【C5】Delphi/C++Builderテクニカルセッション

Delphi/C++Builder MVVM アプローチ

エンバカデロ・テクノロジーズ
Malcolm Groves

embarcadero Developer Camp

Agenda

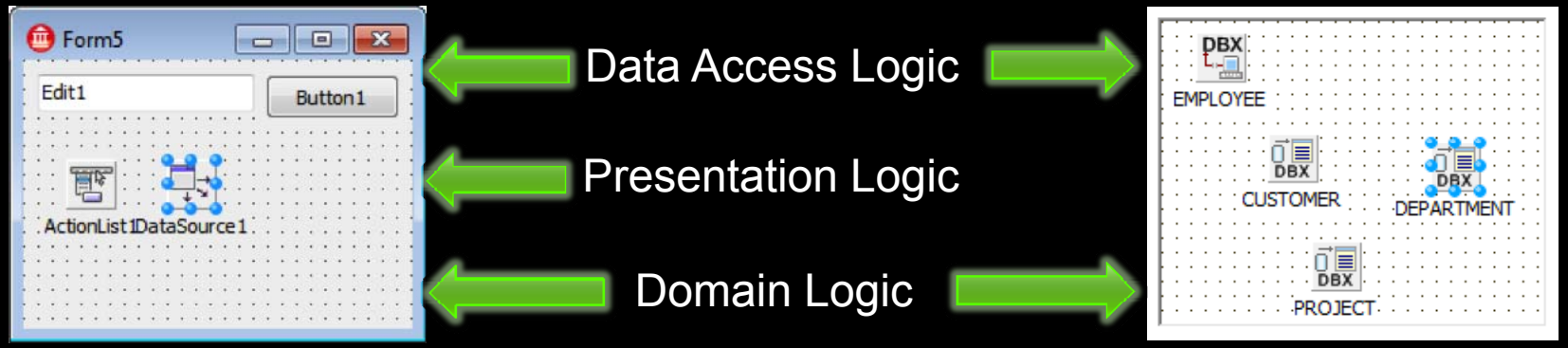
- What's the problem we're trying to solve?
- Different approaches to solving it.
- MVVM
  - What's a Model?
  - What's a View?
  - What's a View Model?
- Demo
  - Common Issues
- Things to think about
- More Resources

Three types of logic in an App

- Presentation Logic
- Domain Logic
- Data Access Logic

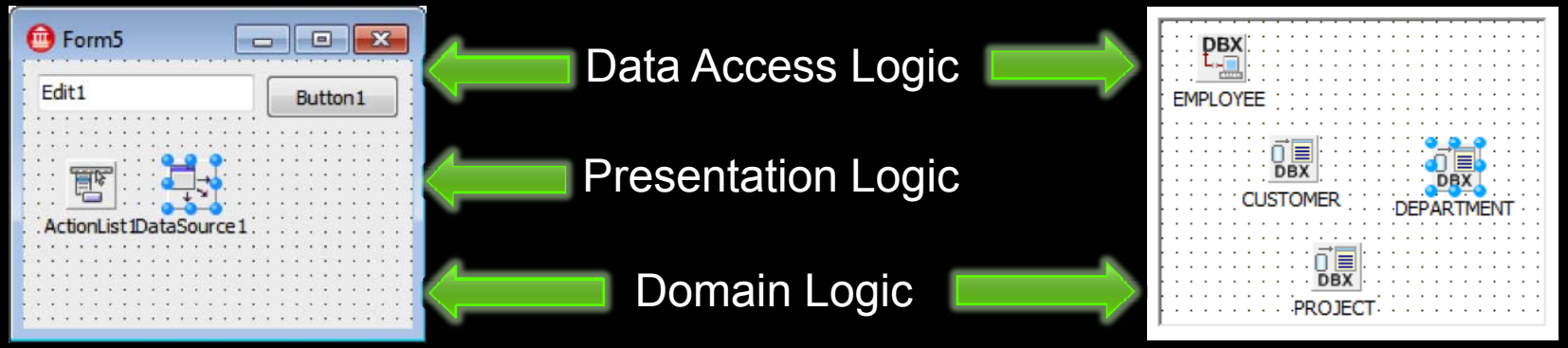
- Where you put it all matters. A LOT.

