Stanford CS193p

Developing Applications for iOS
Fall 2010

Stanford
CS193p
Fall 2010

Today

- Logistics
  - Lectures
  - Communication
  - Homework / Final Project
  - Requirements
- iOS Overview
- MVC
  - Object-Oriented Design Concept

Logistics

- **Lectures**
  - Tell you (Tuesday)
  - Show you (Thursday)
  - Let you do it yourself (Homework)
- **Friday TA Section**
  - Practical matters (e.g. debugging)
  - KPCB Entrepreneurship
  - Special topics (guest lecturers)
- **Communication**
  - http://cs193p.stanford.edu
  - Lectures, demo code and homework
  - Class announcements
- **Homework**
  - 7 weekly assignments
  - Assigned Thursday after lecture
  - Due the following Wednesday at 11:59pm
  - Individual work only
- **Final Project**
  - 3 weeks to work on it
  - Proposal requires instructor approval
  - Some teams of 2 might be allowed
  - Keynote presentation required (3 mins or so)

Requirements

- Must have a Mac
  - Intel-based
  - Snow Leopard
- Hardware
  - Not required for homework
  - Required for final project (iOS4 or iPad)
  - iPod Touch loaners available
- Textbook
  - Apple on-line documentation
  - http://developer.apple.com
- Prerequisites
  - Object-Oriented Programming
  - CS106A&B required, CS107 recommended
- Object-Oriented Terms
  - Class (description/template for an object)
  - Instance (manifestation of a class)
  - Message (sent to objects to make them act)
  - Method (code invoked by a Message)
  - Instance Variable (object-specific storage)
  - Inheritance (code-sharing mechanism)
  - Superclass/Subclass (Inheritance relationships)
  - Protocol (non-class-specific method declaration)
- You should know these!
  - If you are not very comfortable with all of these, this might not be the class for you

iOS4 SDK

- Required for all homework assignments and final project
  - It's free!
  - Download Xcode/SDK from iOS Dev Center at http://developer.apple.com
  - To run on a device (not just in simulator), you must join a Program
  - iOS Developer University Program = free for Stanford students = Device YES, AppStore NO
  - Normal Developer Program = $99/year = Device YES, AppStore YES
- iOS Developer University Program
  - Enrolled students will receive an invitation to their Stanford e-mail accounts
  - Follow the directions to join the Program and download the Xcode/SDK (if you haven't already)
  - Submit your UDID to the staff via e-mail
  - Valid through the end of the quarter only

What will I learn in this course?

- How to build cool apps
  - Easy to build even very complex applications
  - Result lives in your pocket!
  - Very easy to distribute your application through the AppStore
  - Vibrant development community
- Real-life Object-Oriented Programming
  - The heart of Cocoa Touch is 100% object-oriented
  - Application of MVC design model
  - Many computer science concepts applied in a commercial development platform:
    - Databases, Graphics, Multimedia, Multithreading, Animation, Networking, and much, much more!
  - Numerous students have gone on to sell products on the AppStore

Core OS

OSX Kernel	Power Management
Mach 3.0	Keychain Access
BSD	Certificates
Sockets	File System
Security	Bonjour

Core Services

Collections	Core Location
Address Book	Net Services
Networking	Threading
File Access	Preferences
SQLite	URL Utilities

Media

Core Audio	JPEG, PNG, TIFF
OpenAL	PDF
Audio Mixing	Quartz (2D)
Audio Recording	Core Animation
Video Playback	OpenGL ES

Cocoa Touch

- Multi-Touch
- Core Motion
- View Hierarchy
- Localization
- Controls
- Alerts
- Web View
- Map Kit
- Image Picker
- Camera

Platform Components

- Tools

Xcode

Interface Builder

Instruments

- Language

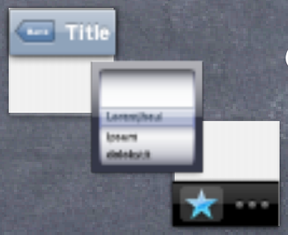
```
[display setTextColor:[UIColor blackColor]];
```

