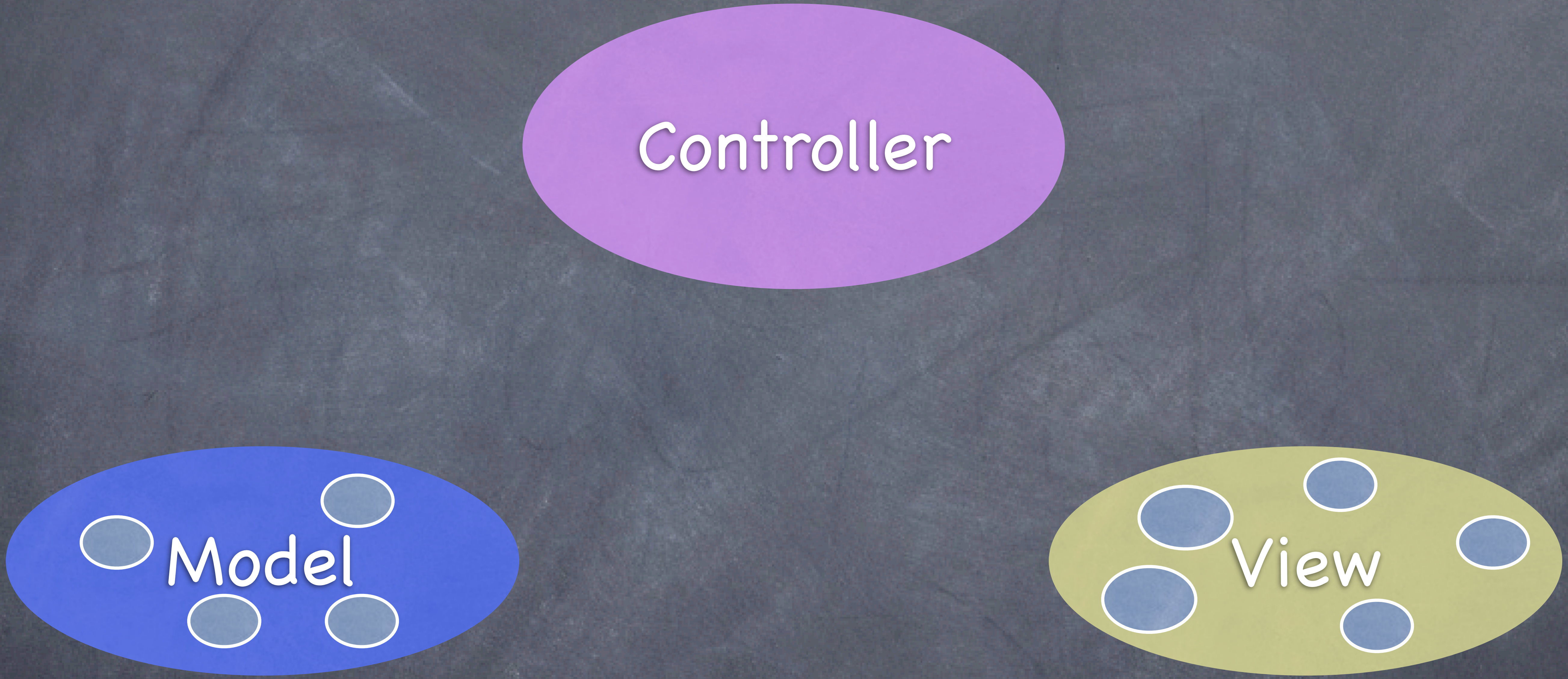
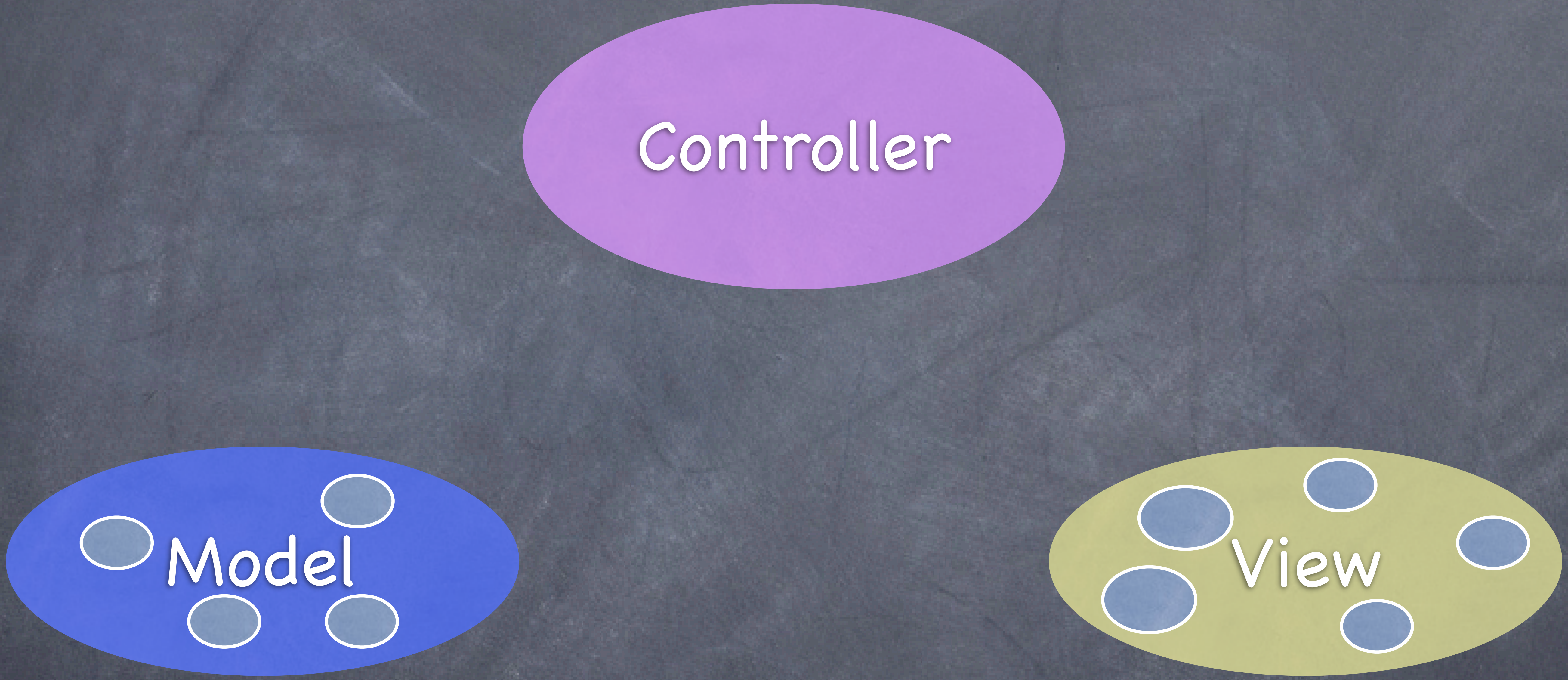
MVC



