Model-View-ViewModel (MVVM)

Model-View-ViewModel

Model

Represents and holds **raw** data

- Some of this data, in some form, may be displayed in the view
- Can also contain logic to retrieve the data from some source
- Contains no logic associated with displaying the model

Model-View-ViewModel

View

User interface

- ✧ In a web app, it’s just the HTML and CSS
- ✧ Only displays the data that it is given
- ✧ Never changes the data
- ✧ Declaratively broadcasts events, but never handles them

Model-View-ViewModel

ViewModel

Representation of the state of the view

- ✧ Holds the data that's displayed in the view
- ✧ Responds to view events, aka **presentation logic**
- ✧ Calls other functionality for business logic processing
- ✧ Never directly asks the view to display anything

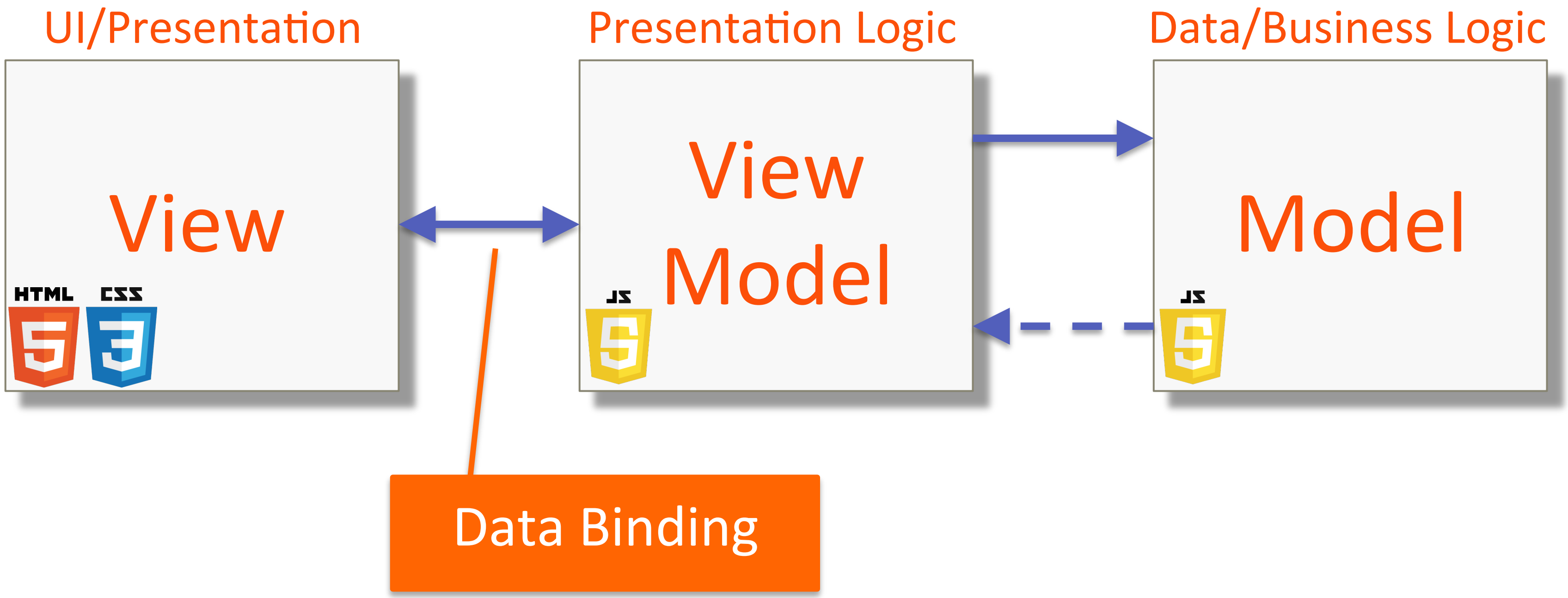
Model-View-ViewModel

Declarative Binder

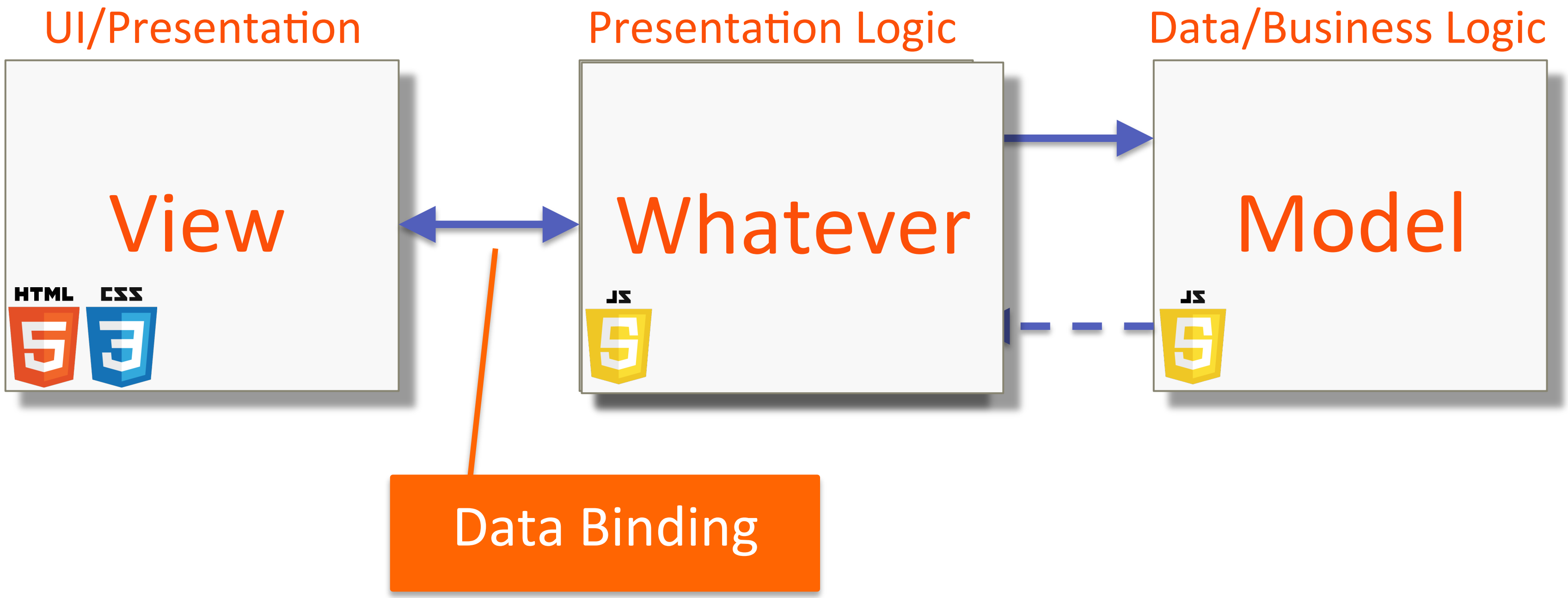
Declaratively binds the model of the ViewModel to the View

- Declaratively means you don't have to write **any** code
  - The framework does this "magic"
- Key enabler of the whole MVVM pattern

Model-View-ViewModel



Model-View-**Whatever** or **MV***



Summary

- Design Patterns are cookie-cutter approaches to common software development issues
- MVVM is a common design pattern for structuring Uis with functionality
- Model – represents and holds raw data
- View – UI that never changes data, declares events
- ViewModel – data representation of the state of the UI
- Declarative Binder – binds ViewModel to the View
- Angular is not restricted to MVVM