- Frameworks

Foundation

Core Data

Map Kit



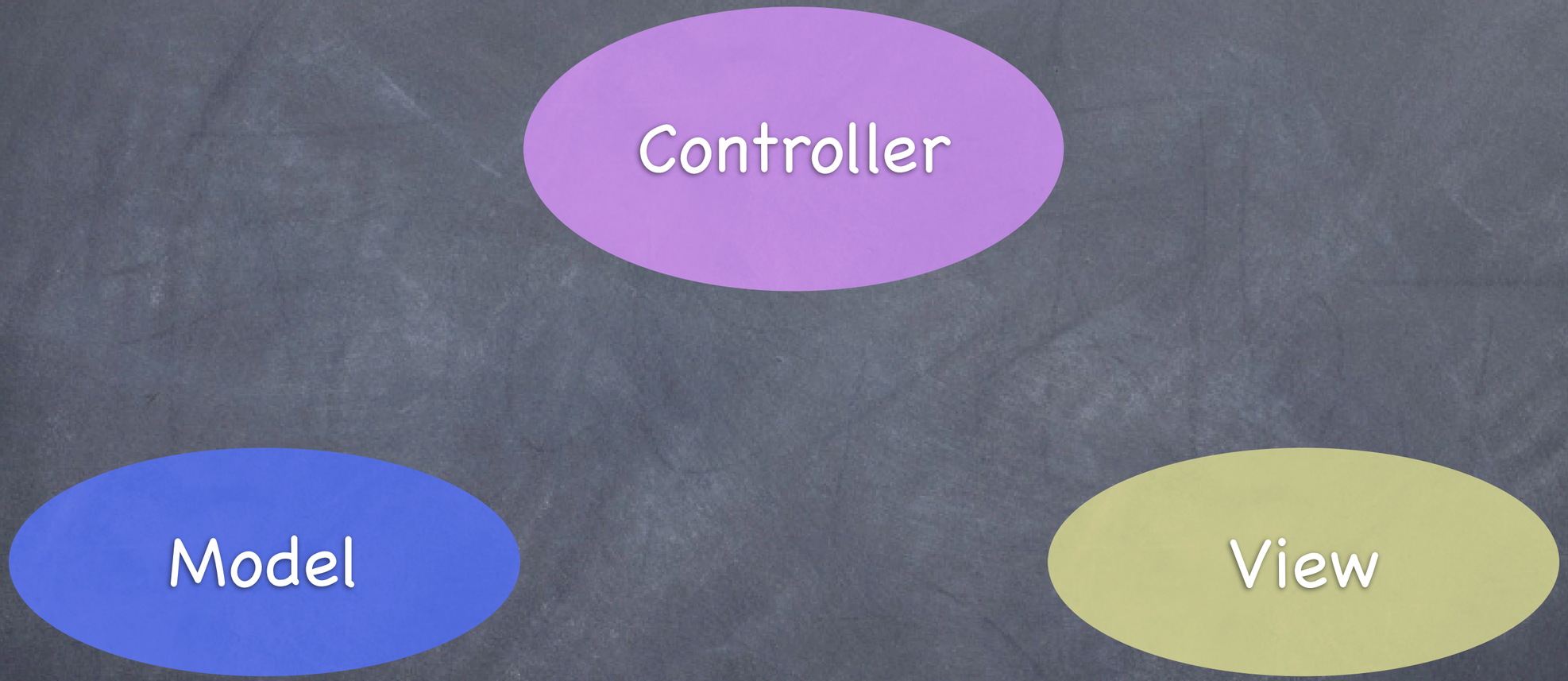
UIKit

Core Motion

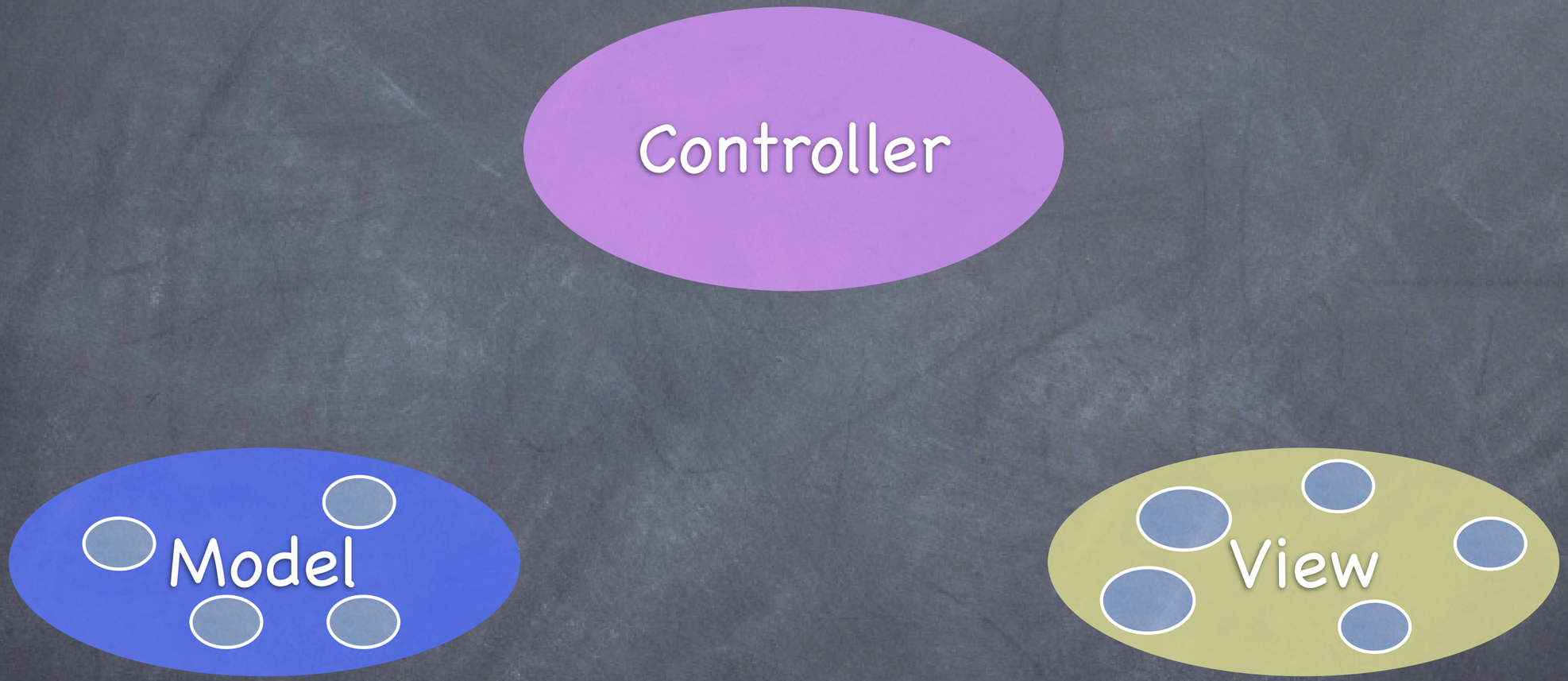
- Design Strategies

MVC

MVC

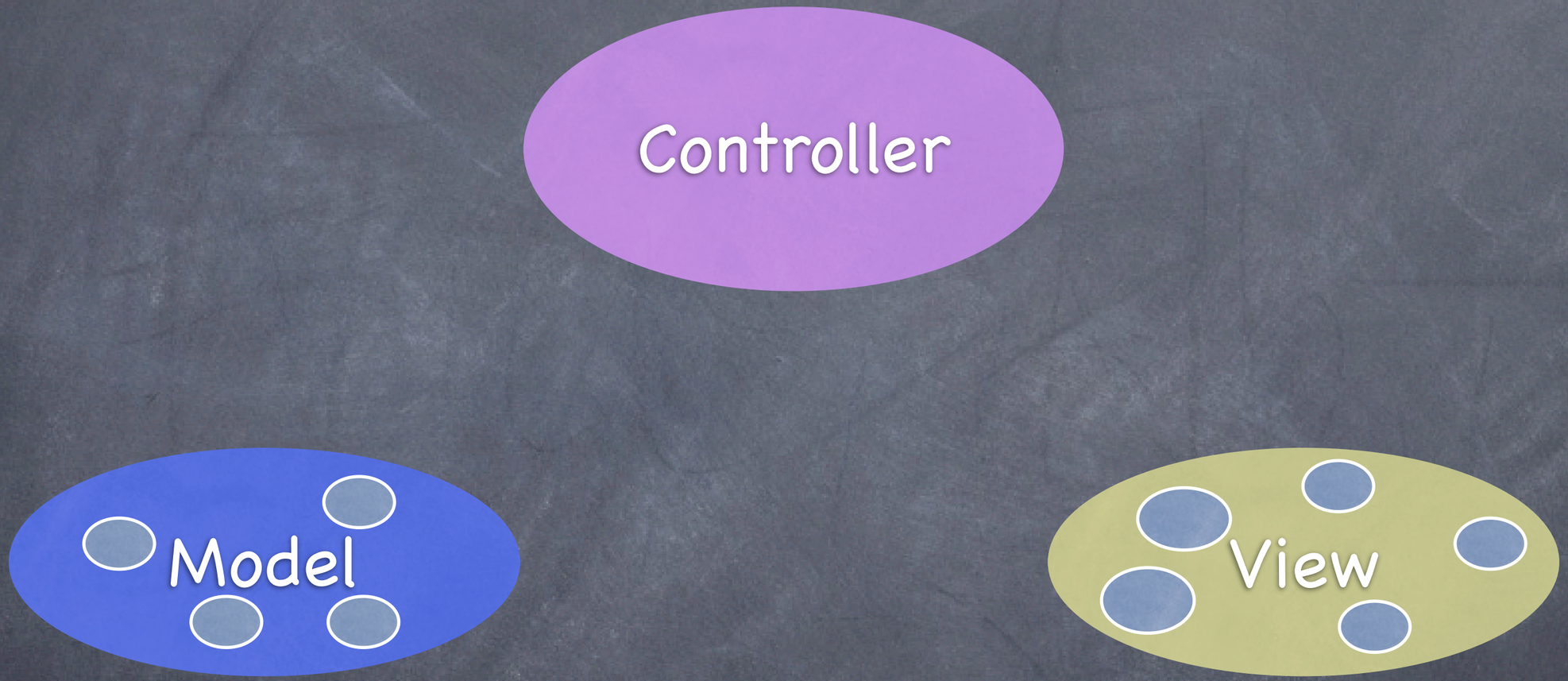


MVC



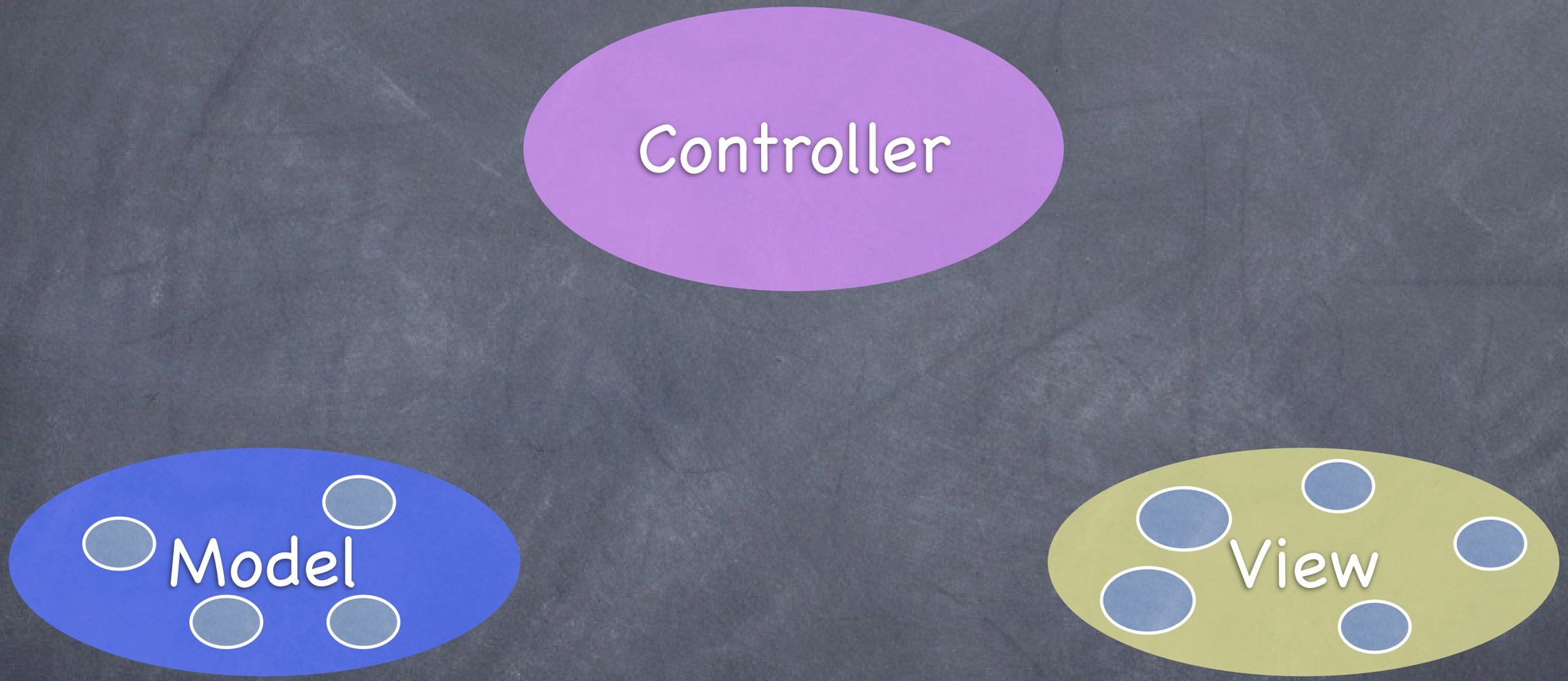
Divide objects in your program into 3 "camps."

MVC



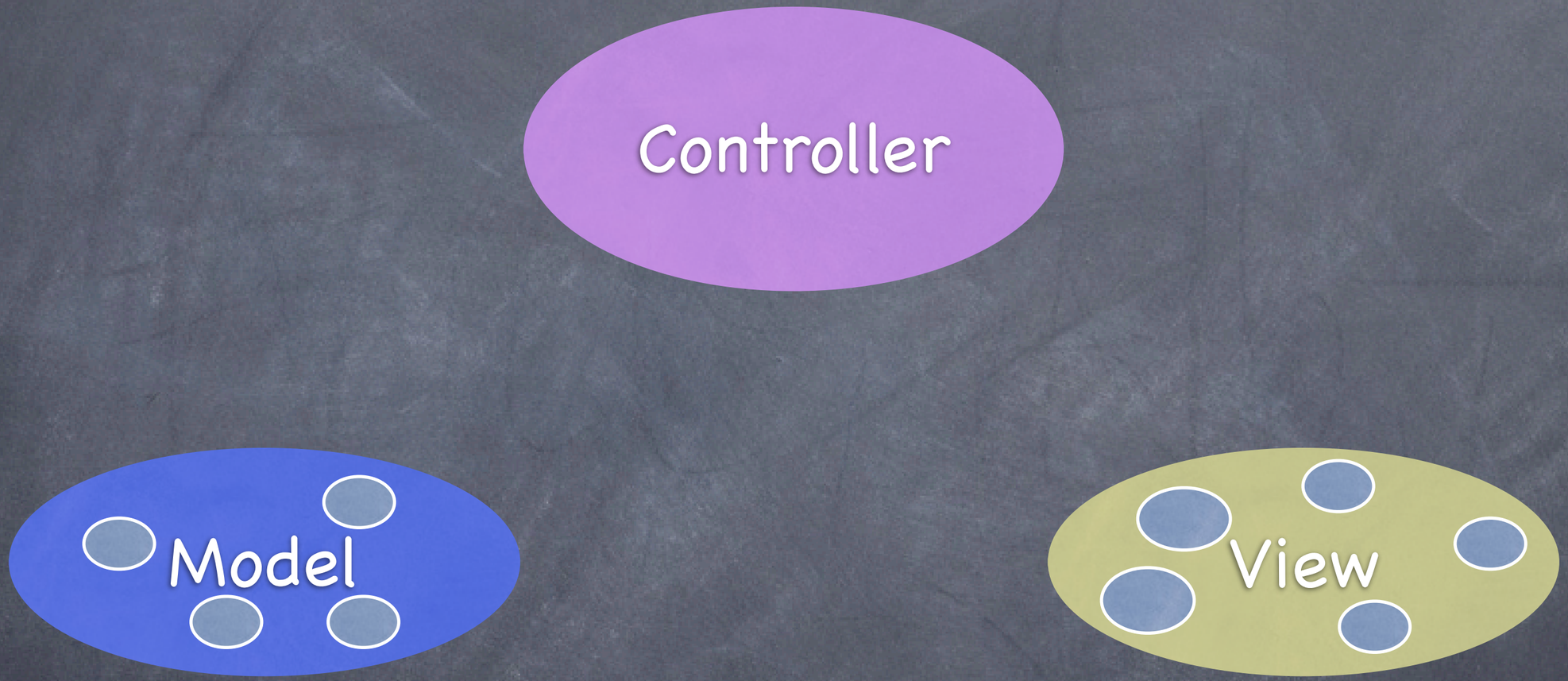
Model = What your application is (but not how it is displayed)

MVC



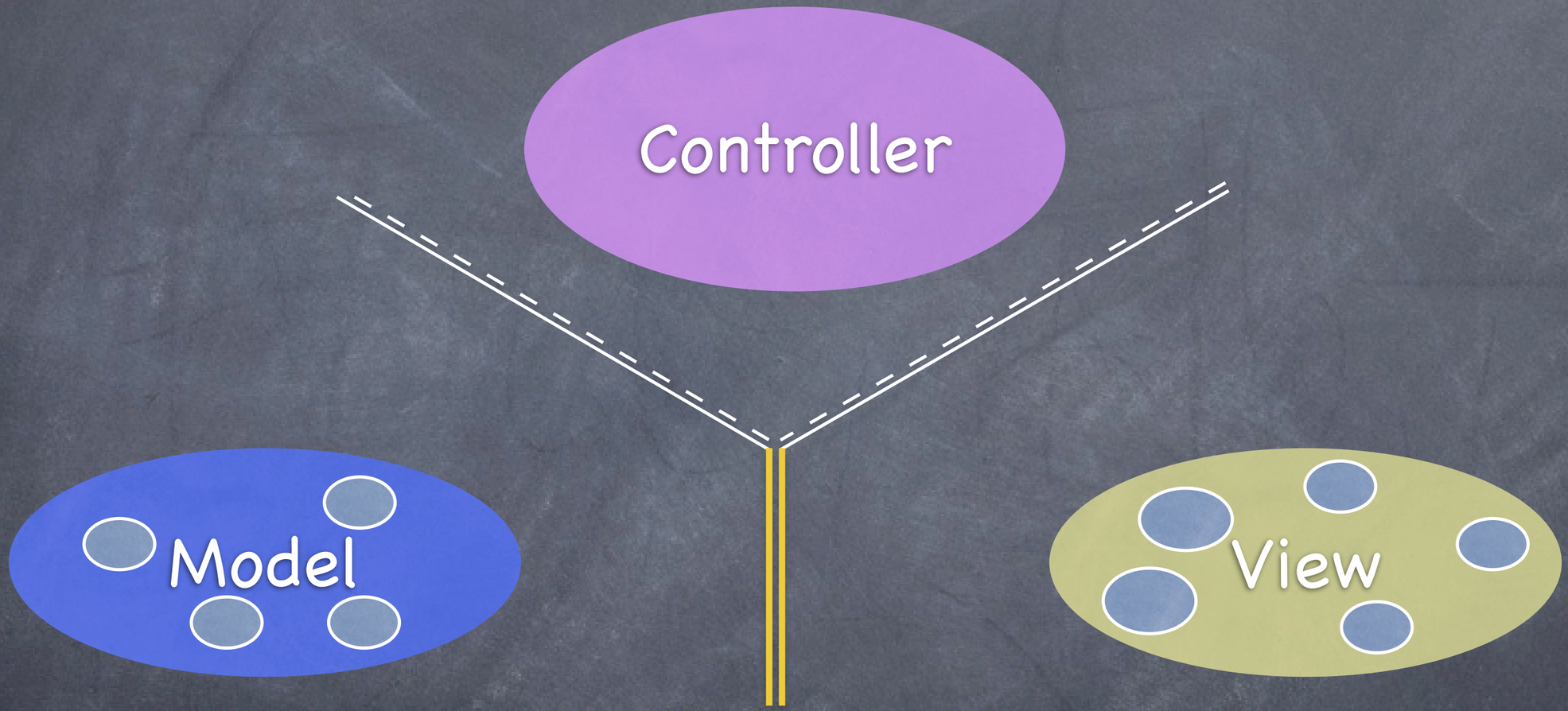
Controller = How your Model is presented to the user (UI logic)

