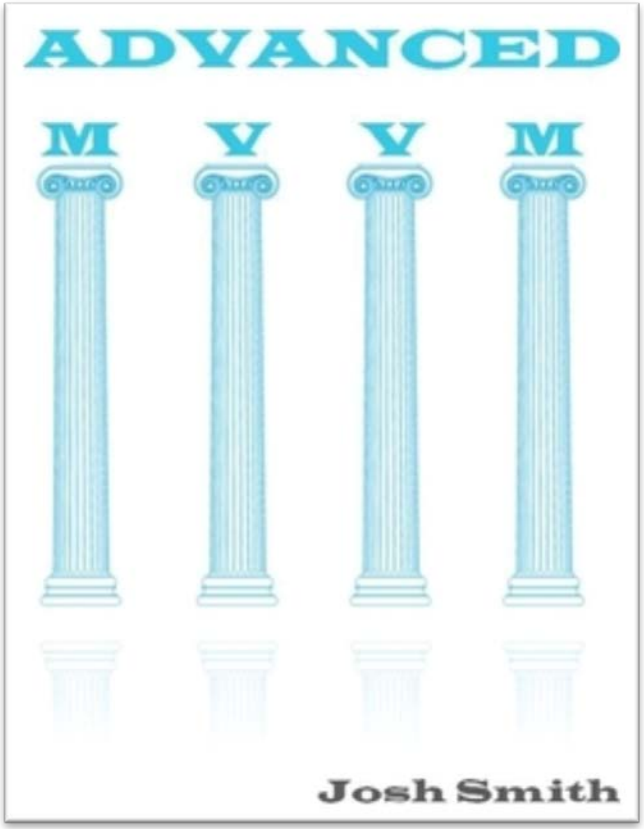
Relay Command

Commands without classes

ICommand

```
public interface ICommand
{
    void Execute(object parameter);
    bool CanExecute(object parameter);
    event EventHandler CanExecuteChanged;
}
```