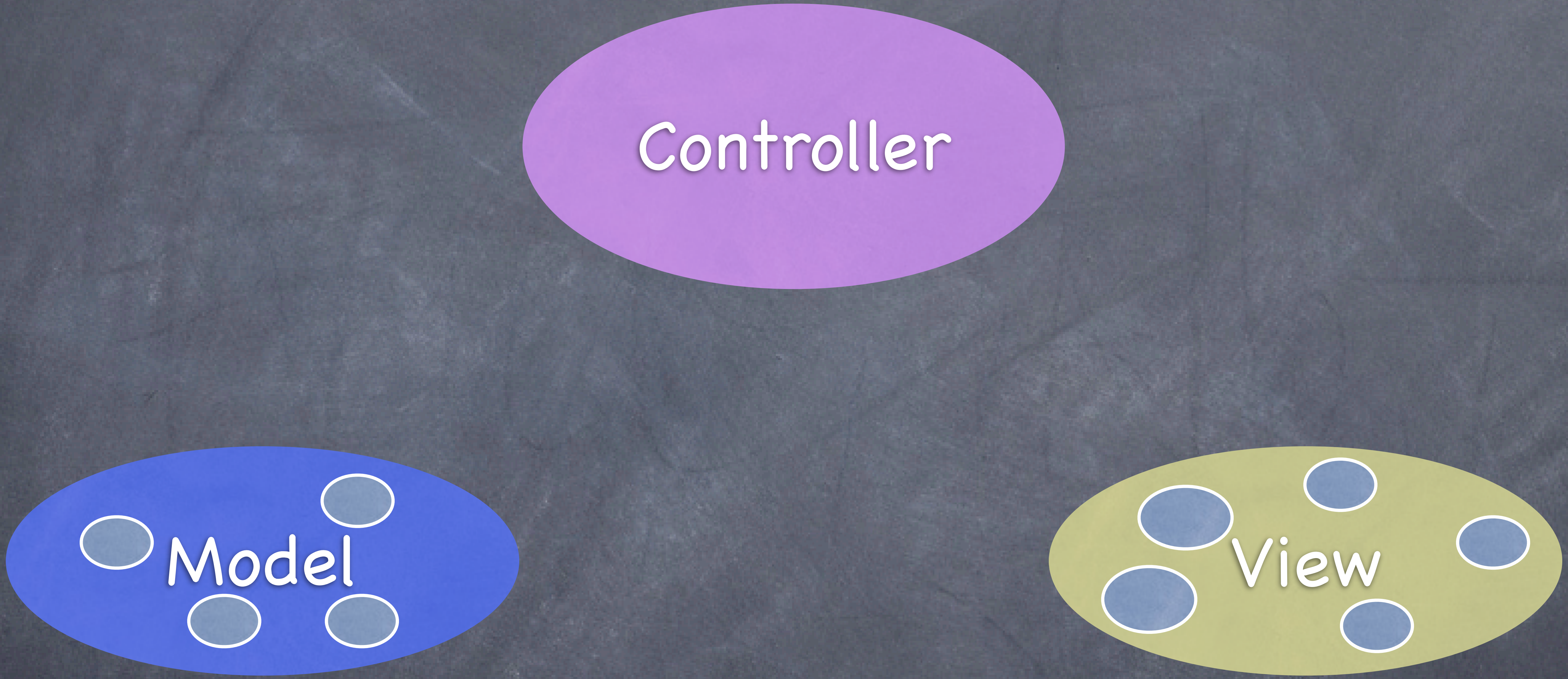
Divide objects in your program into 3 "camps."

MVC



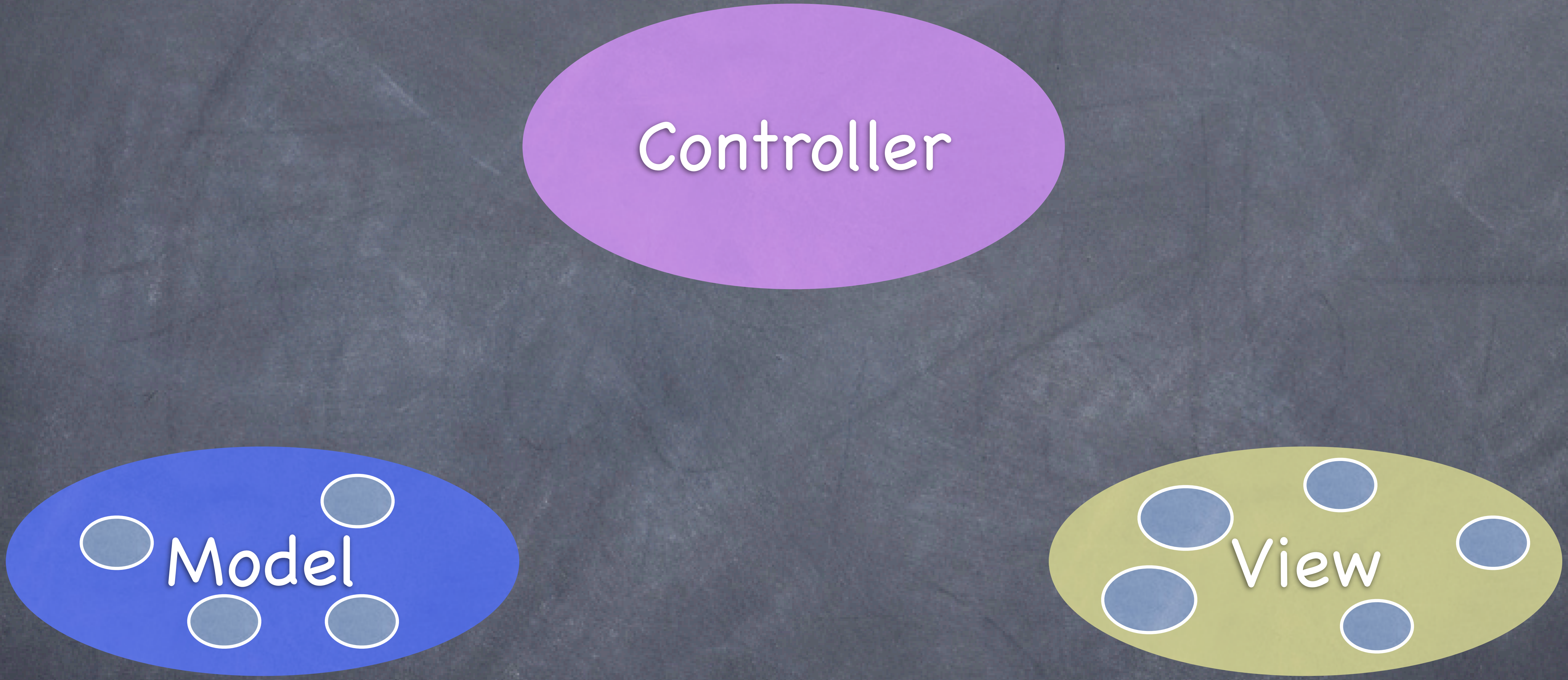
Model = What your application is (but not how it is displayed)