MVC



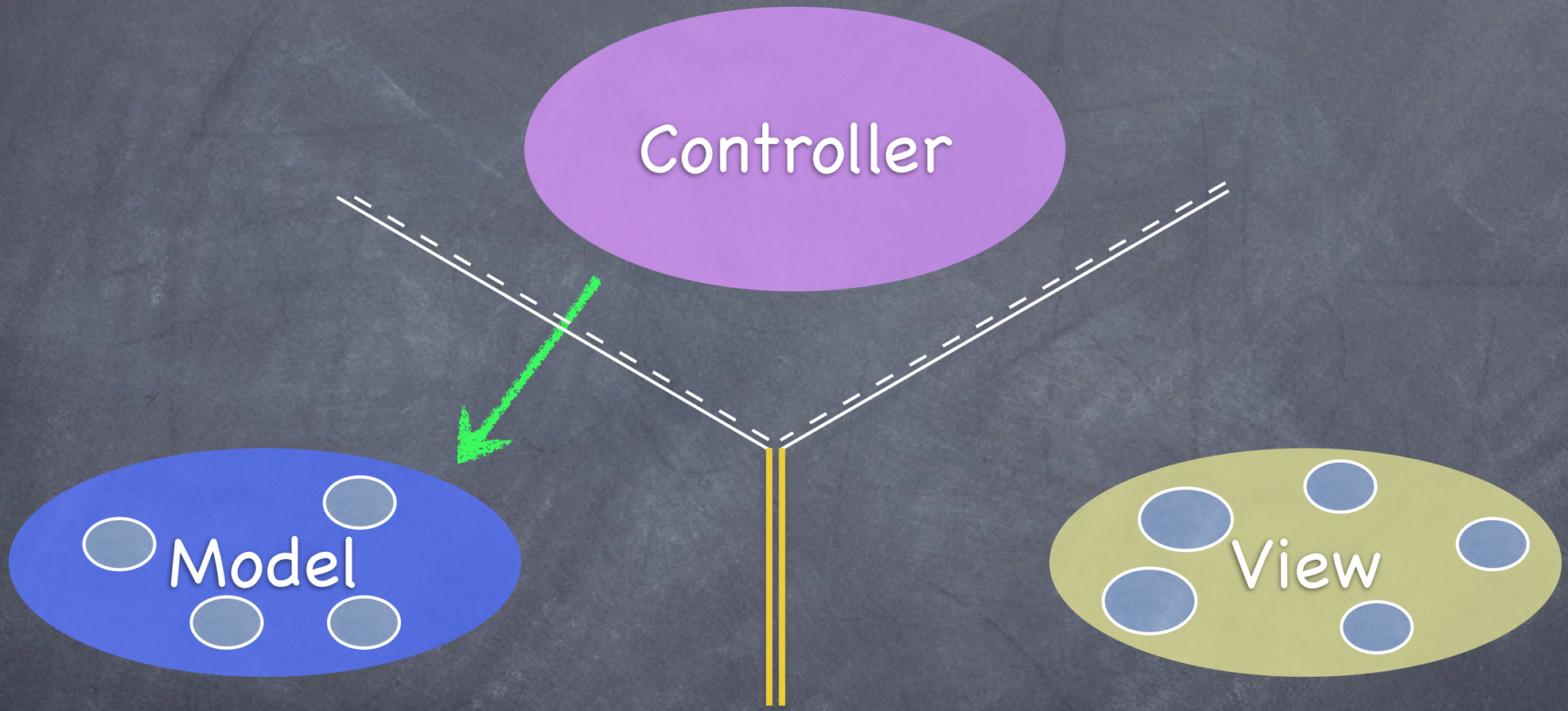
View = Your Controller's minions

MVC



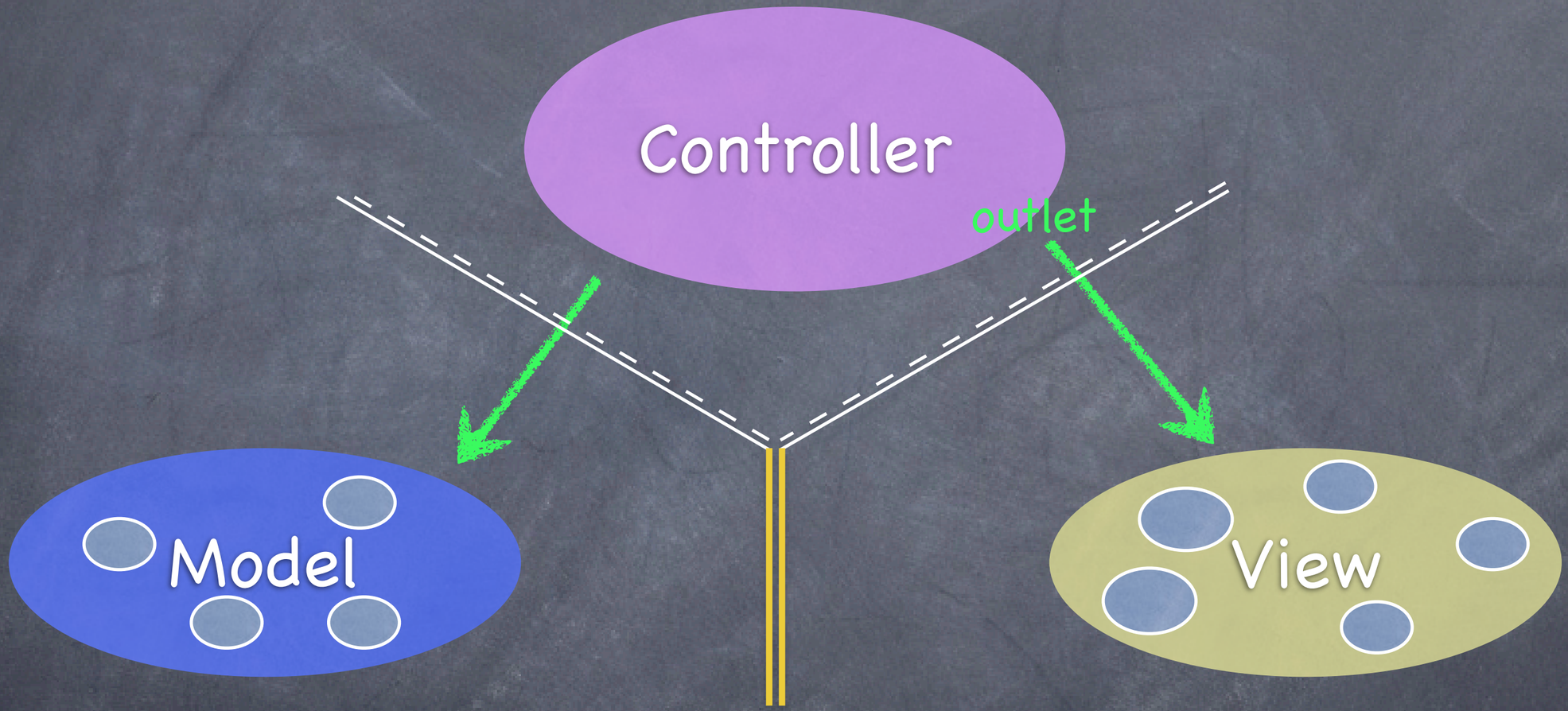
It's all about managing communication between camps

MVC



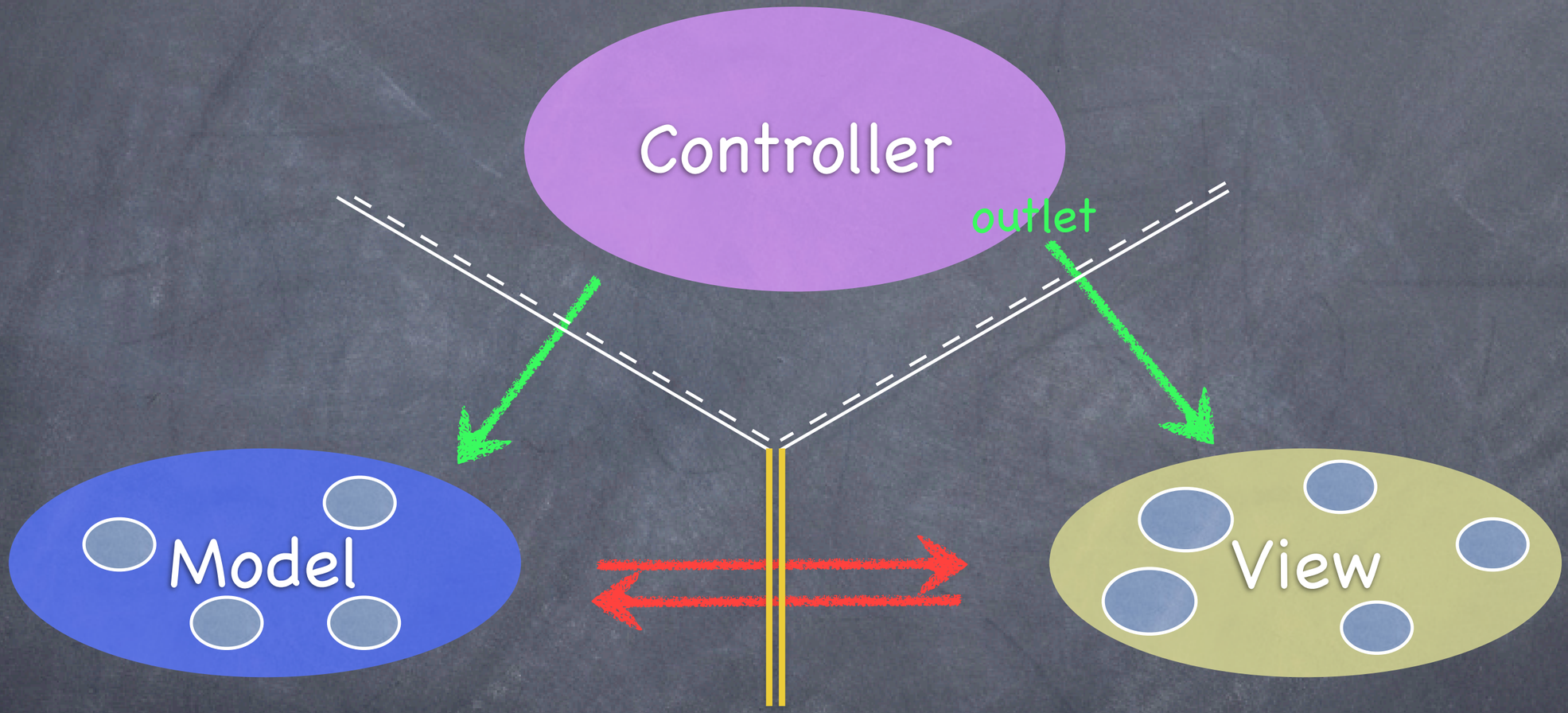
Controllers can always talk directly to their Model.

MVC



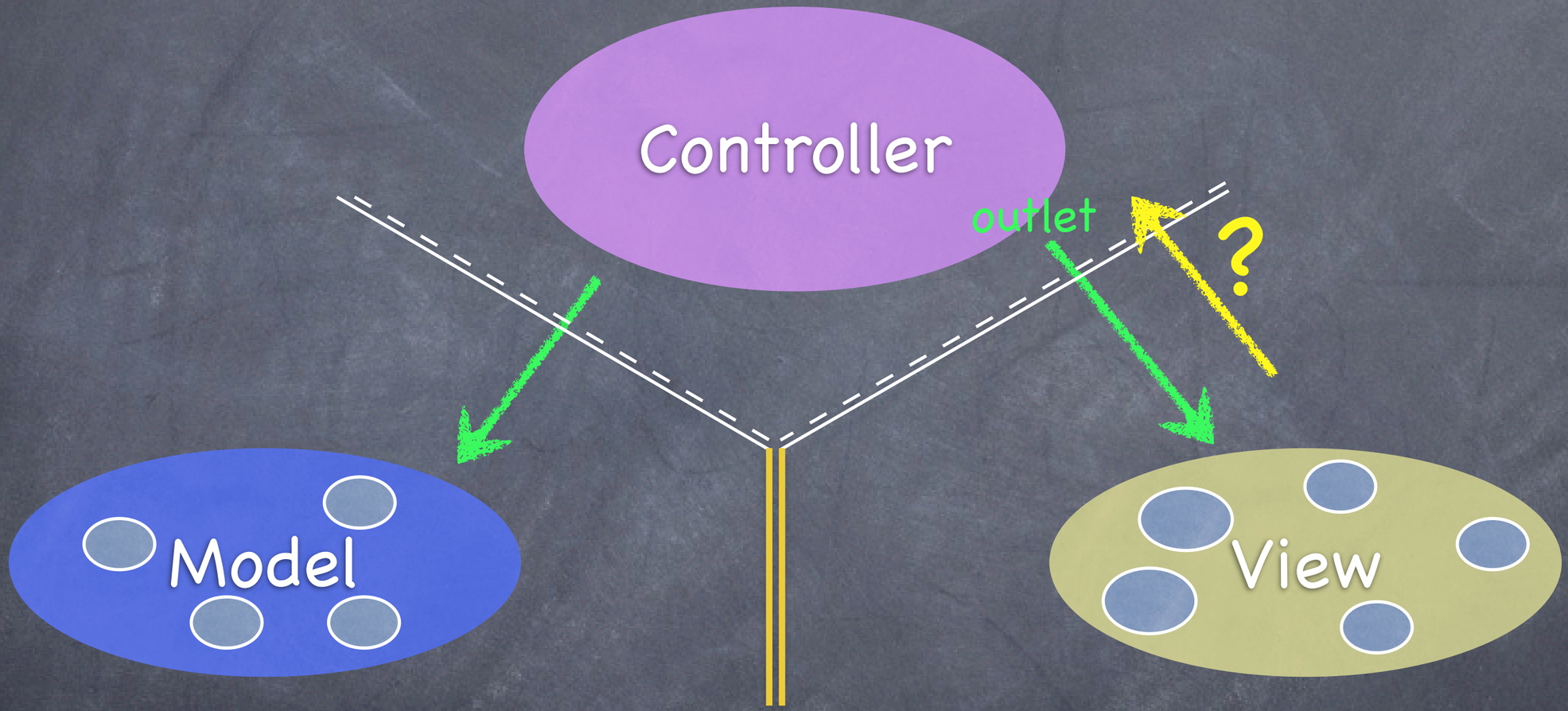
Controllers can also talk directly to their View.

MVC



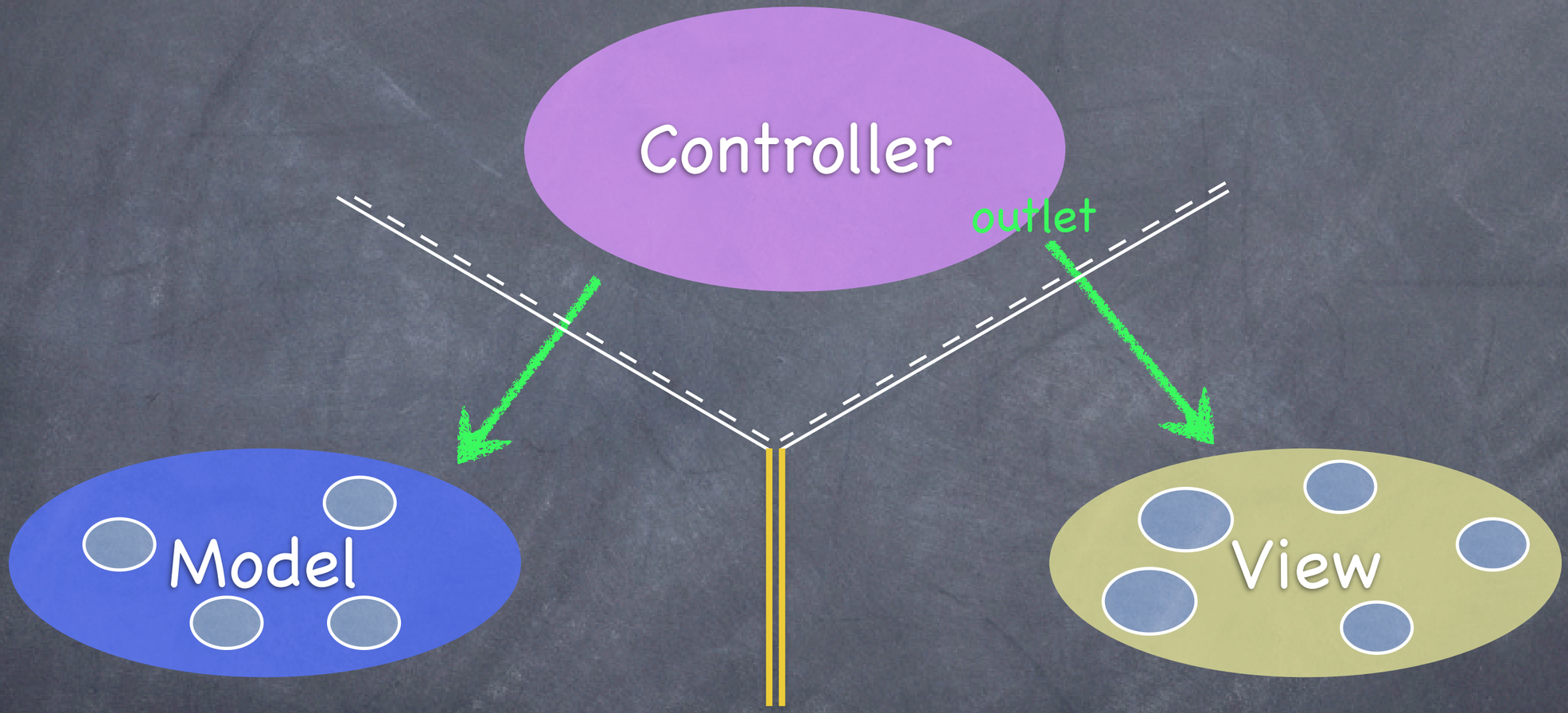
The Model and View should never speak to each other.