Traditional Delphi Approach



- Presentation Logic and Domain Logic mixed together in On… events
- Data Access logic either in Form or Data Modules or both

Traditional Delphi Approach

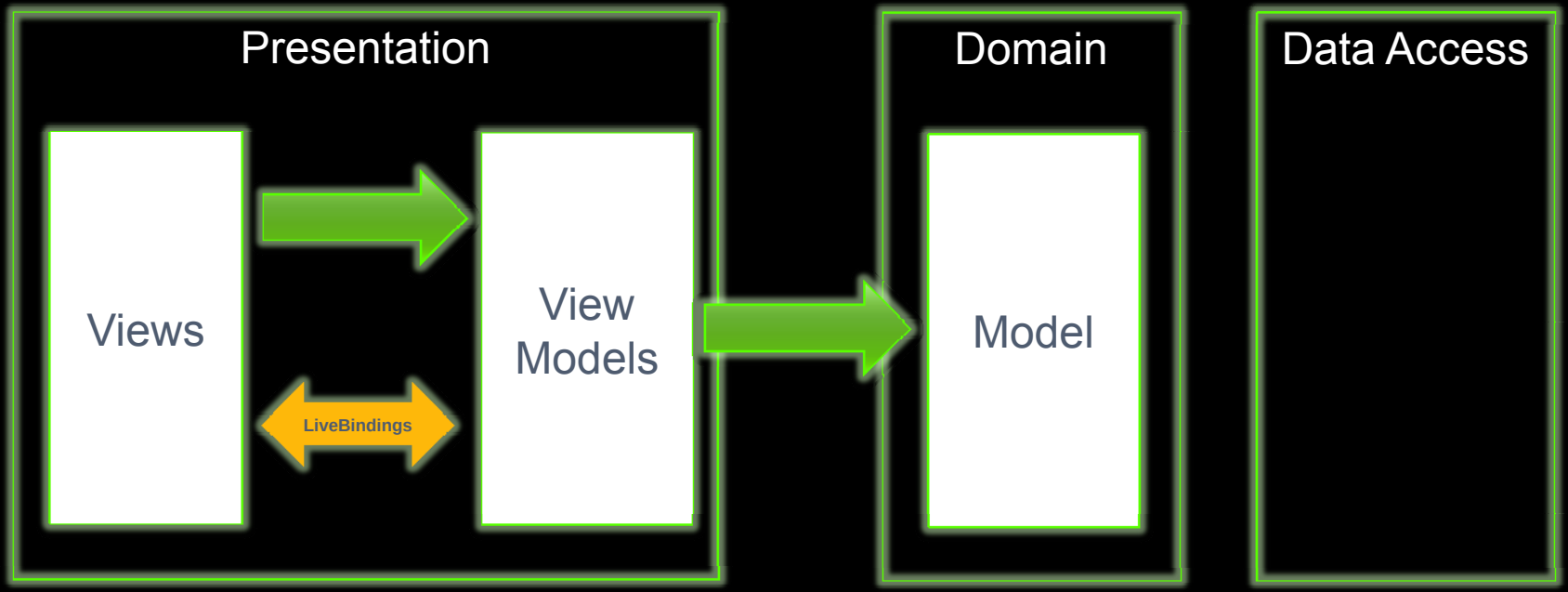


- Advantages
  - Quick to Develop
  - Well understood
- Disadvantages
  - Hard to maintain. Brittle.
  - Hard to replace UI with a new one.
  - Hard to test.