MVC



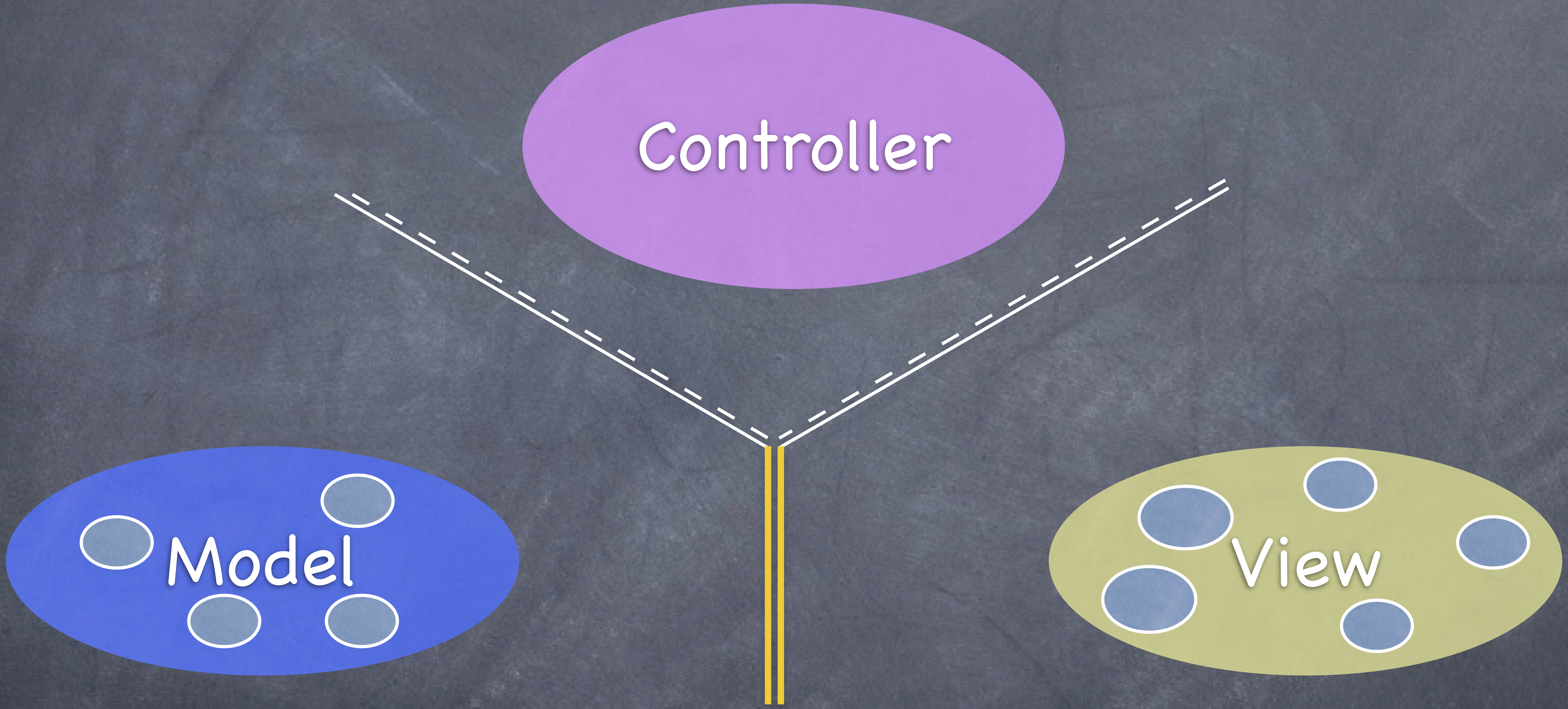
Controller = How your Model is presented to the user (UI logic)