MVC



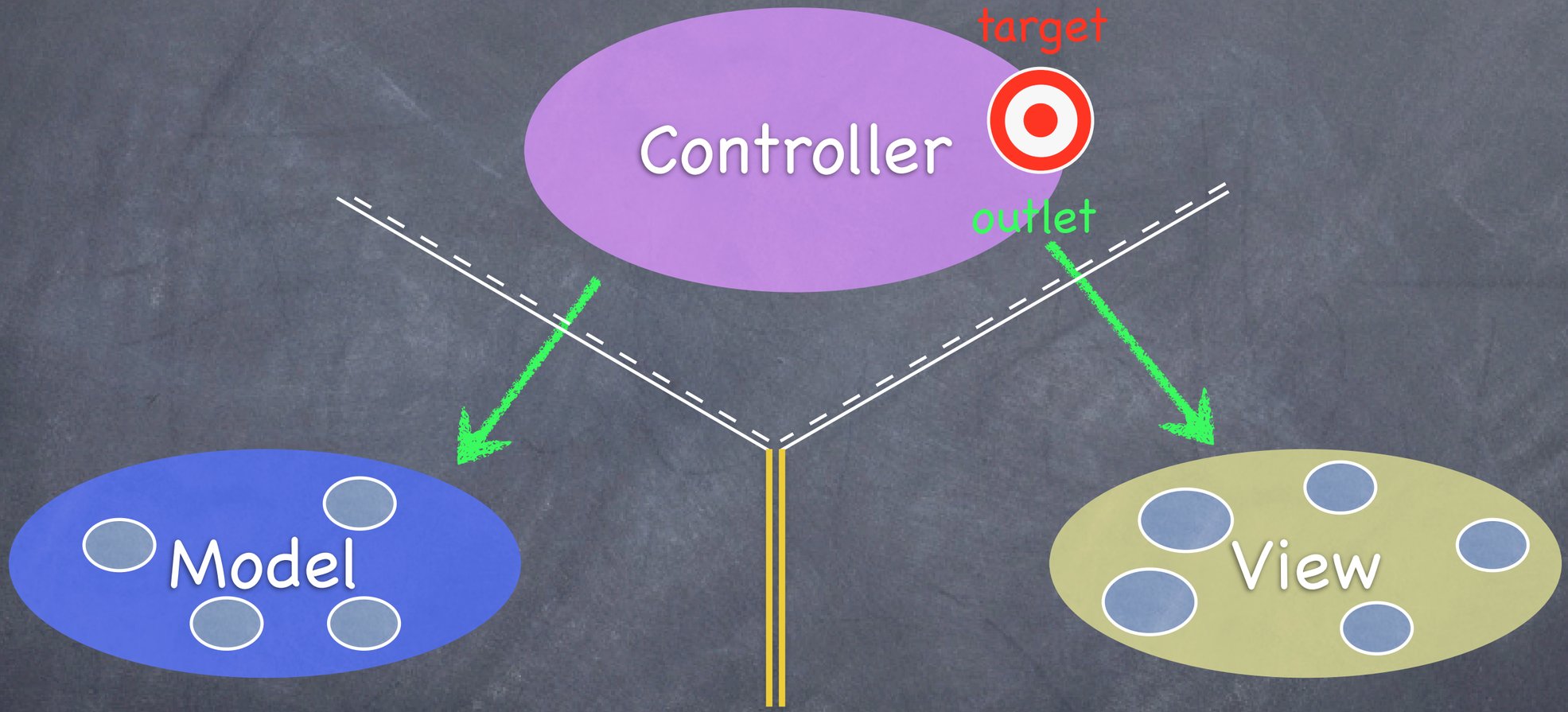
Can the View speak to its Controller?

MVC



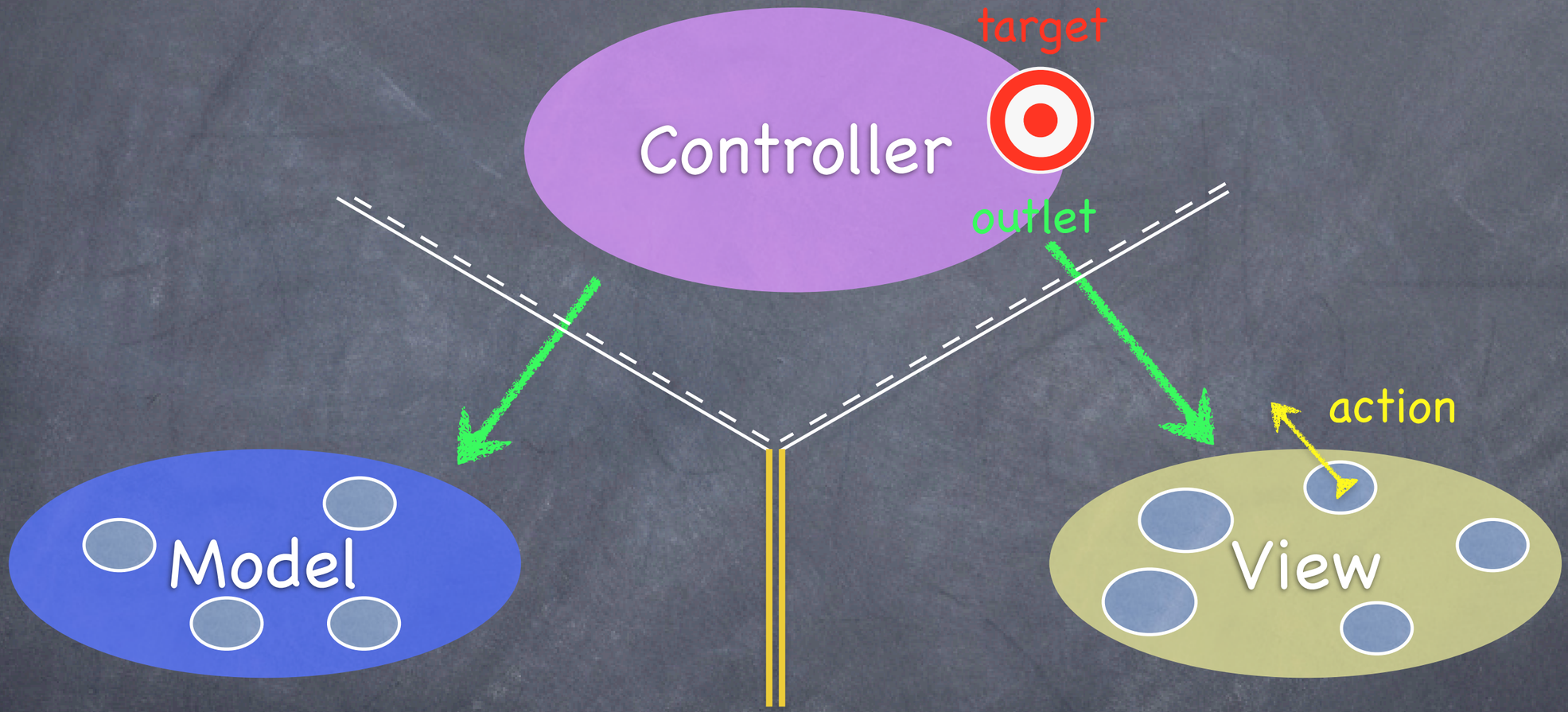
Sort of. Communication is "blind" and structured.

MVC



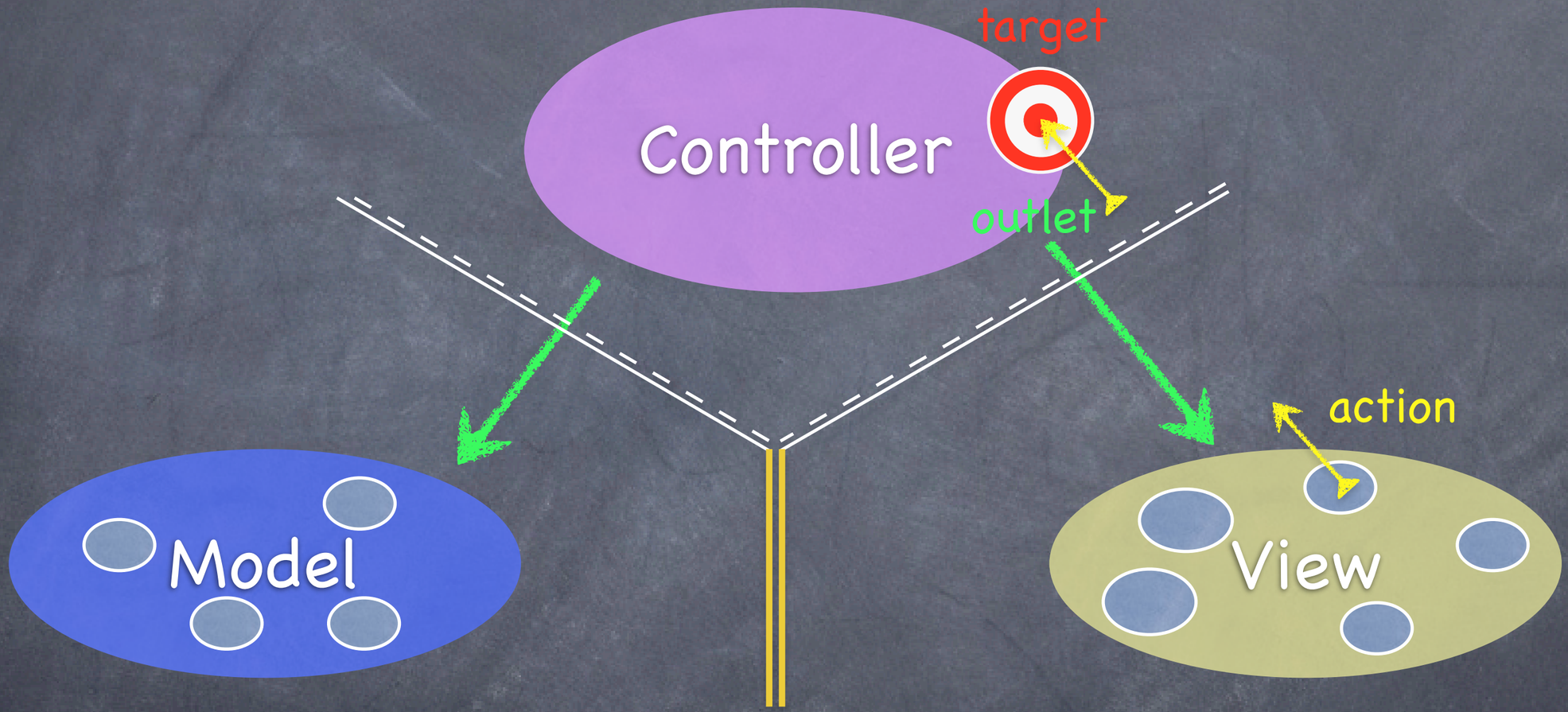
The Controller can drop a target on itself.

MVC



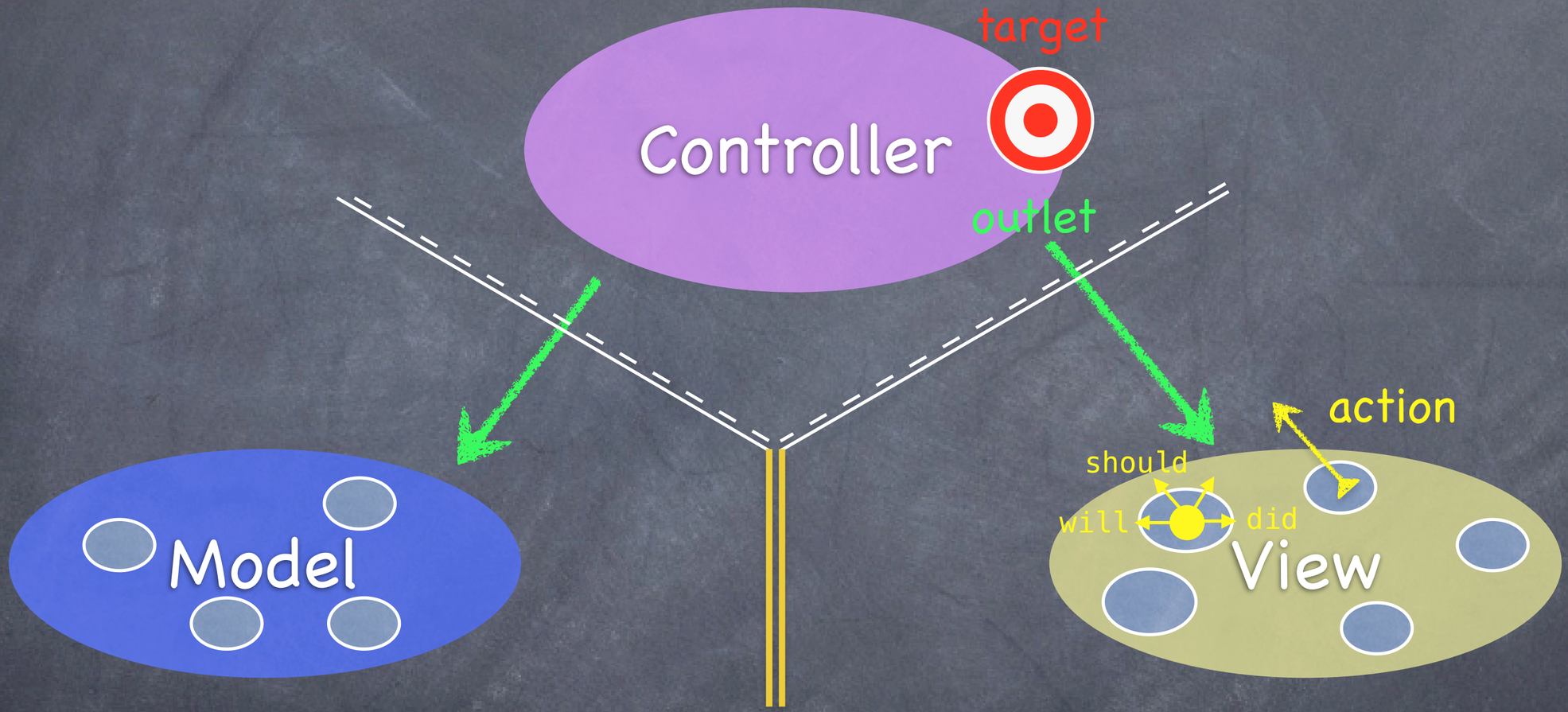
Then hand out an action to the View.