Not a new problem

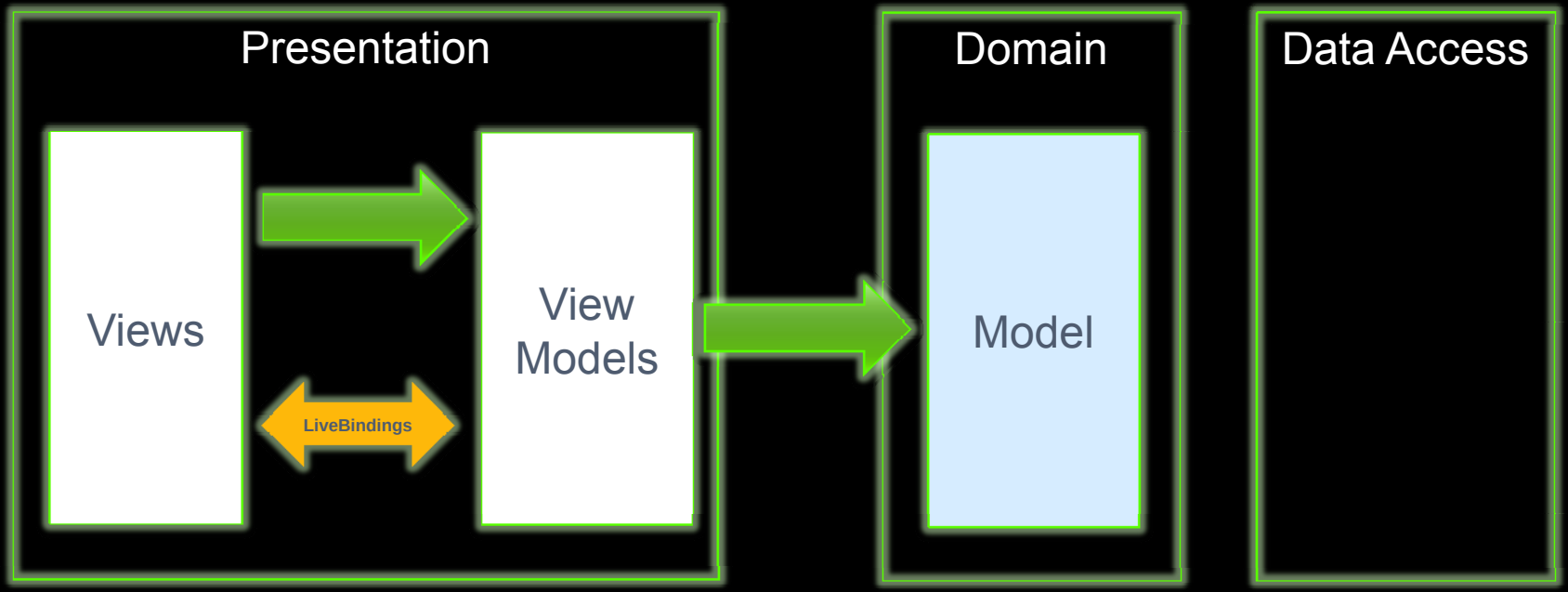
- Lots of Presentation Patterns
  - MVC – Model-View-Controller
  - MVP – Model-View-Presenter
  - MVVM – Model-View-ViewModel
    - Based on Presentation Model by Martin Fowler
      - In turn, based on Application Model from SmallTalk