MVC



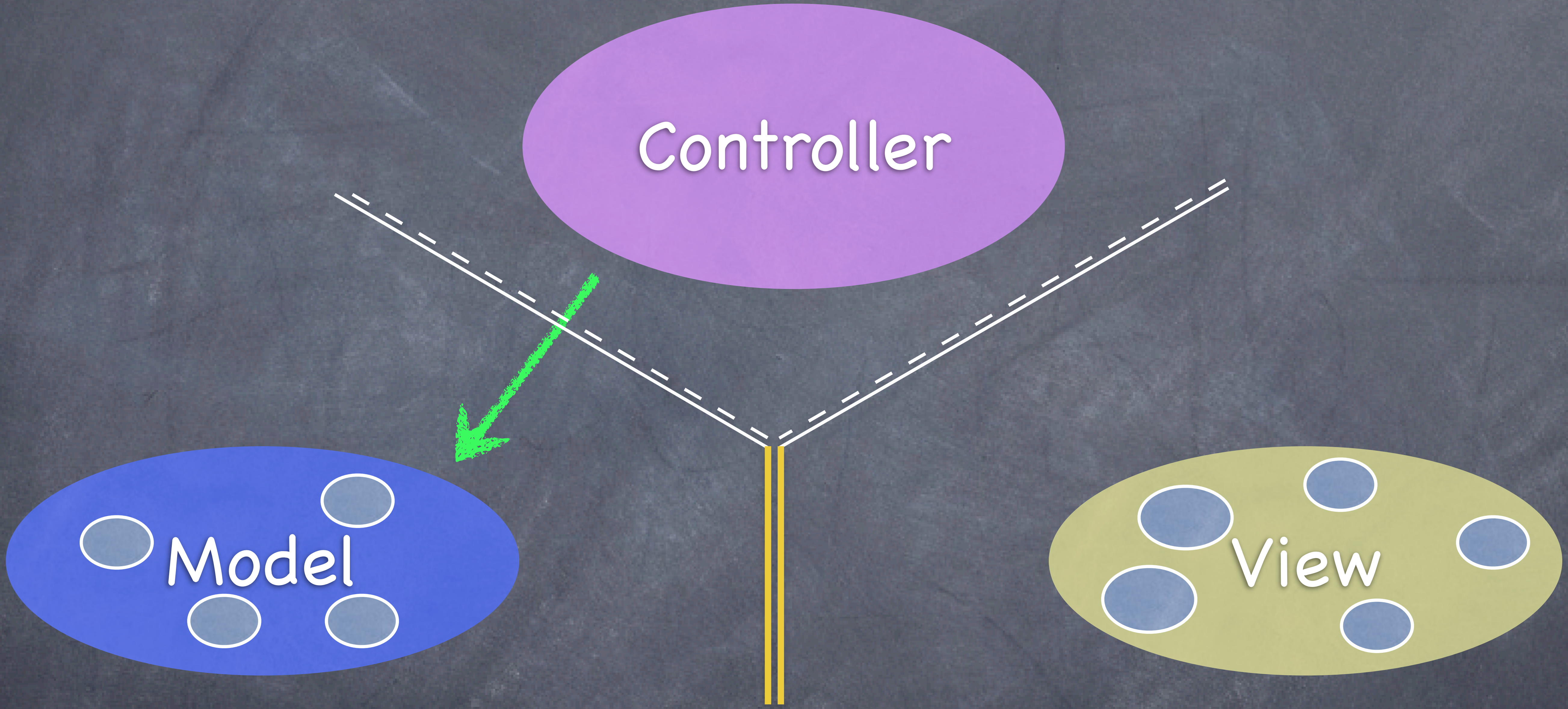
View = Your Controller's minions

MVC



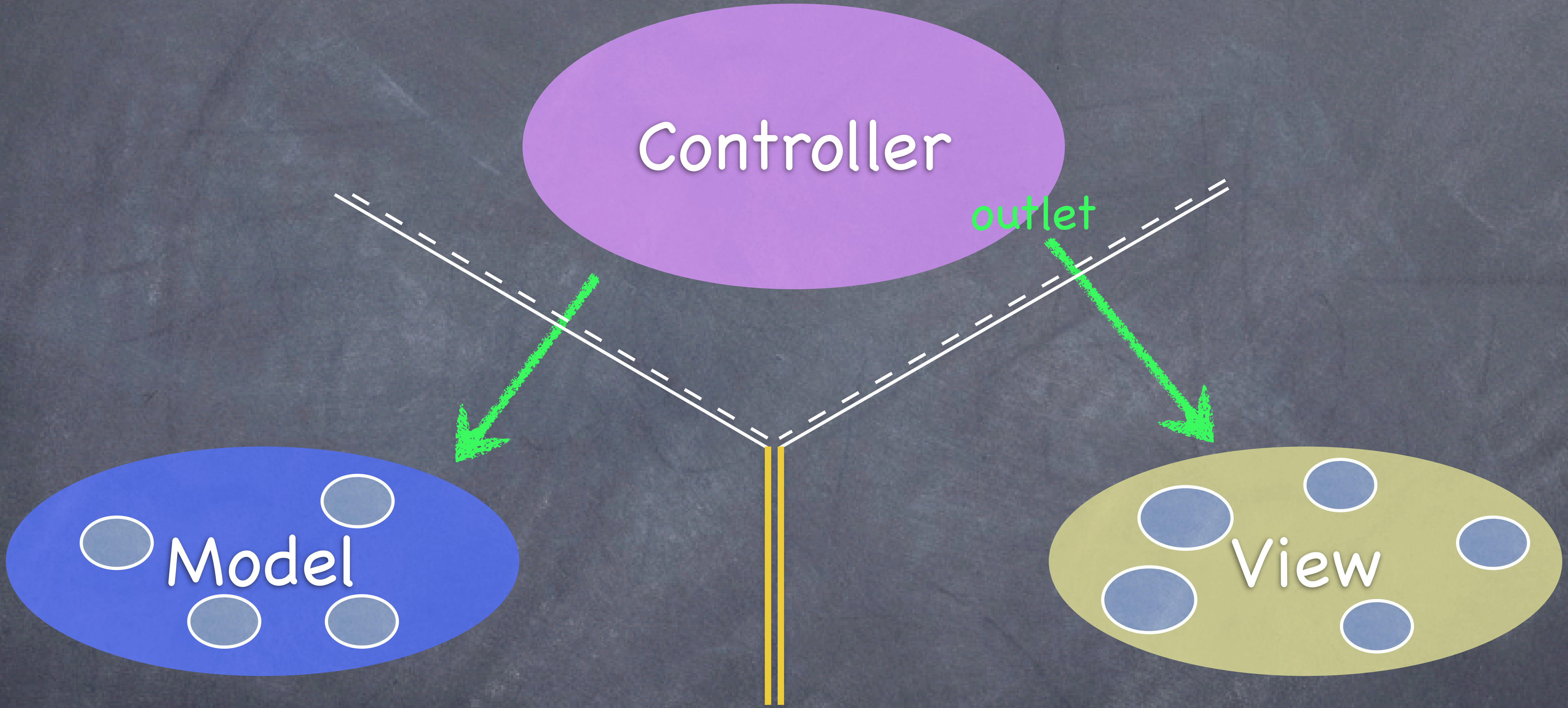
It's all about managing communication between camps

MVC



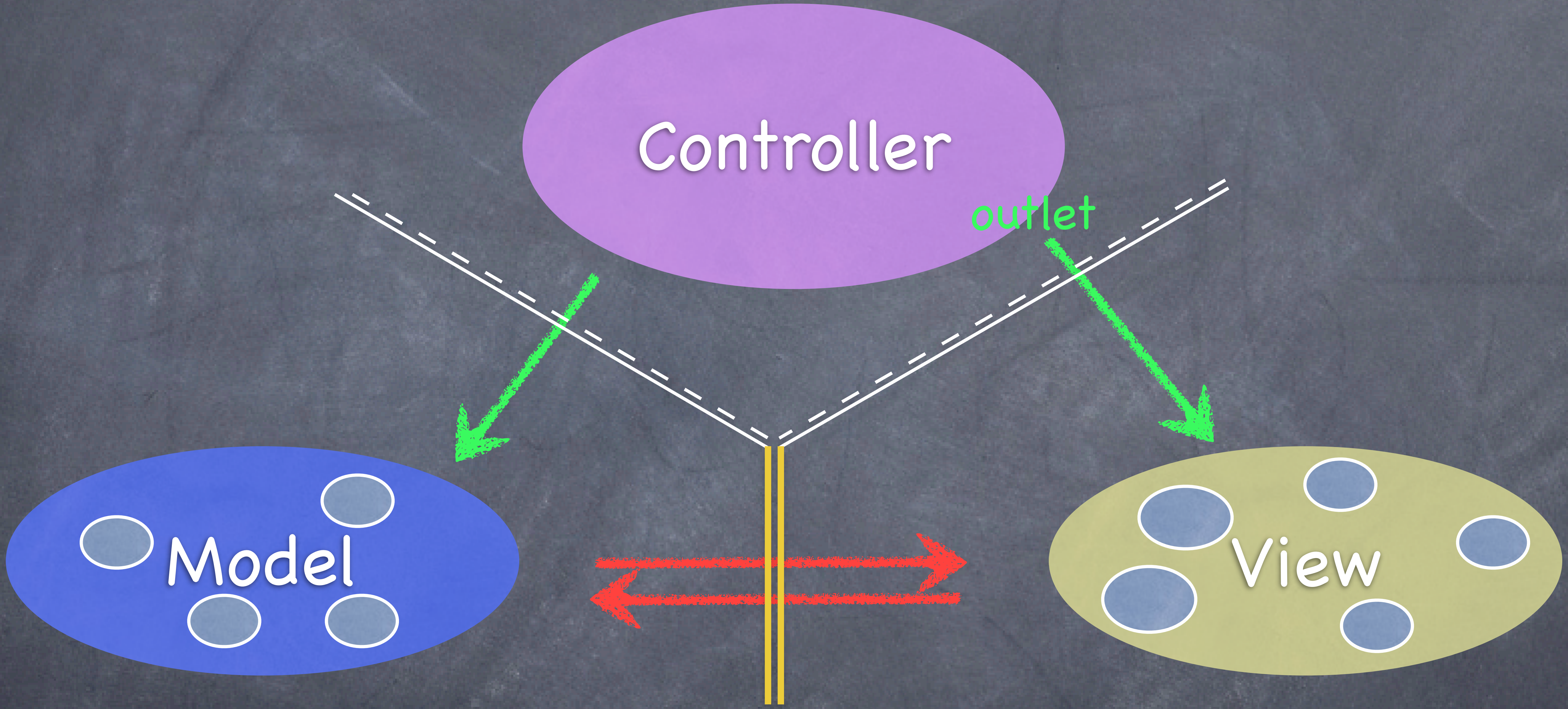
Controllers can always talk directly to their Model.