MVC



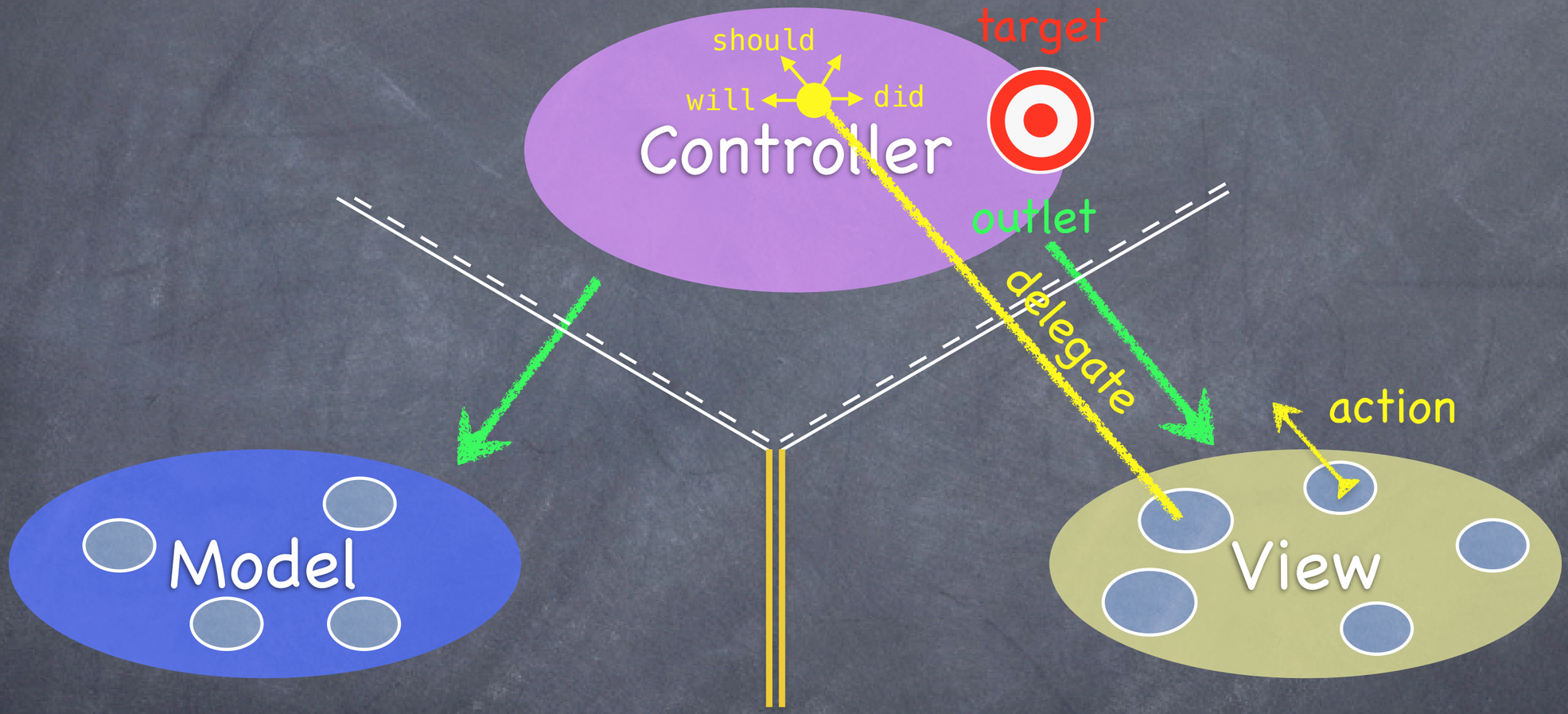
The View sends the action when things happen in the UI.

MVC



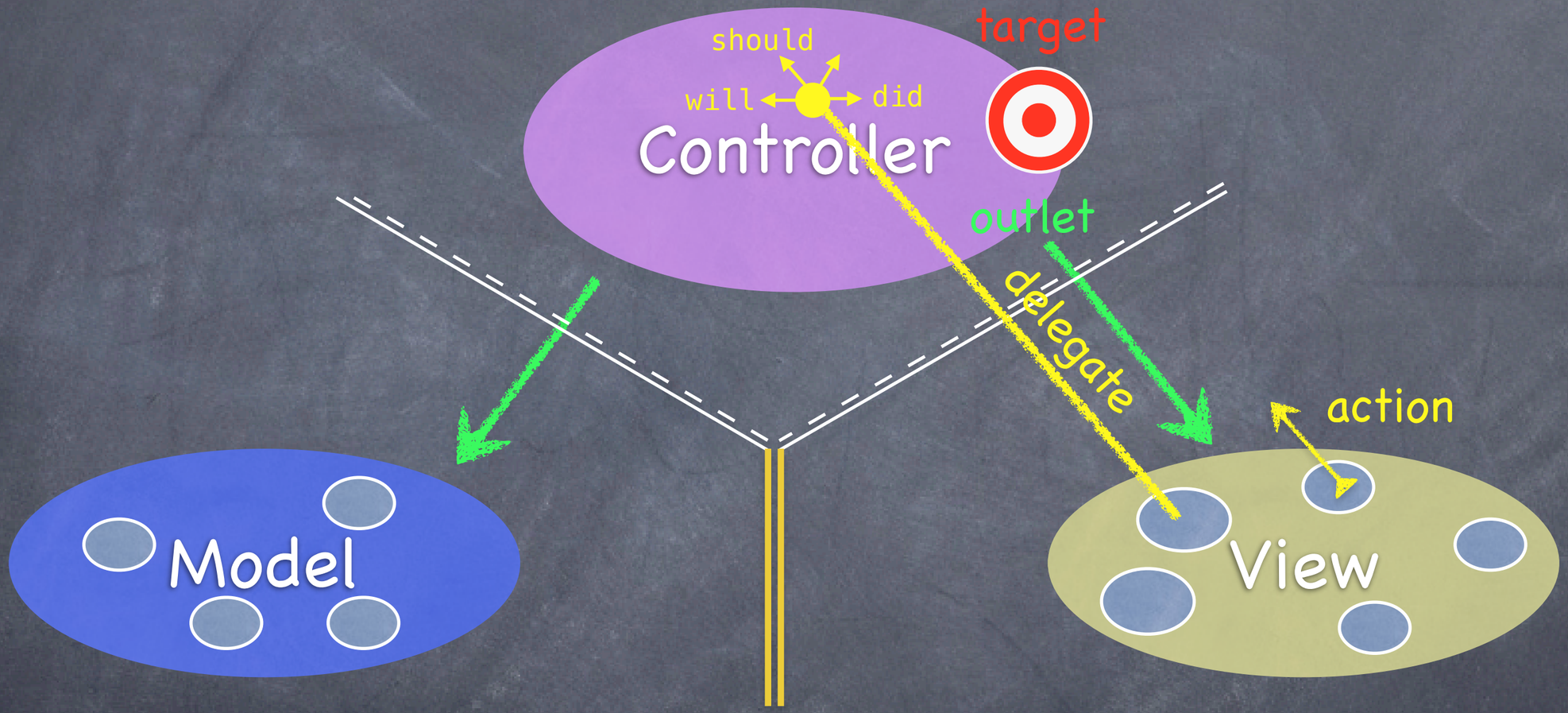
Sometimes the View needs to synchronize with the Controller.

MVC



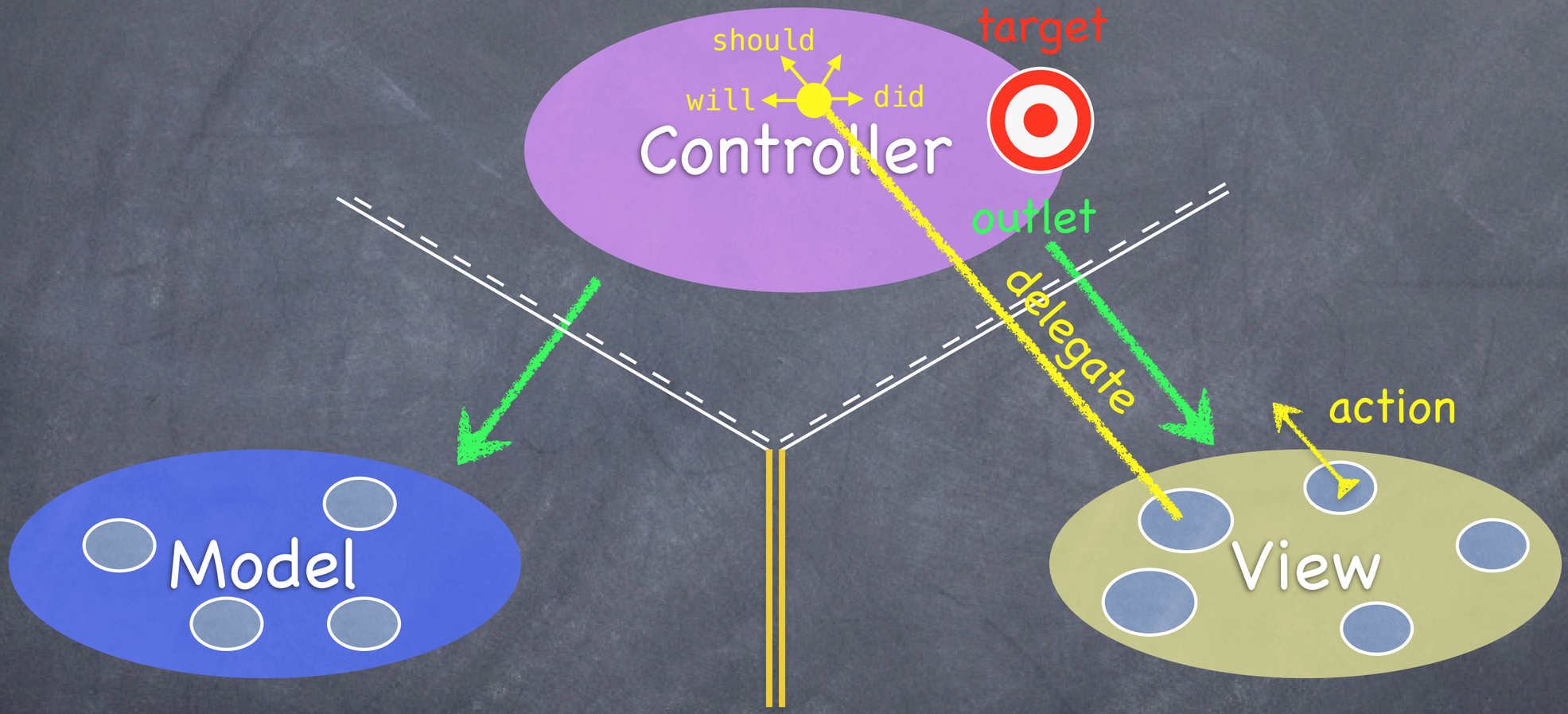
The Controller sets itself as the View's delegate.

MVC



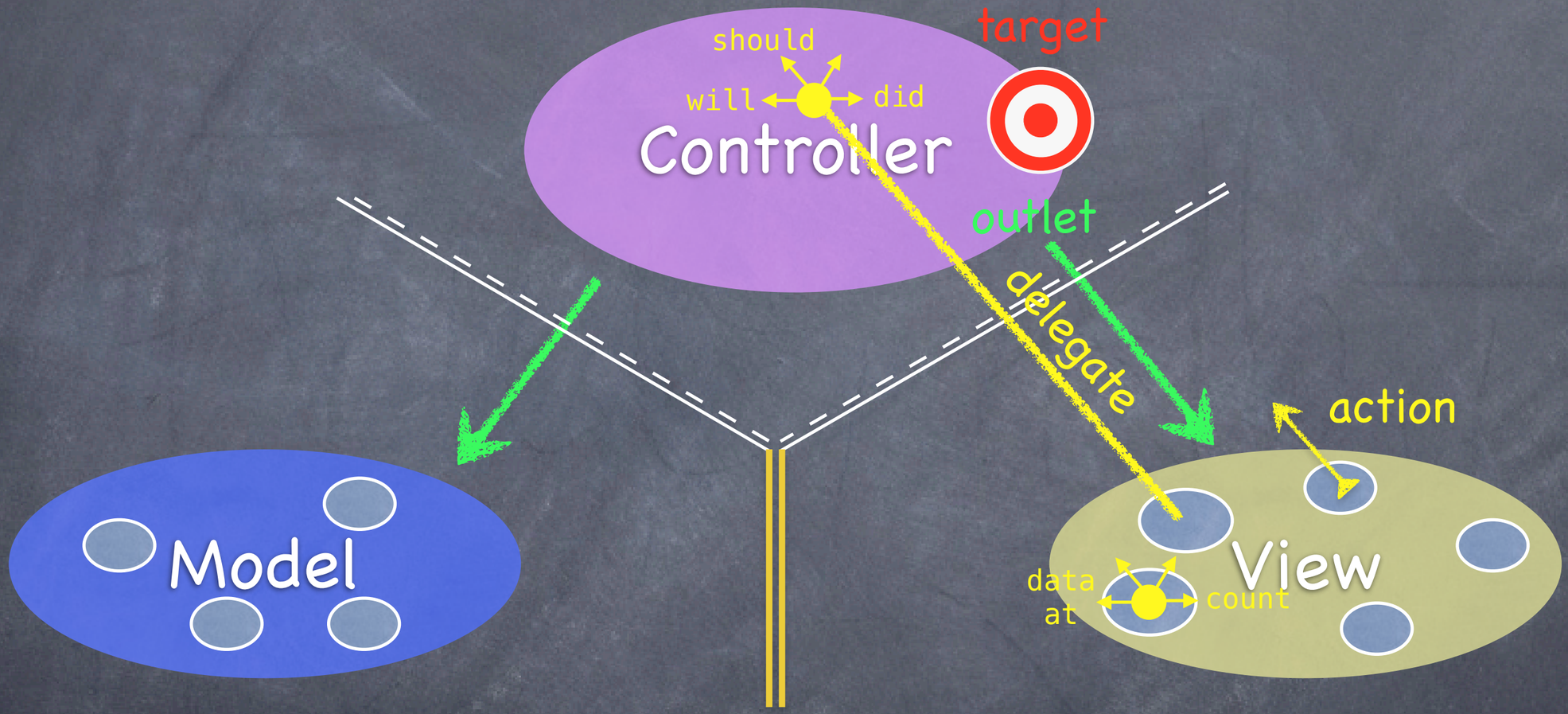
The delegate is set via a protocol (i.e. it's "blind" to class).