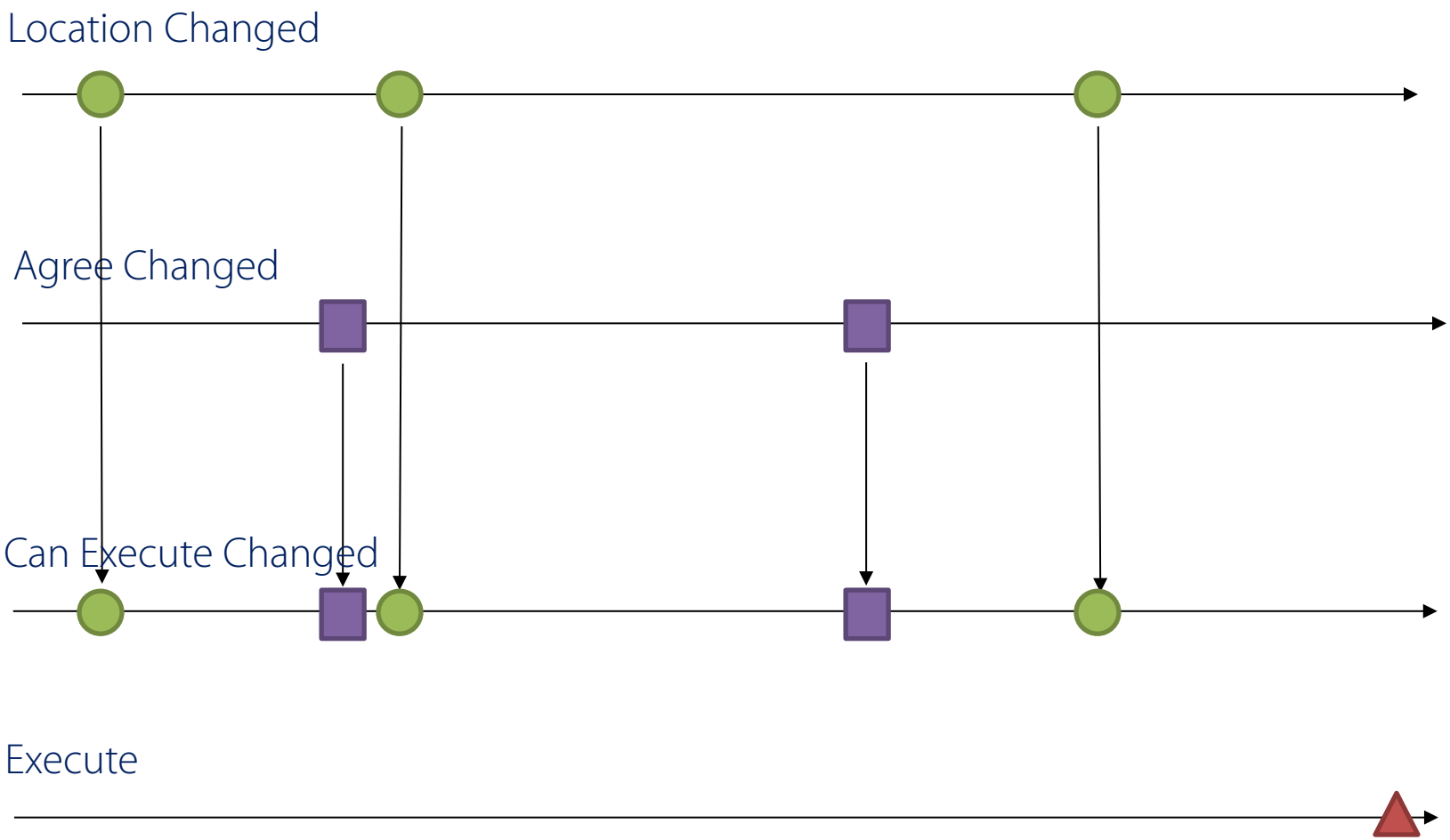
Josh Smith



Reactive Command

Compose streams of events

Streams


Location Changed
Agree Changed
Can Execute Changed
Execute

Dependent Command

Don’t repeat yourself

Conditions

```
!string.IsNullOrEmpty(Location) && Agree
```

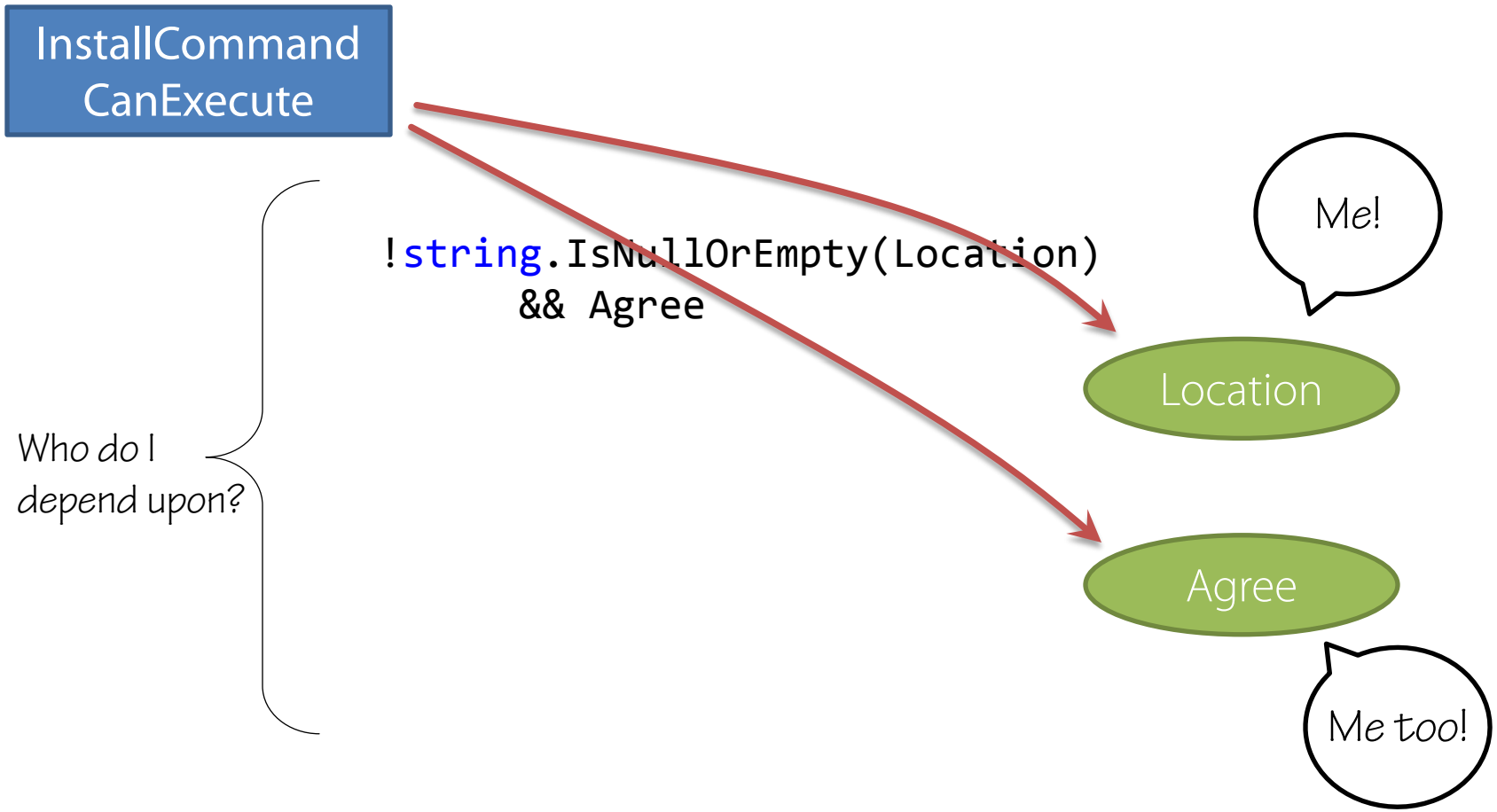
Relay Command (revisited)

```
_installCommand = new RelayCommand(
    () => Install(),
    () => !string.IsNullOrEmpty(Location) && Agree);

public string Location
{
    set
    {   // ...
        _installCommand.RaiseCanExecuteChanged();
    }
}

public bool Agree
{
    set
    {   // ...
        _installCommand.RaiseCanExecuteChanged();
    }
}
```