MVC



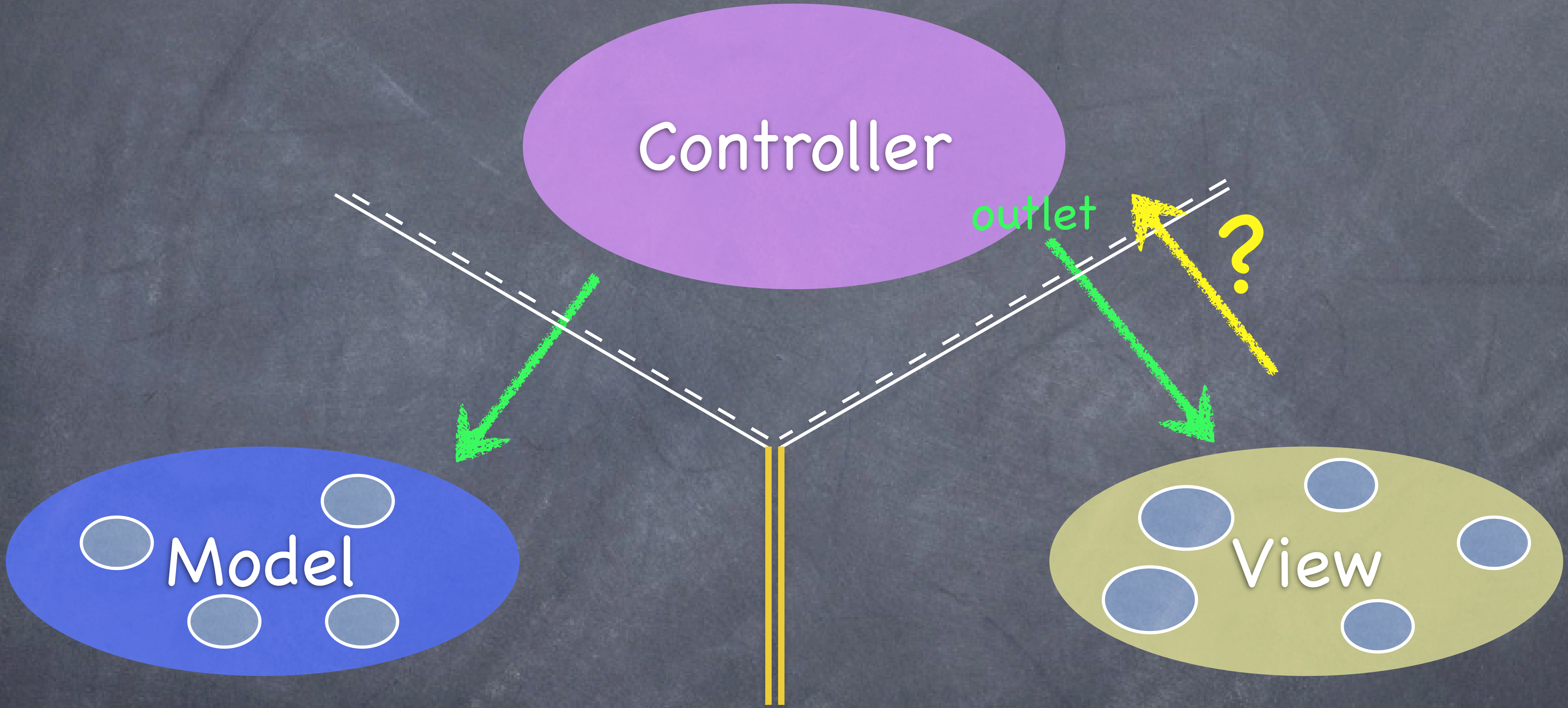
Controllers can also talk directly to their View.