MVC



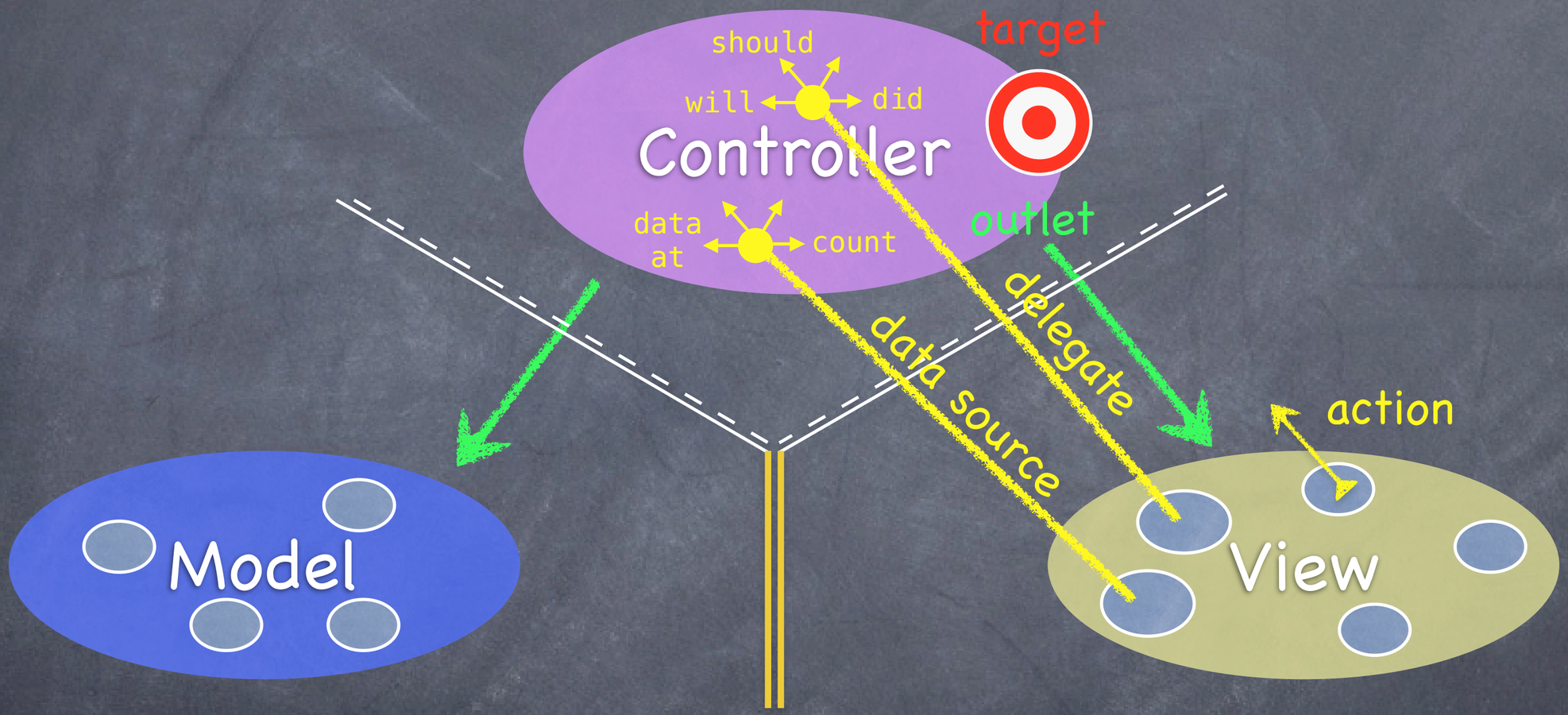
Views do not own the data they display.

MVC



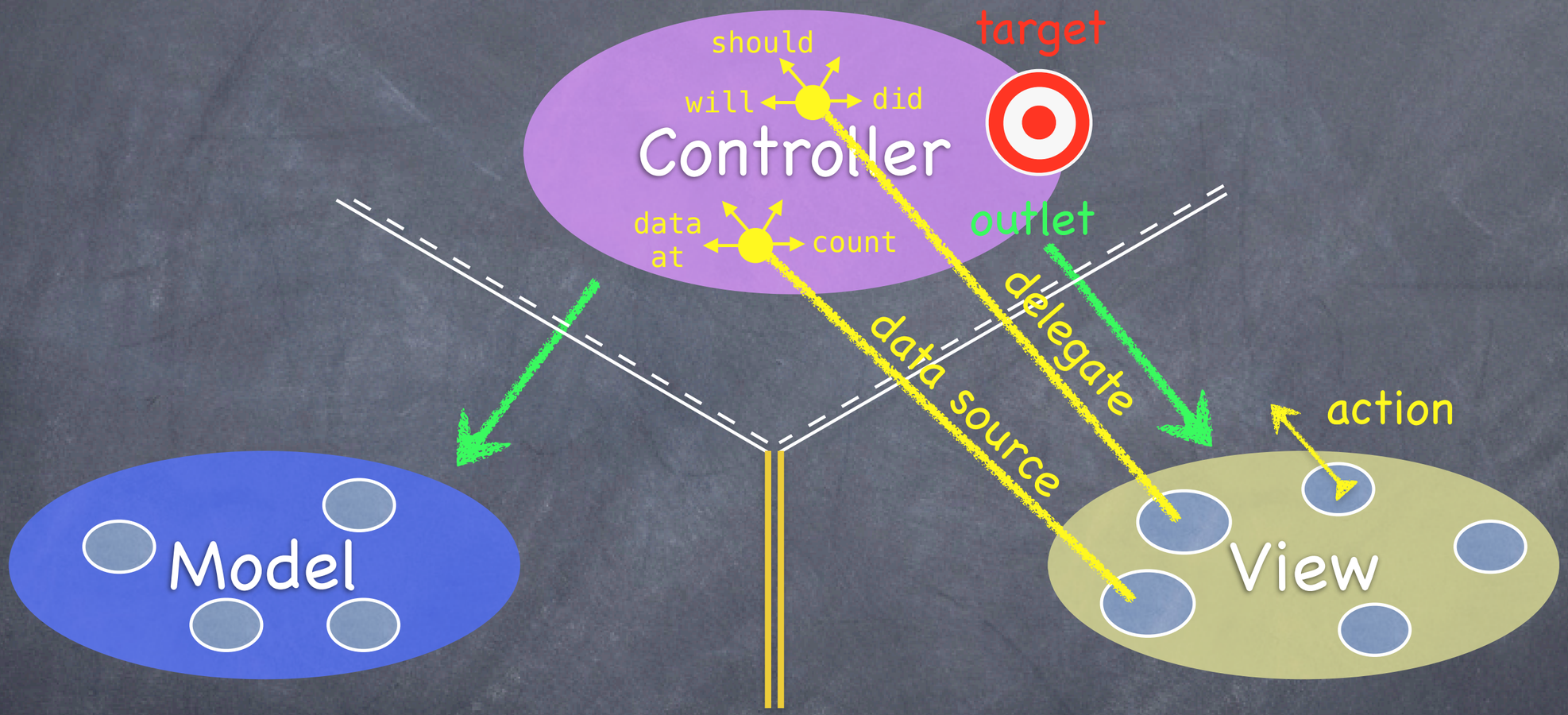
So, if needed, they have a protocol to acquire it.

MVC



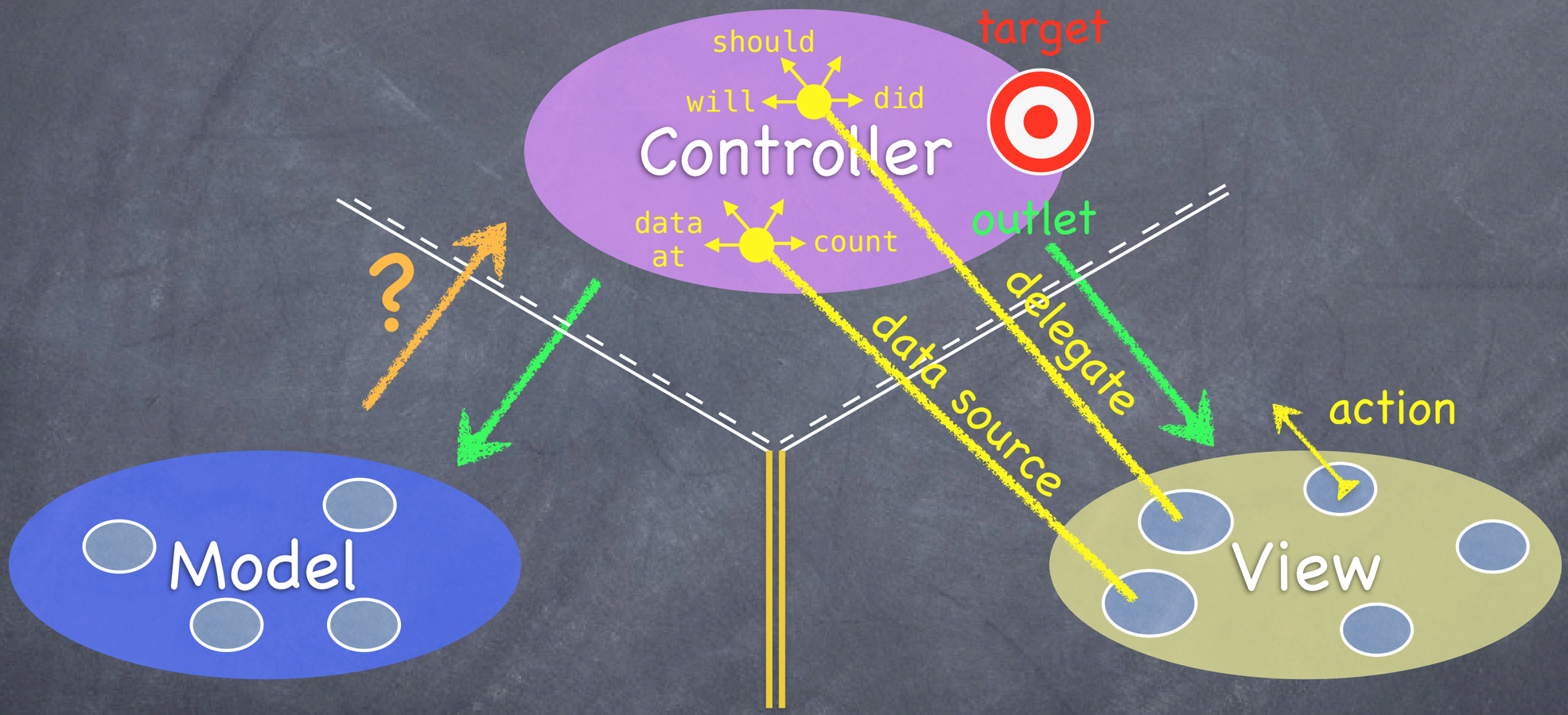
Controllers are almost always that data source (not Model!).

MVC



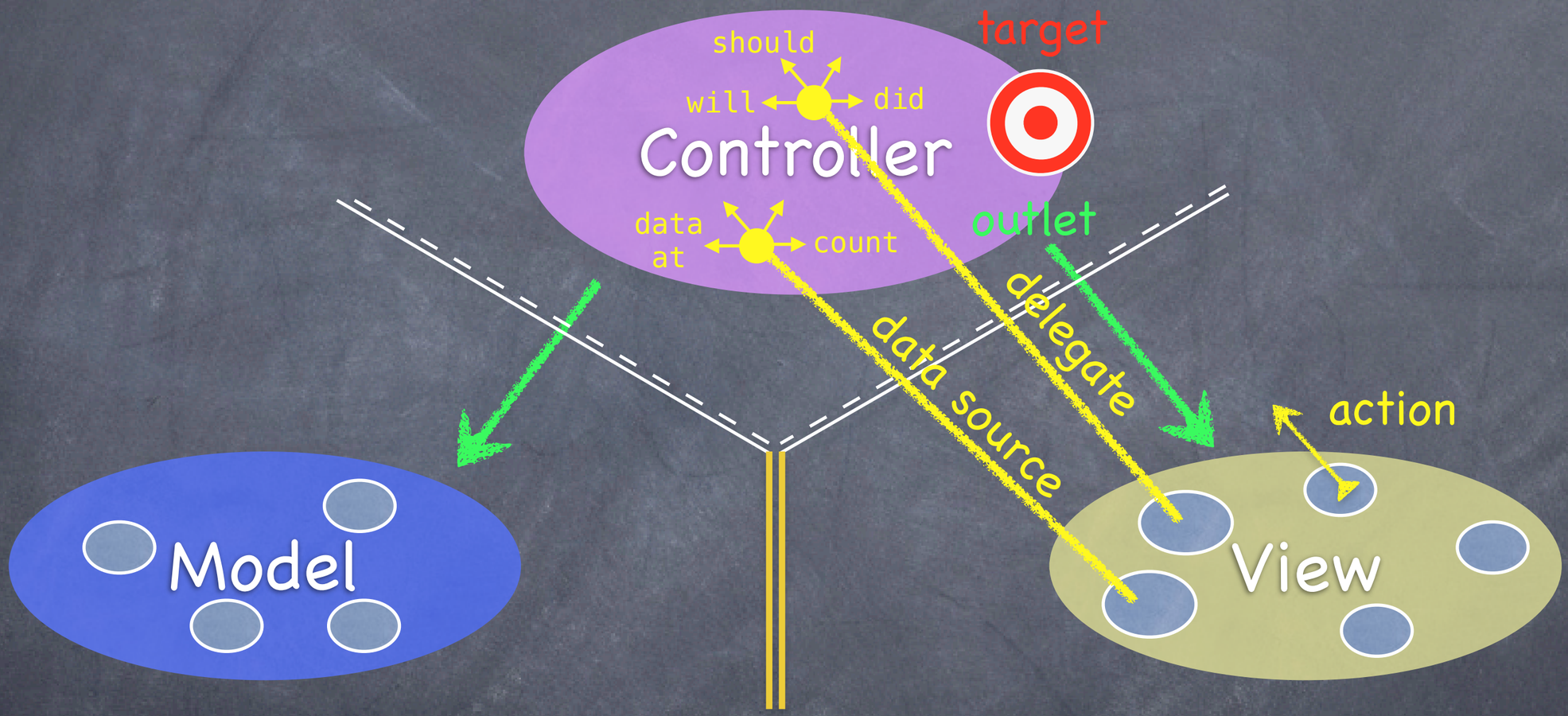
Controllers interpret/format Model information for the View.