Dependency Tracking



Attached Behaviors

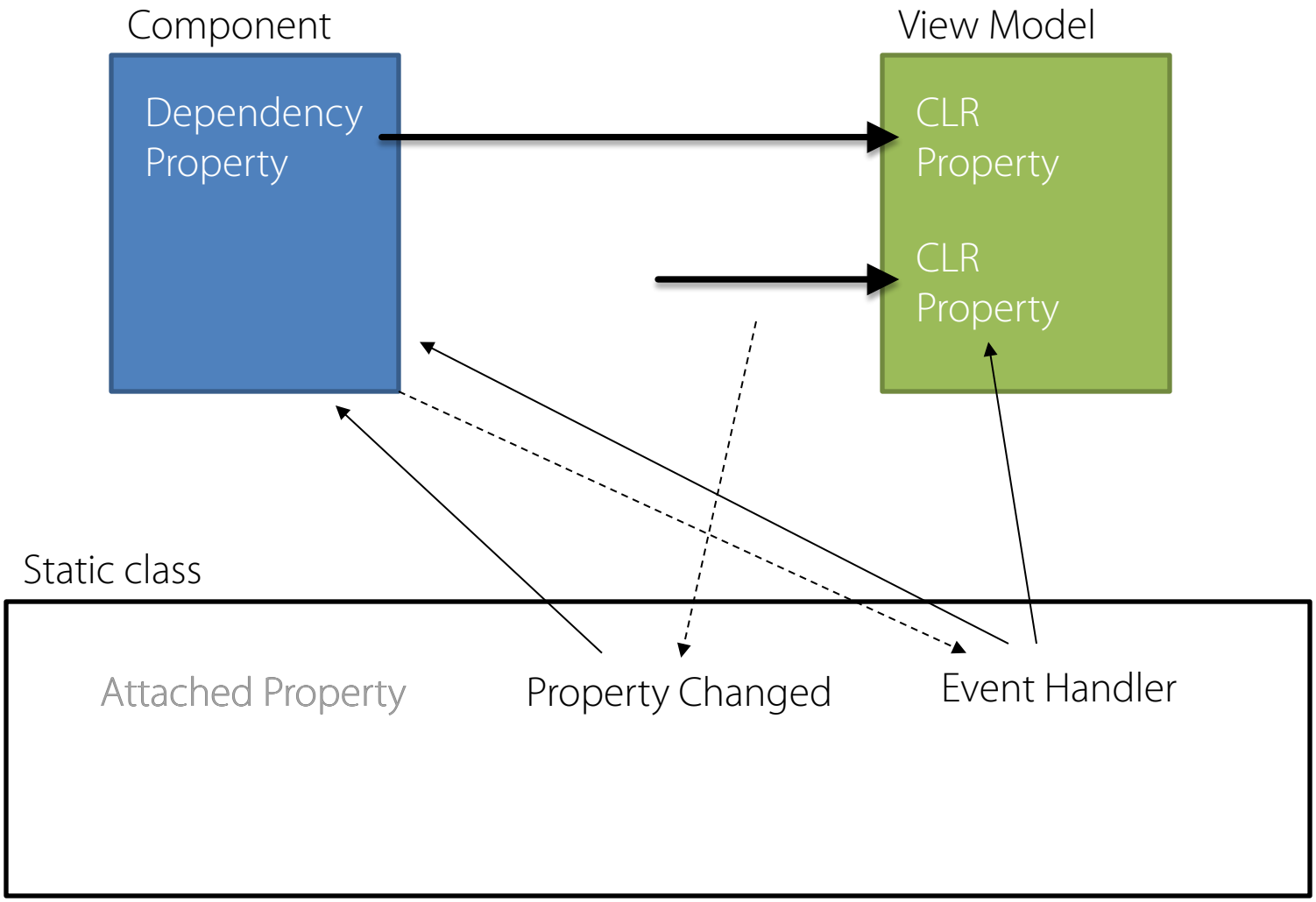
Extend existing controls

Attached Properties

```
<Grid>
    <Views:CategoryView
        Grid.RowSpan="2" />
    <Views:ProfileView
        Grid.Column="1" />
    <Views:FeedView
        Grid.Column="1"
        Grid.Row="1" />
    <Views:TickerView
        Grid.Row="2"
        Grid.ColumnSpan="2" />
</Grid>
```

Nikhil Kothari

Attached Behavior Pattern



Blend Behaviors

Apply behaviors in the designer

Evolution

- **Attached behaviors**
  - No encapsulation
  - Memory leaks
  - No designer support
- **Blend behaviors**
  - Components
  - Attach to objects
  - Designer support

Behavior

```
public abstract class Behavior<T>
    where T : DependencyObject
{
    protected T AssociatedObject { get; }
    protected virtual void OnAttached();
    protected virtual void OnDetaching();
}
```

… plus dependency properties

Trigger Actions

Reusable event handlers

TriggerAction

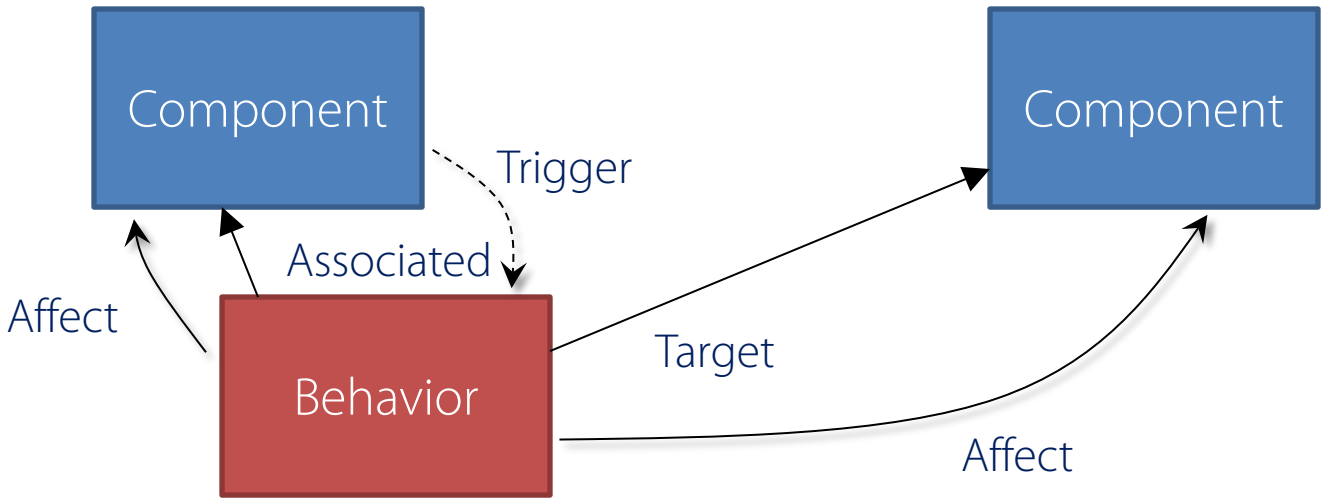
- Similar to Behavior
- Just for handling events

```
public abstract class TriggerAction<T>
    where T : DependencyObject
{
    protected T AssociatedObject { get; }
    protected virtual void OnAttached();
    protected virtual void OnDetaching();
    protected abstract void Invoke(object par);
}
```