MVC



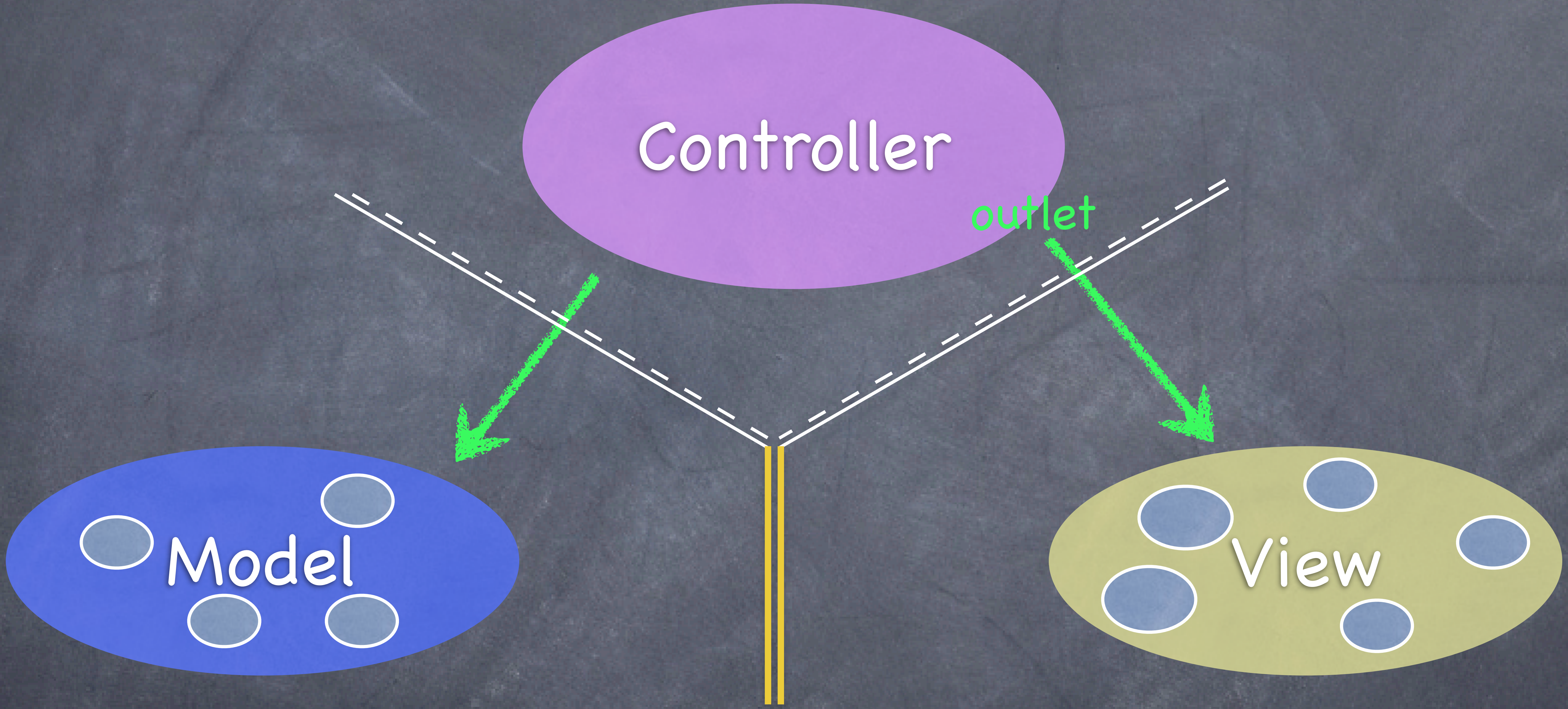
The Model and View should never speak to each other.

MVC



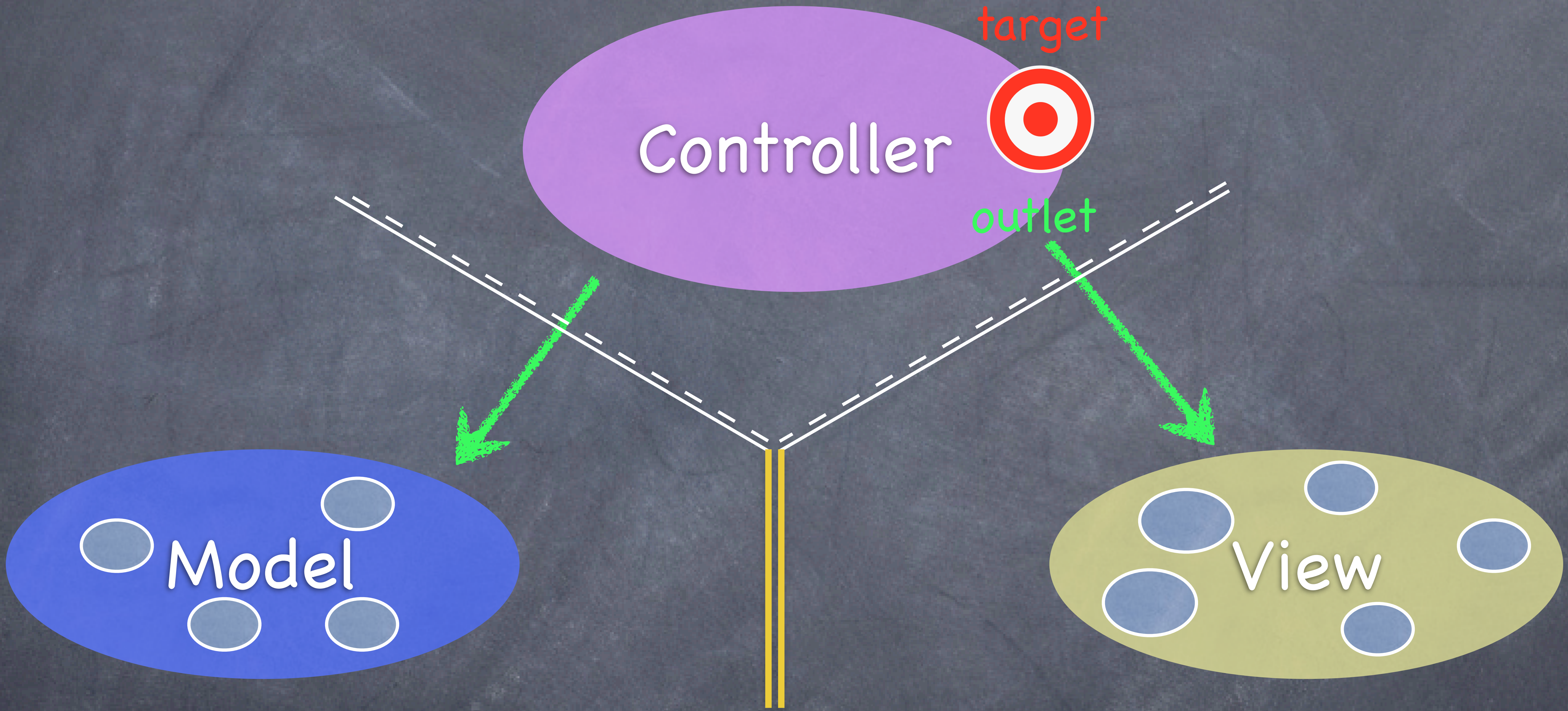
Can the View speak to its Controller?

