Fields in Stateful Model

By Thomas Kwan

What is FISM

- Software architectural pattern promotes more efficient and manageable GUI development
- Specialization of MVVM which is a variant of MVC
- Redefined the View Model of MVVM

MVP / MVVM / FISM

	MVP	MVVM	FISM
View / UI	View	View	Physical UI Layer
Mediator	Presenter	View Model	Logical UI Layer
Data Object	Model	Model	Data Model

MVVM Advantage

- Maintainability
- Testability
- Extensibility

MVVM Disadvantage

1. For simple UI, set up cost is high.
2. In bigger cases, hard to design ViewModel.
3. Debugging would be bit difficult when we have complex data bindings.

FISM Physical UI Layer

- Analogous to MVVM VIEW
- UI Implementation at end user visible level.
- With the same underlying Logical UI Layer
- Different Physical UI Layer can be implemented
- E.g. Angular/React/Android/iOS/WPF….

FISM Logical UI Layer

- Analogous to MVVM View Model
- Defines structures and building blocks
- FISM View Model (don’t confuse with MVVM View Model) and FSIM View Field
- FISM View Model and View Field forms extensible Folder-File like structure

FISM Logical UI Layer

- View Field carries all info of primitive UI widgets. E.g. Dropdown List, Input Box,etc
- View Model encapsulates View Fields
- View Model can have children View Model
- The View Model-Field structure should be logical very close (better exactly) to Physical UI

FISM Data Model

- Analogous to MVVM Model
- Biz data object layer
- MUST implement de/serialization
- In such as way that the Logical UI Layer and thus the Physical UI Layer can be Perfectly De / Serialisable and thus the system can be called STATEFUL

FISM Pros – MVVM Cons

- Building the Logical UI Layer is very straight forward
- Just follow the structure of End User UI Requirement
- Implementing using View Model – View Field building blocks

FISM Pros – MVVM Cons

- Debugging is very easy
- Binding is very simple because Physical / Logical UI are exactly the same structure
- Binding must be one to one mapping
- Debug at Logical UI level is very easy
- Rarely need to debug Physical UI level