MVVM is about the Presentation Layer



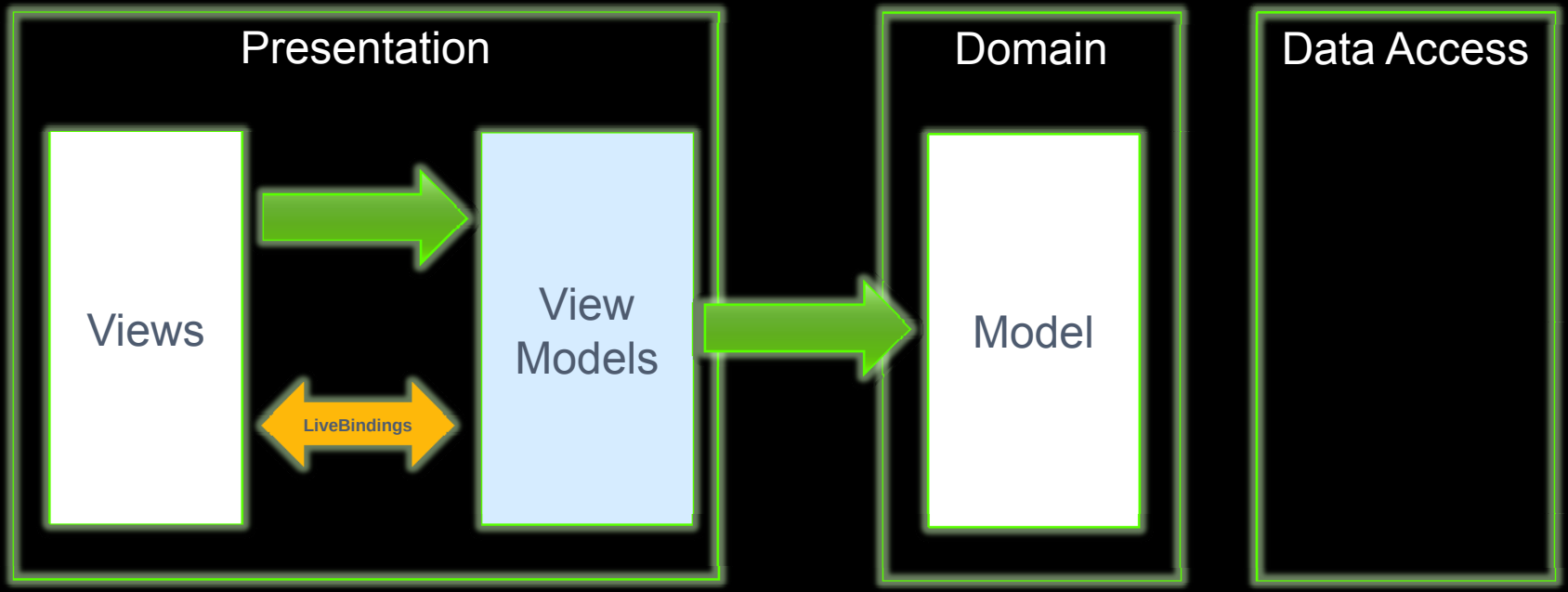
- Model is ignorant of Views and View Models
- View Models are ignorant of Views