MVC



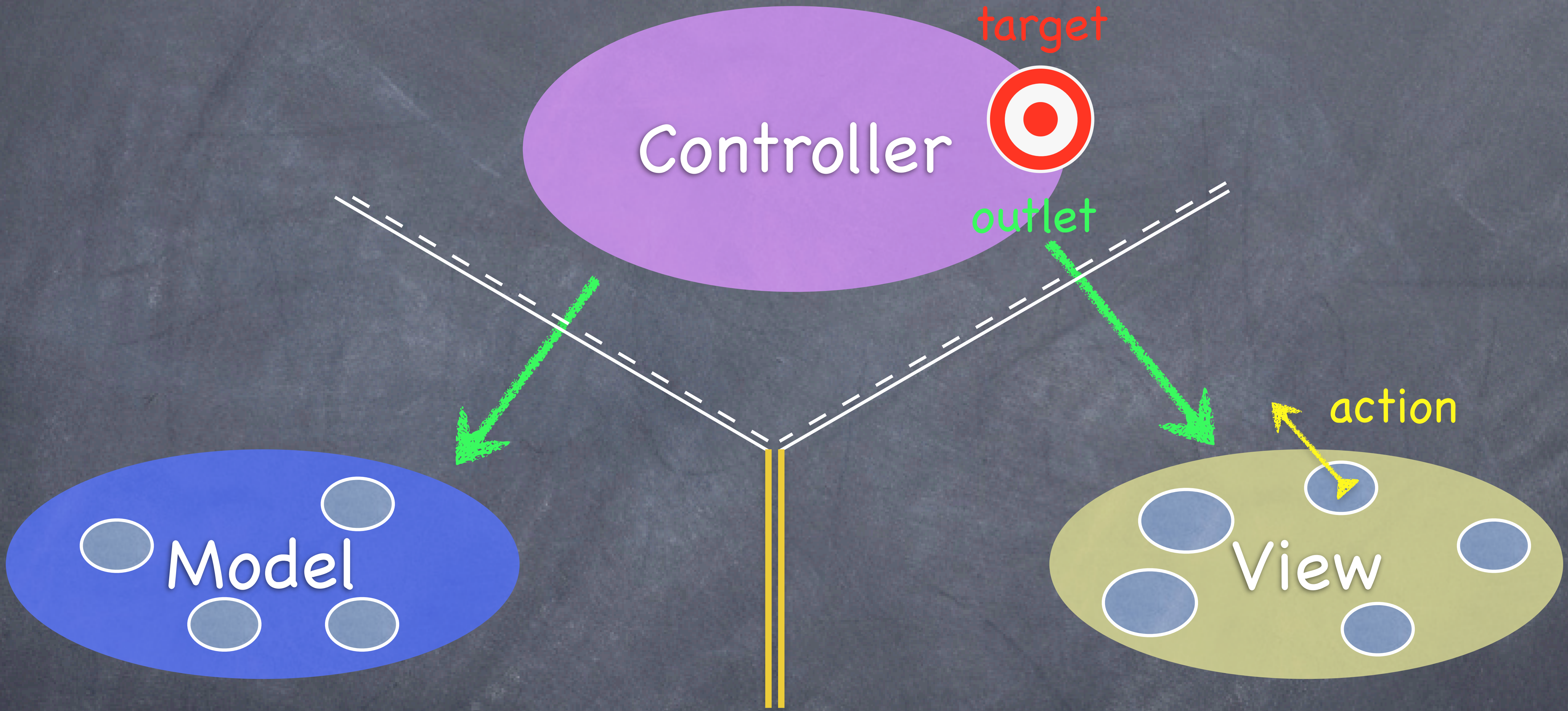
Sort of. Communication is "blind" and structured.

MVC



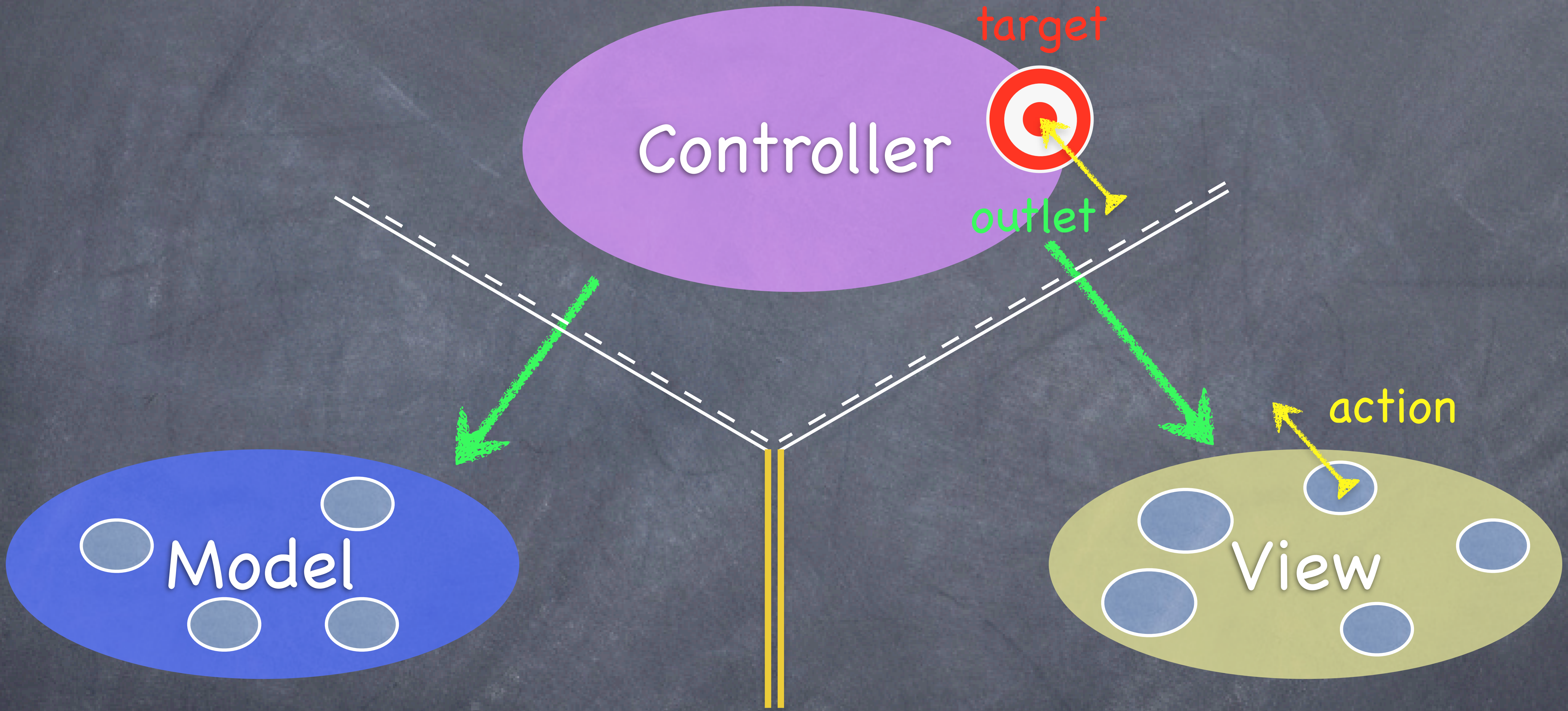
The Controller can drop a target on itself.