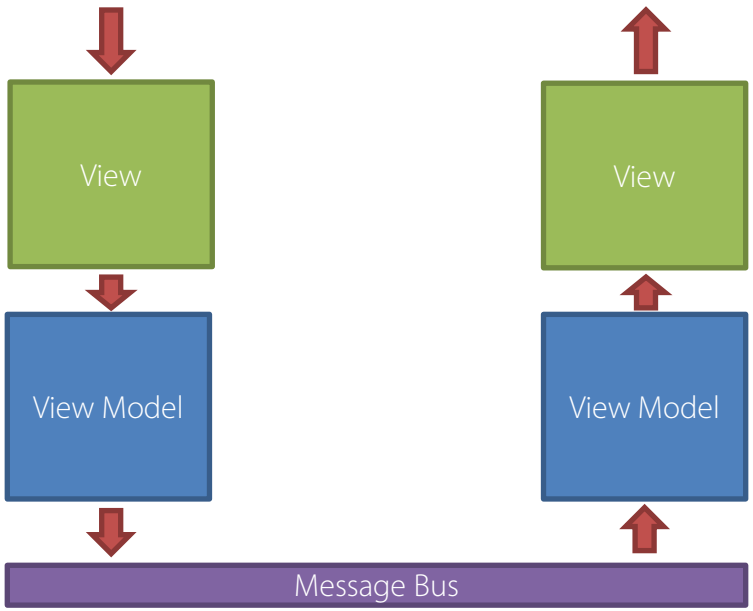
Targeted Trigger Actions

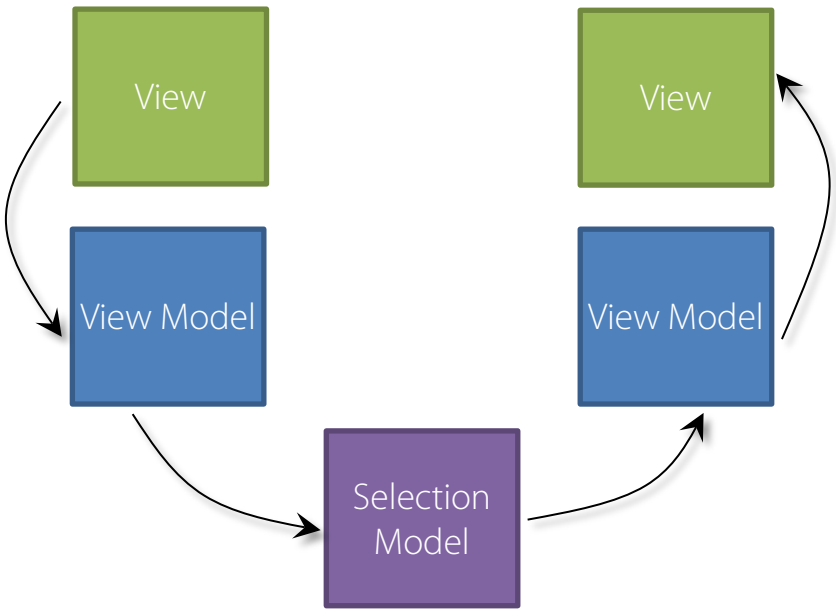
Interaction between components

Associated and Target Objects



Interaction Patterns





Message Bus

Selection Model

Summary

- **Commands**
  - Stateful → Relay Command
  - Stateless → Dependent Command
  - Reactive → Reactive Command
- **Behaviors**
  - Attached Behaviors
    - Early pattern
  - Blend Behaviors
    - General
  - Trigger Actions
    - Event handlers
  - Targeted Trigger Actions
    - Interactions
    - Favor MVVM