MVVM - Model



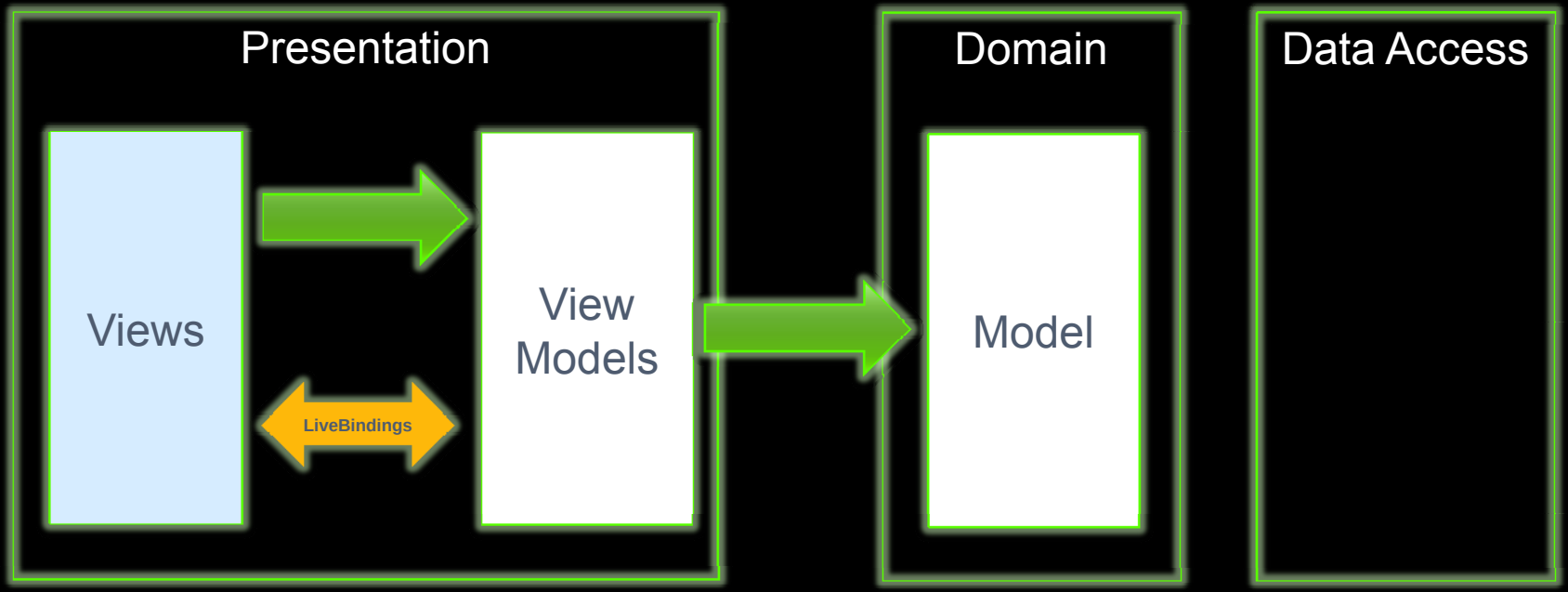
- Model – Business Domain of your App
  - In iTunes it's all about artists, songs, plays, ratings, etc
  - In an Accounting App it's Accounts, Transactions, etc
  - In SalesForce it's Customers, Activities, Forecasts, Opportunities, etc
- Important: Model can be Unit Tested!

MVVM – View Model



- View Model – literally the Model of your View (UI)
  - The data that is acted upon in this View
  - The workflows/processes that are invoked in this View
  - The UI rules that are built into this View
- Important: ViewModels can be Unit Tested!