MVC



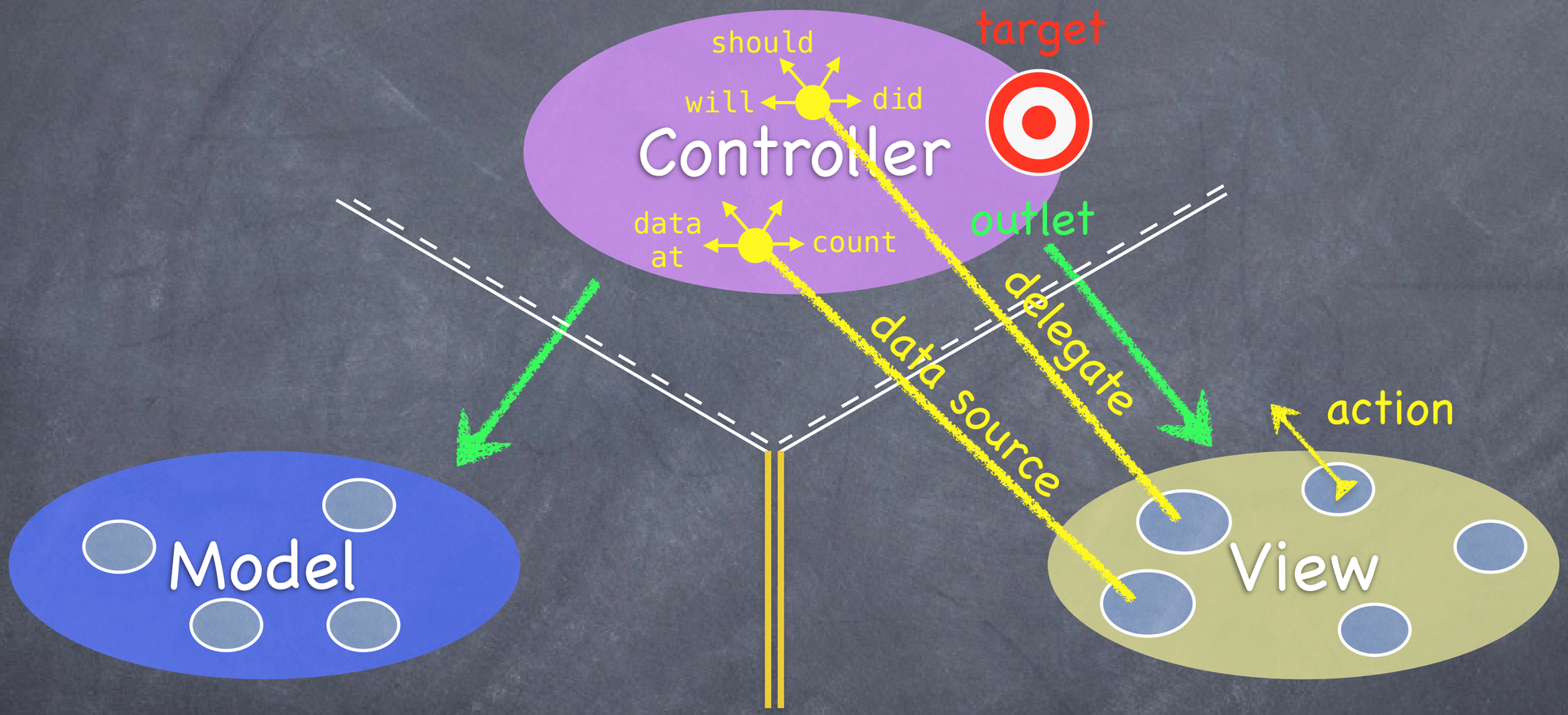
Can the Model talk directly to the Controller?

MVC



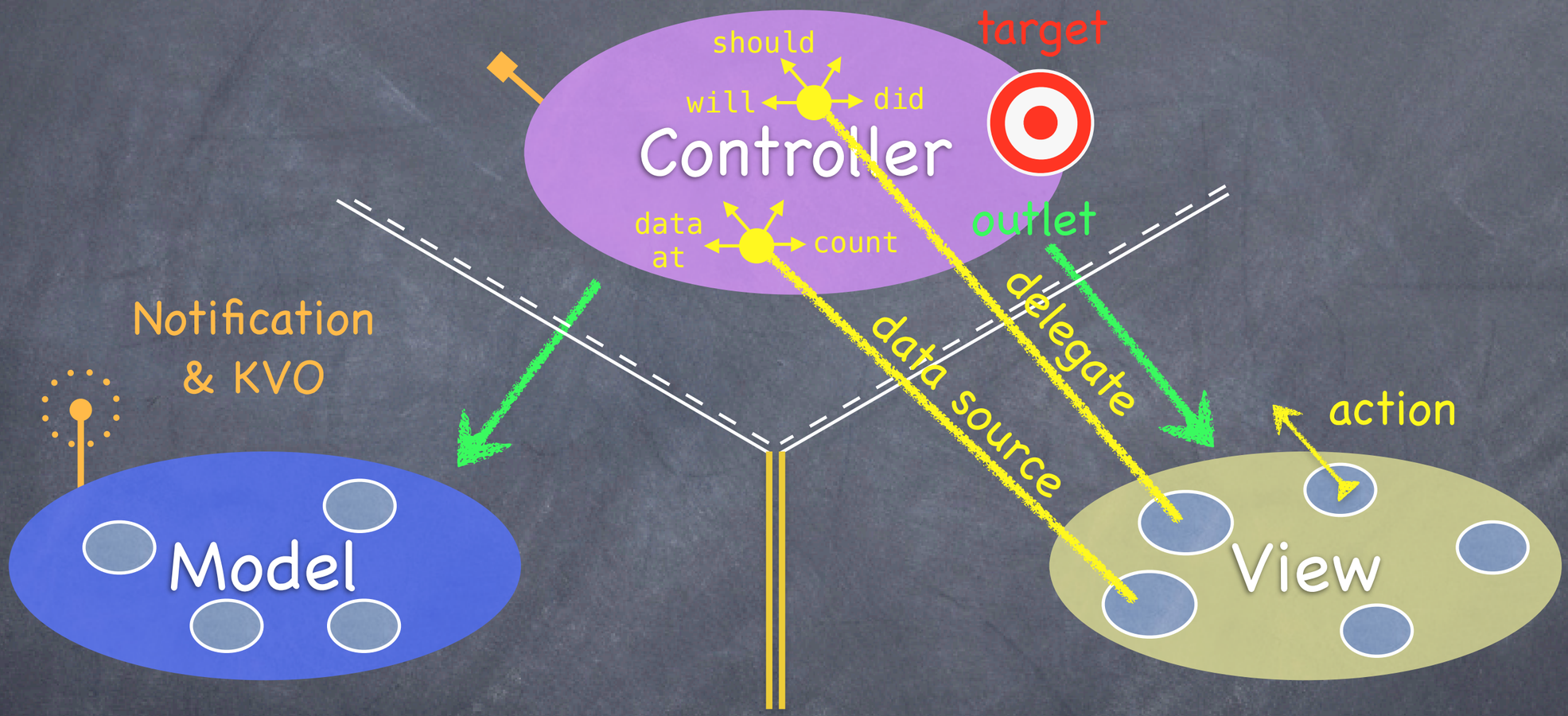
No. The Model is (should be) UI independent.

MVC



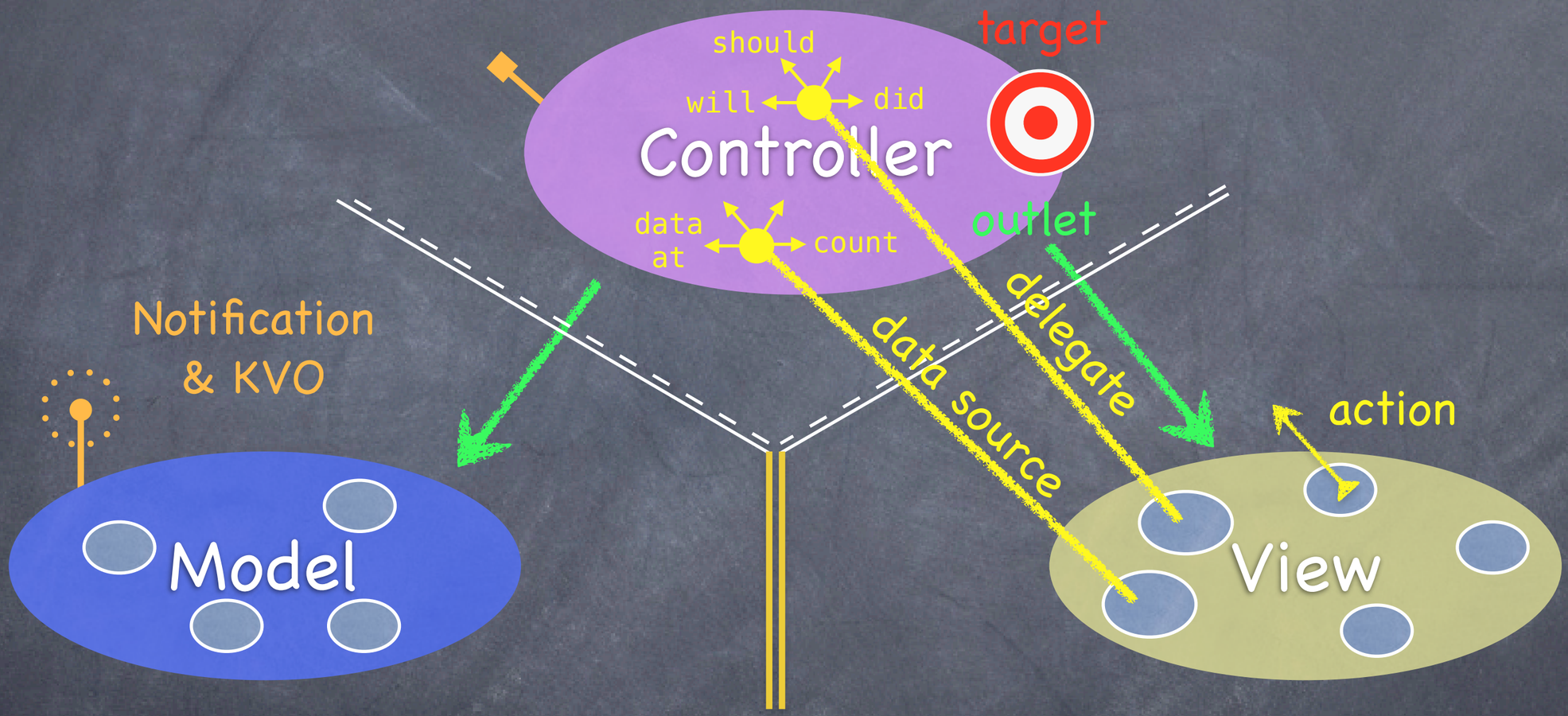
So what if the Model has information to update or something?