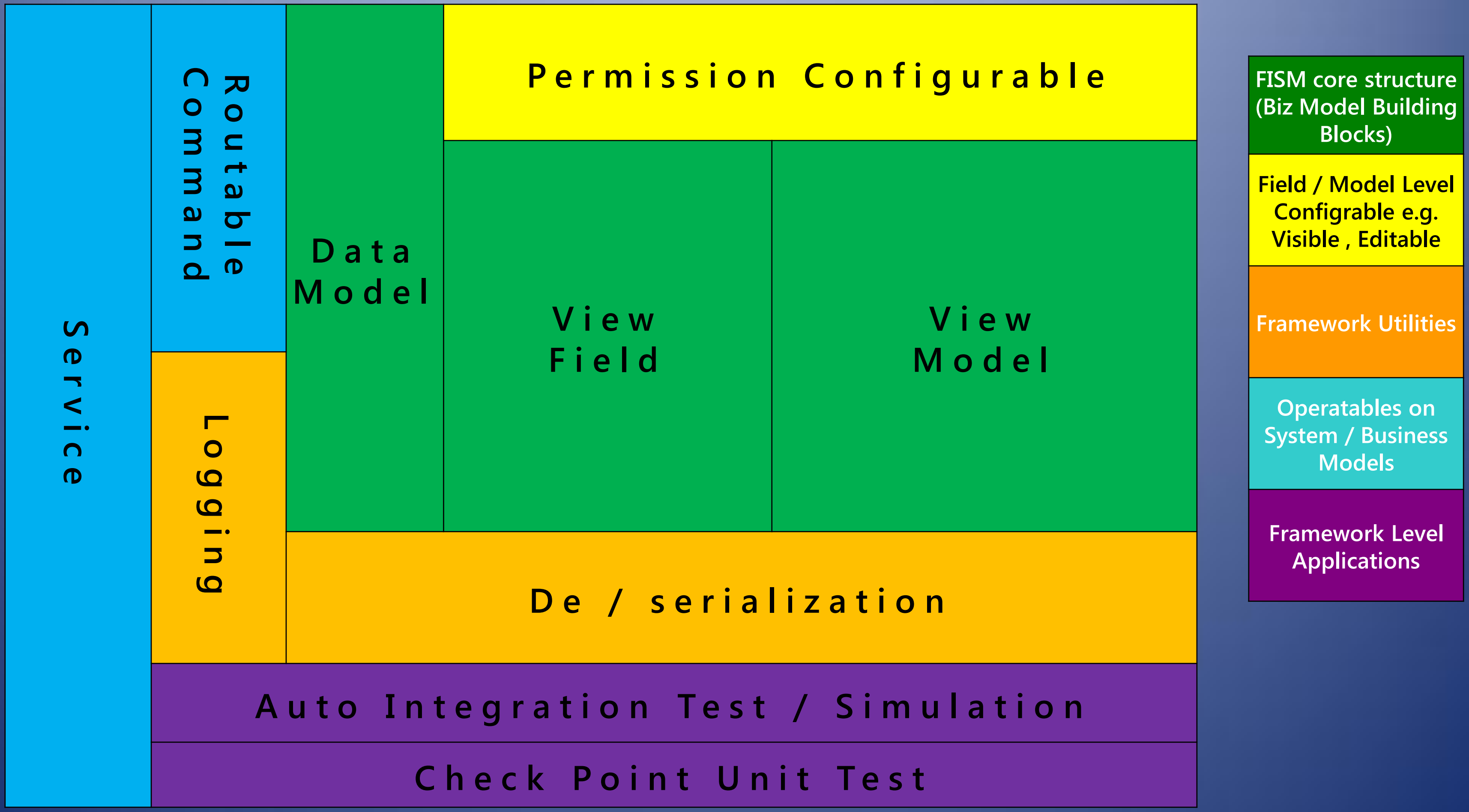
FISM Pro - Standardization

- Standardize the building blocks
- Framework level manipulation is possible
- Configurbility (vs. hard coding) of attributes (e.g. value / visibility /editability) of any level of Logical UI Layer (down to View Field) is a MUST

TomFism

- Implementation of basic infrastructure of FSIM using typescript
- Infrastructure Data Model Level
- Infrastructure for Logical UI Level
- Infrastructure for App Level Structure
- Utilities for STATEFUL etc
- UI technology independent

TomFism Composition



Service

Routable Command

Logging

Data Model

Permission Configurable

View Field

View Model

De / serialization

Auto Integration Test / Simulation

Check Point Unit Test

FISM core structure (Biz Model Building Blocks)