MVVM - View

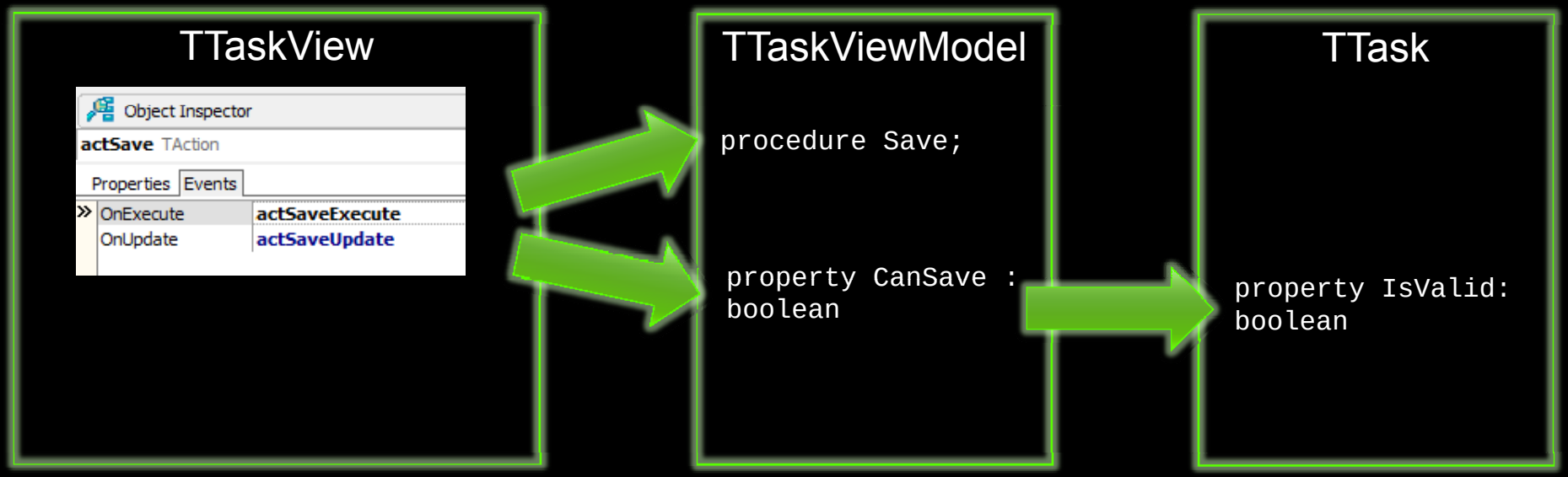


- View – The visual rendering of a View Model
  - LiveBindings to bind the ViewModel data to the UI
  - Minimum code needed to invoke methods in the ViewModel
  - Code specific to THIS View
- Important: Views are hard to Unit Test