MVC



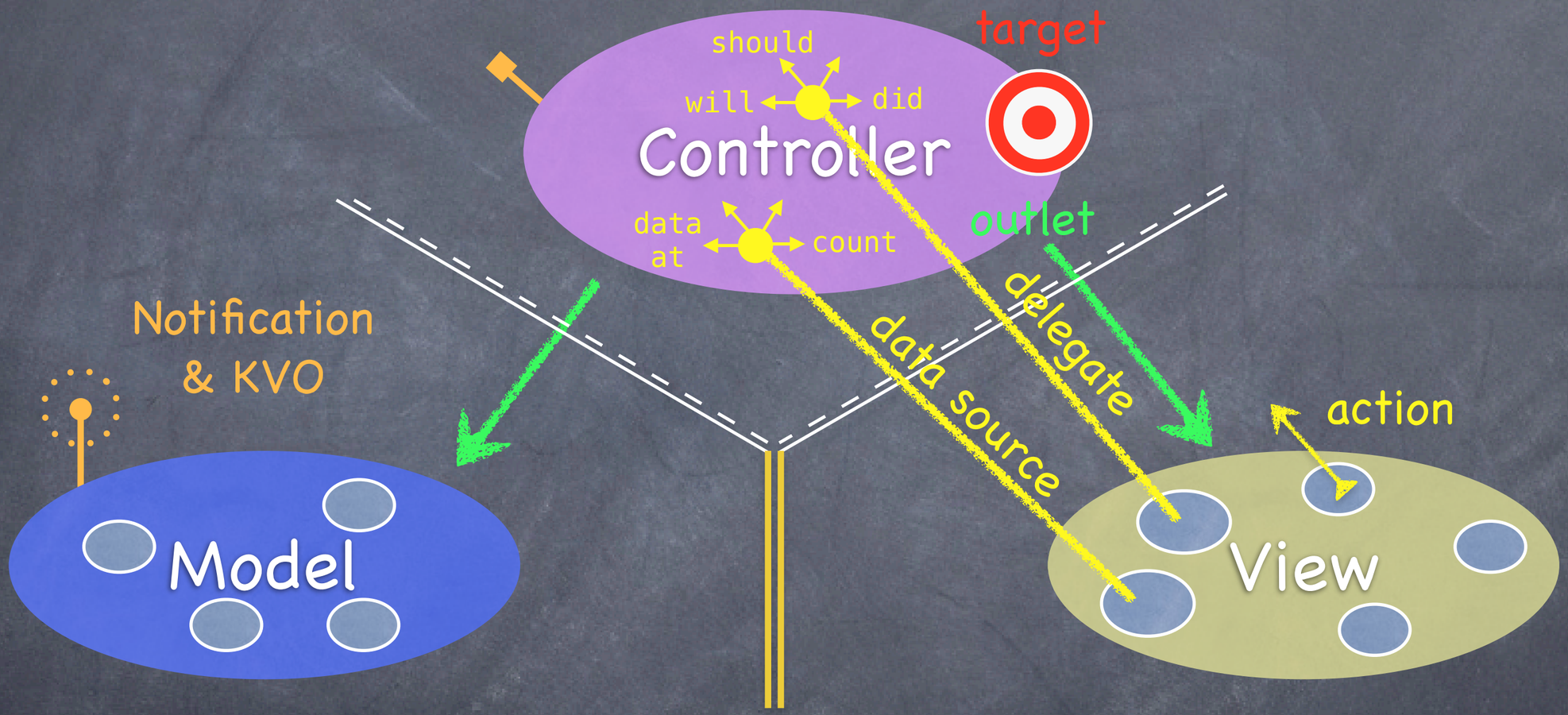
It uses a "radio station"-like broadcast mechanism.

MVC



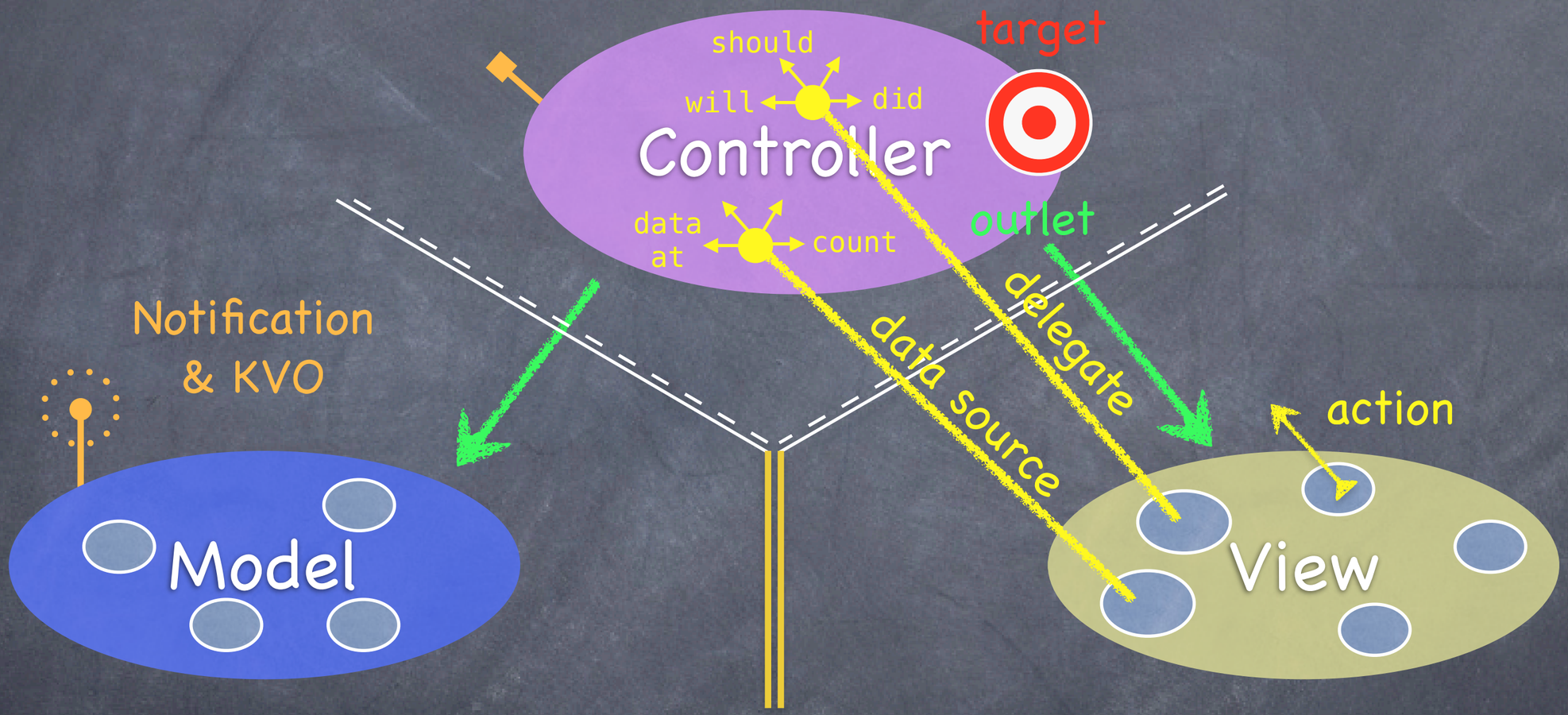
Controllers (or other Model) "tune in" to interesting stuff.

MVC



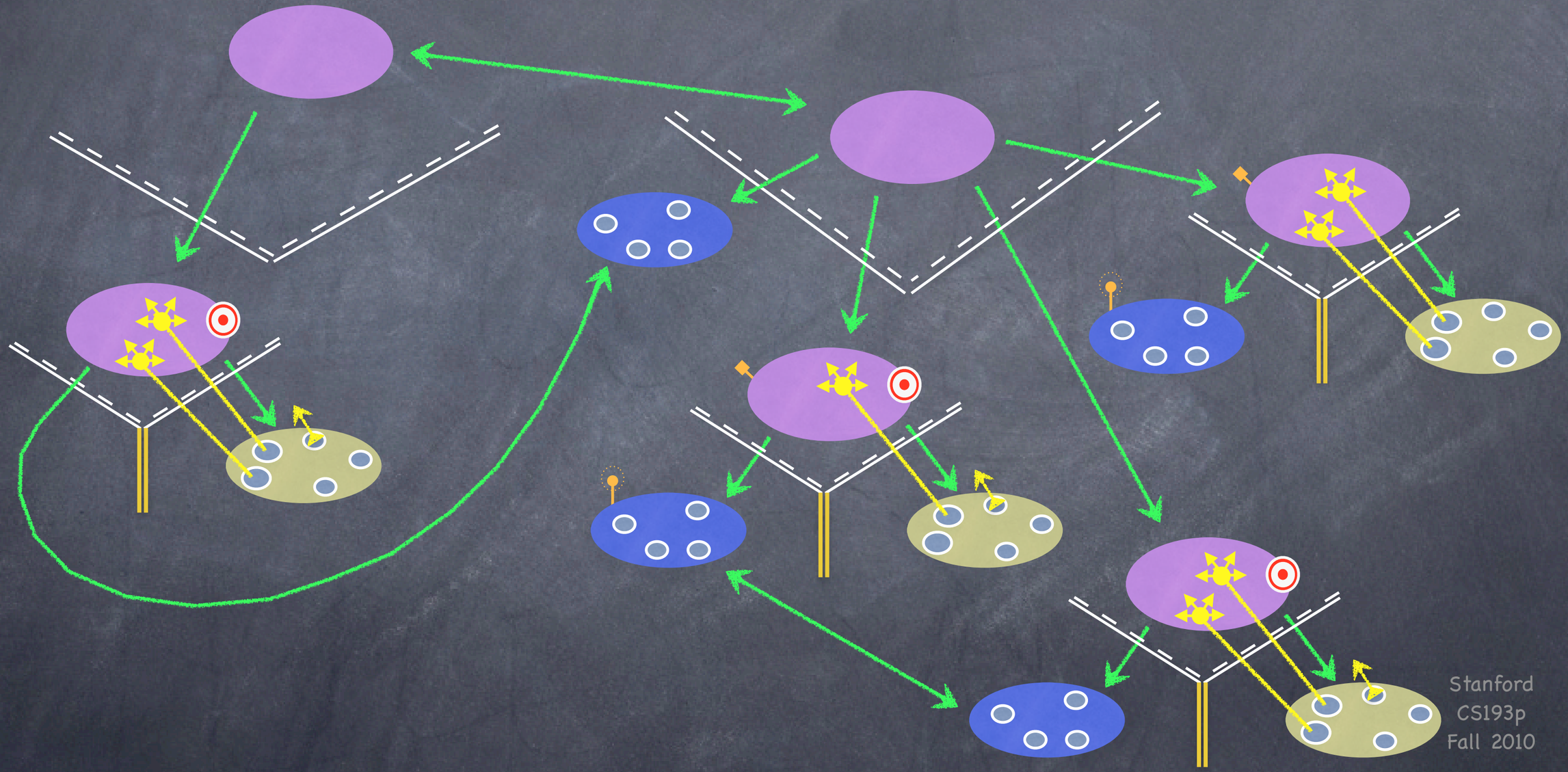
A View might "tune in," but probably not to a Model's "station."

MVC

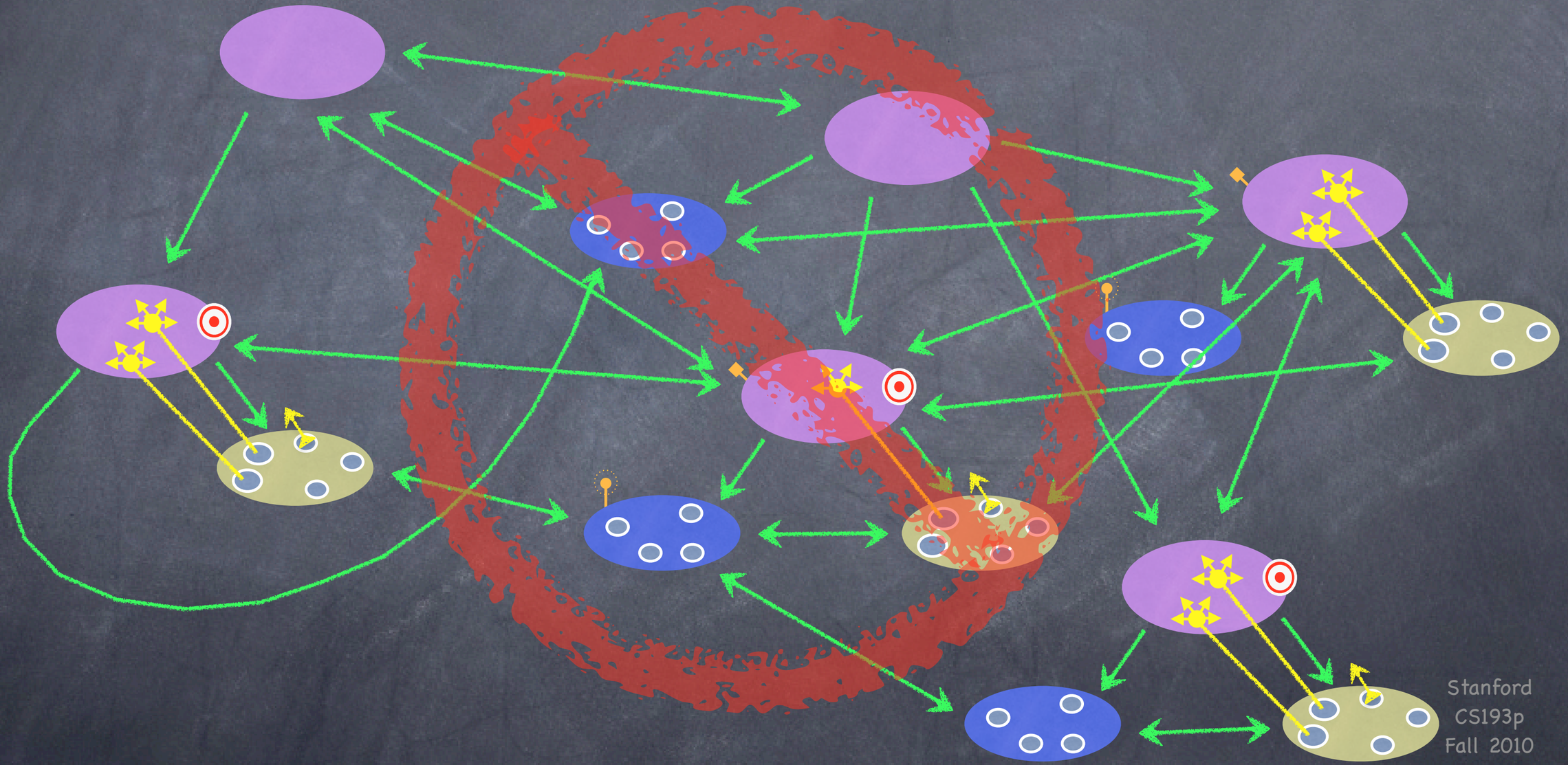


Now combine MVC groups to make complicated programs ...

MVCs working together



MVCs not working together



Coming Up

- Next Lecture
  - Overview of the Development Environment
  - Xcode
  - Objective-C intro
  - Interface Builder
  - Concrete example of MVC
  - Major demonstration of all of the above: Calculator
- Friday (optional)
  - Installing the SDK and using the debugger (this is the only time that will be covered)
- Next Week
  - Objective-C language in depth
  - Foundation classes: arrays, dictionaries, strings, etc.
  - Dynamic and static typing
  - Memory Management