Field / Model Level Configrable e.g. Visible , Editable

Framework Utilities

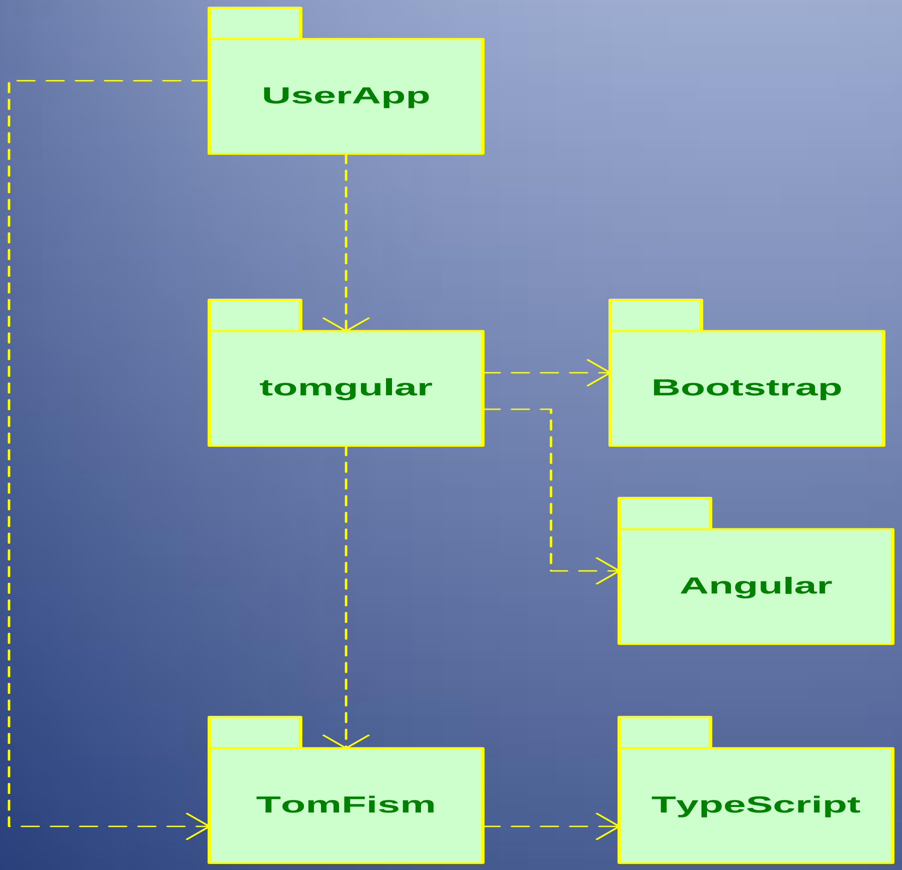
Operatables on System / Business Models

Framework Level Applications

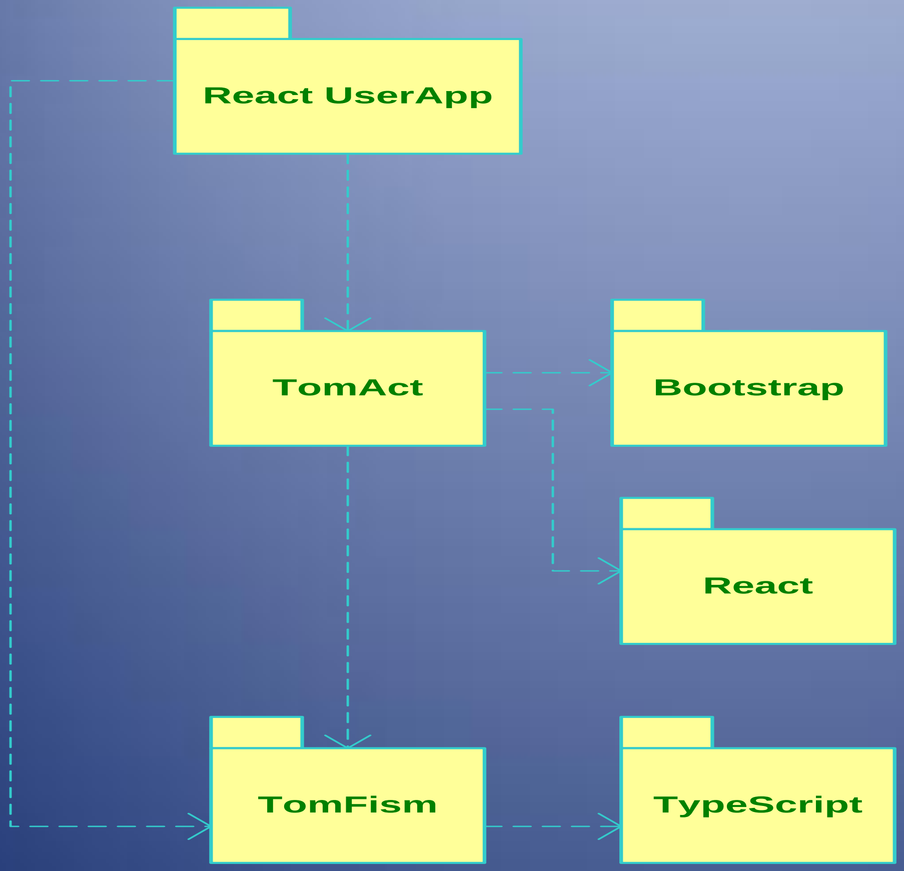
Tomgular

- Implementation of a Physical UI Layer
- Using Angular 7

Package Dependency (Tomgular App)



Package Dependency (TomAct App)


React UserApp
TomAct
Bootstrap
React
TomFism
TypeScript