  - So, the only Good View is a Dumb View!

Quick Example Tour - MenialTasks

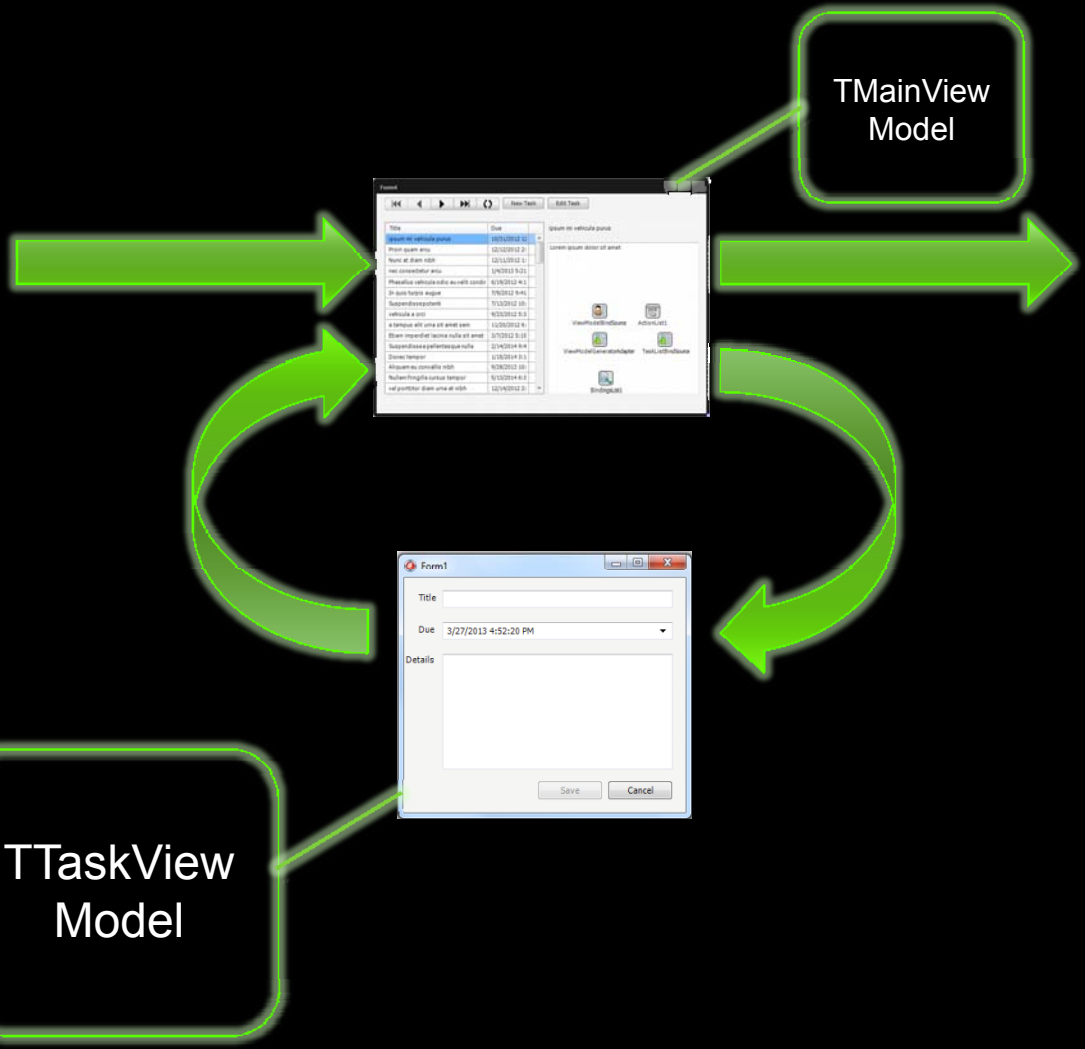
- Disclaimers:
  - Example is focused on introducing MVVM, nothing else
  - Purposely simple
    - Bare minimum of features to show key MVVM topics
  - No Persistence

Example : Actions

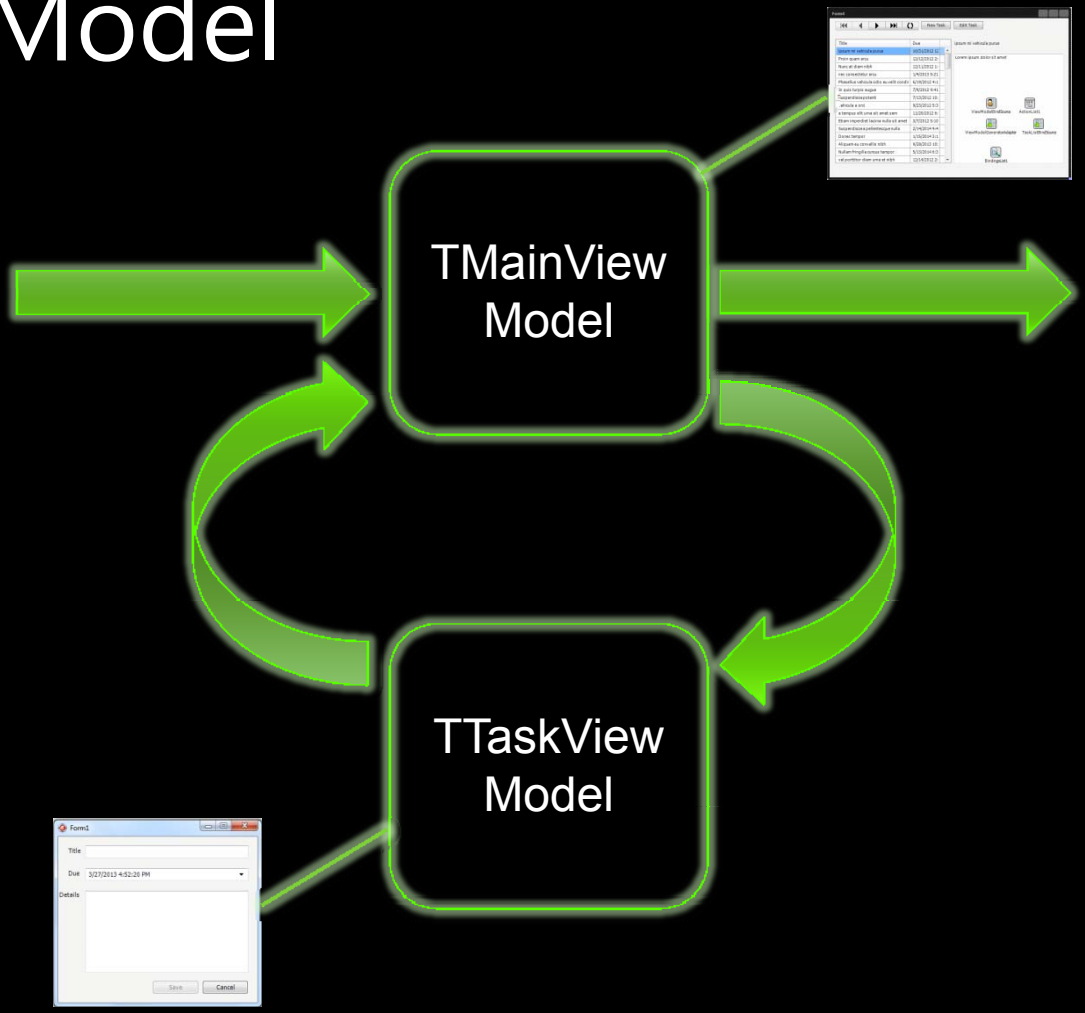


- ViewModel decides if it is OK to Save given the current state
- View decides how to represent that in the UI, eg. Disabling the button

View First / Dominant View

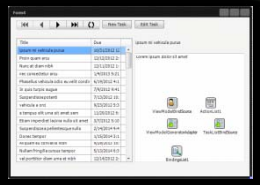


ViewModel First / Dominant ViewModel



Example: Displaying a child Form

TMainView



1. At startup, View provides the ViewModel an Anonymous Method to call when a Child View is needed
2. Sometime later, the Add Task button is clicked

`ViewModel.AddNewTask`

TMainViewModel

3. ViewModel creates the child ViewModel, and calls the Anonymous Method, passing the child ViewModel as a parameter

`OnEditTask invoked`

4. The anonymous method creates the child View, passing it the child ViewModel, then shows it.
5. When the child View closes, the anonymous method cleans up, refreshes bindings, etc, as needed.

`OnEditTask returns`

6. When the anonymous method returns, the ViewModel still has access to the child ViewModel, so can access its data if required

ViewModel First vs View First

- \- Less obvious at first
- \- A few more hoops to jump through
- \+ Views stay dumb
- \+ More logic becomes unit testable

- \+ More closely resembles traditional Delphi model.
- \+ Simpler to understand
- \- Views end up smarter than I like
- \- Hard to Unit Test View interactions

Things to think about:

- Pure Views
  - The less code you write in your View, the less there is untested
- View First vs View Model First?
- View <> Form
  - A View could be a Panel, a Frame, a control…

Things to think about:

- MVVM doesn't dictate IoC, but it can simplify your code and your tests.
- You don't need a MVVM Framework
  - But you'll probably end up writing one. Everyone else does.
- Try it. Everything is clumsy at first. If MVVM doesn't work for you, try MVC, MVP, etc.

More Information

- Series of more detailed blog posts on MVVM in Delphi over the coming months at www.malcolmgroves.com
- MenialTasks source online at https://github.com/malcolmgroves
- Other Resources (list online at http://mgrov.es/mvvmlist)
  - Books
    - Developer's Guide to Microsoft Prism http://msdn.microsoft.com/en-us/library/gg406140.aspx
    - Advanced MVVM http://amzn.com/B0038KX9FW
  - Articles
    - WPF Apps with the Model-View-ViewModel Pattern – Josh Smith http://msdn.microsoft.com/en-us/magazine/dd419663.aspx
  - Videos
    - Build Your Own MVVM Framework – Rob Eisenberg (MIX10 Conference) http://channel9.msdn.com/Events/MIX/MIX10/EX15

Thank you!

embarcadero®
Developer Camp