MVC



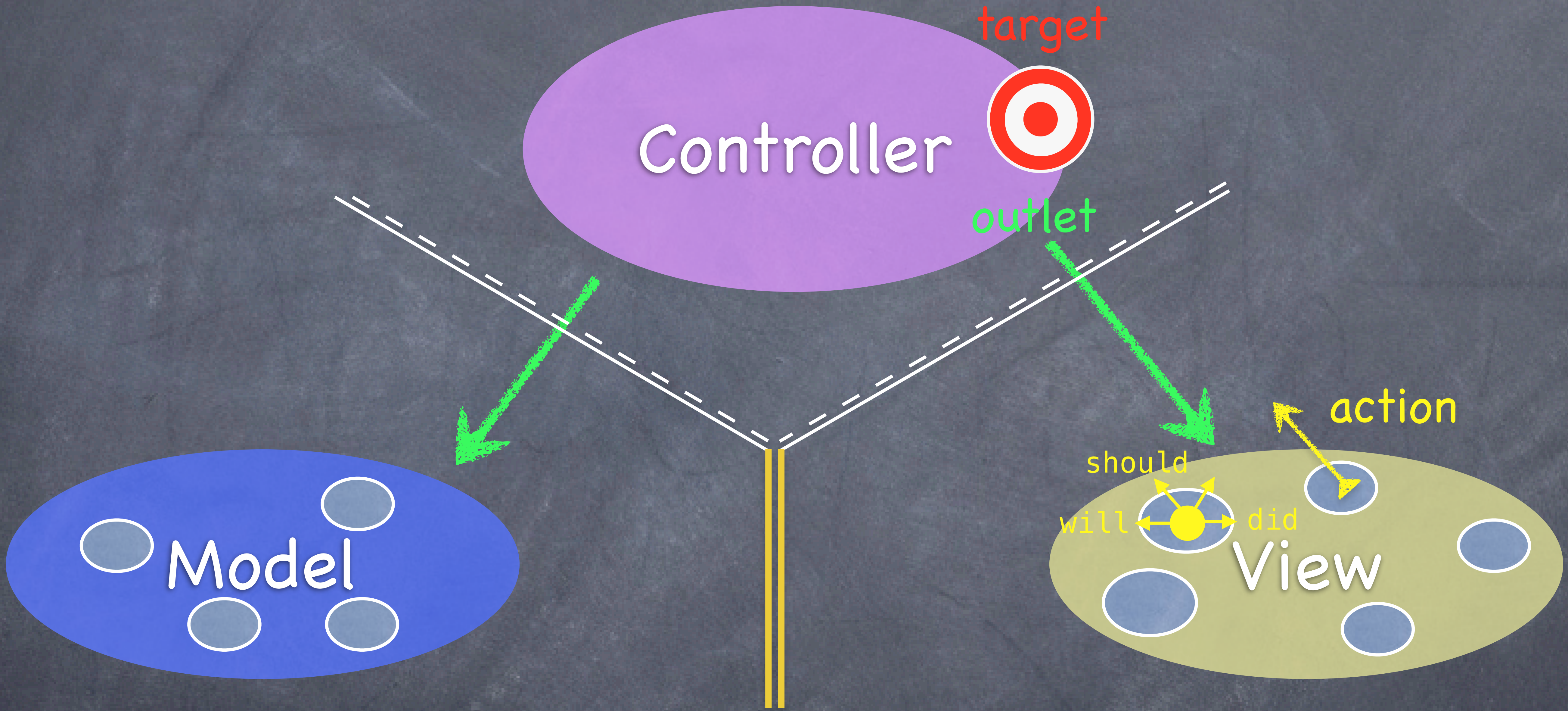
Then hand out an action to the View.

MVC



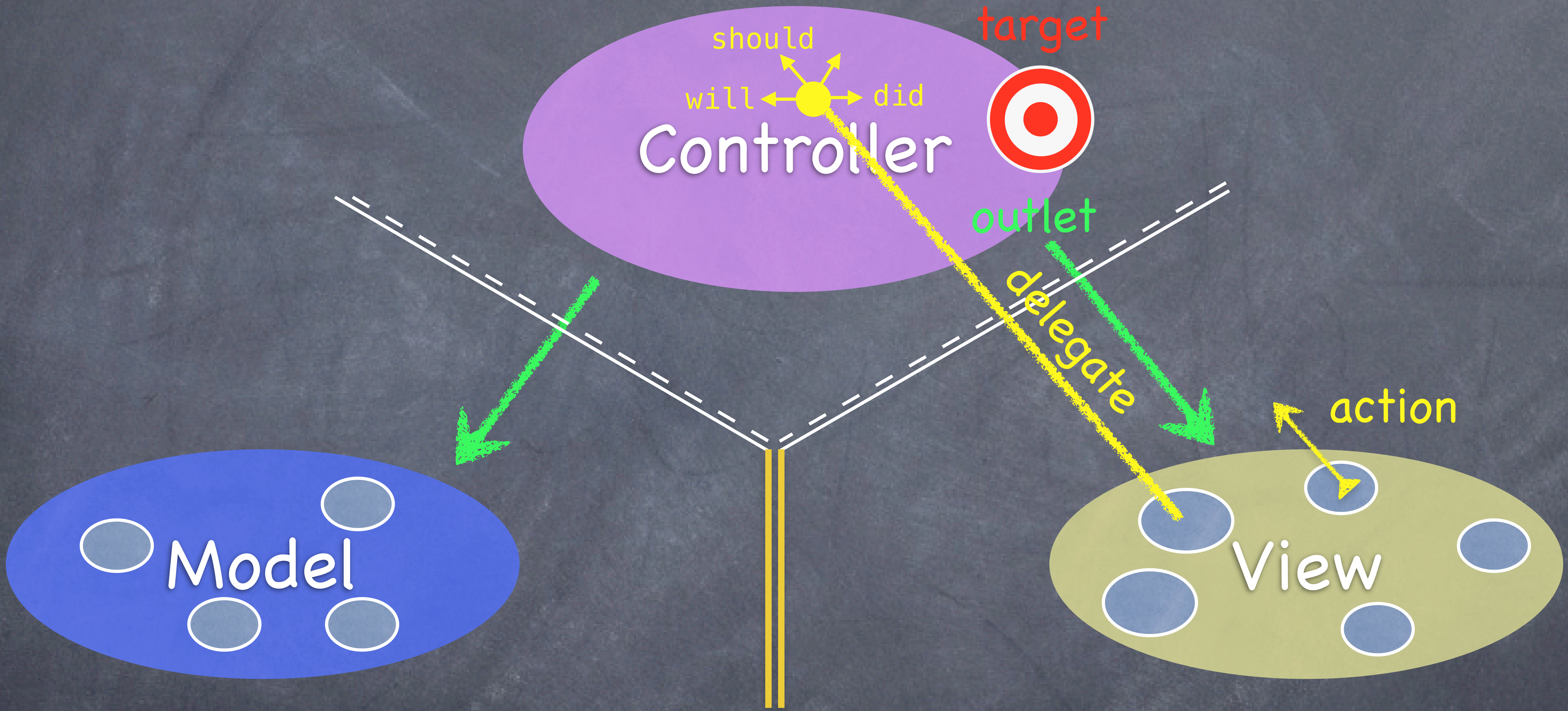
The View sends the action when things happen in the UI.

MVC



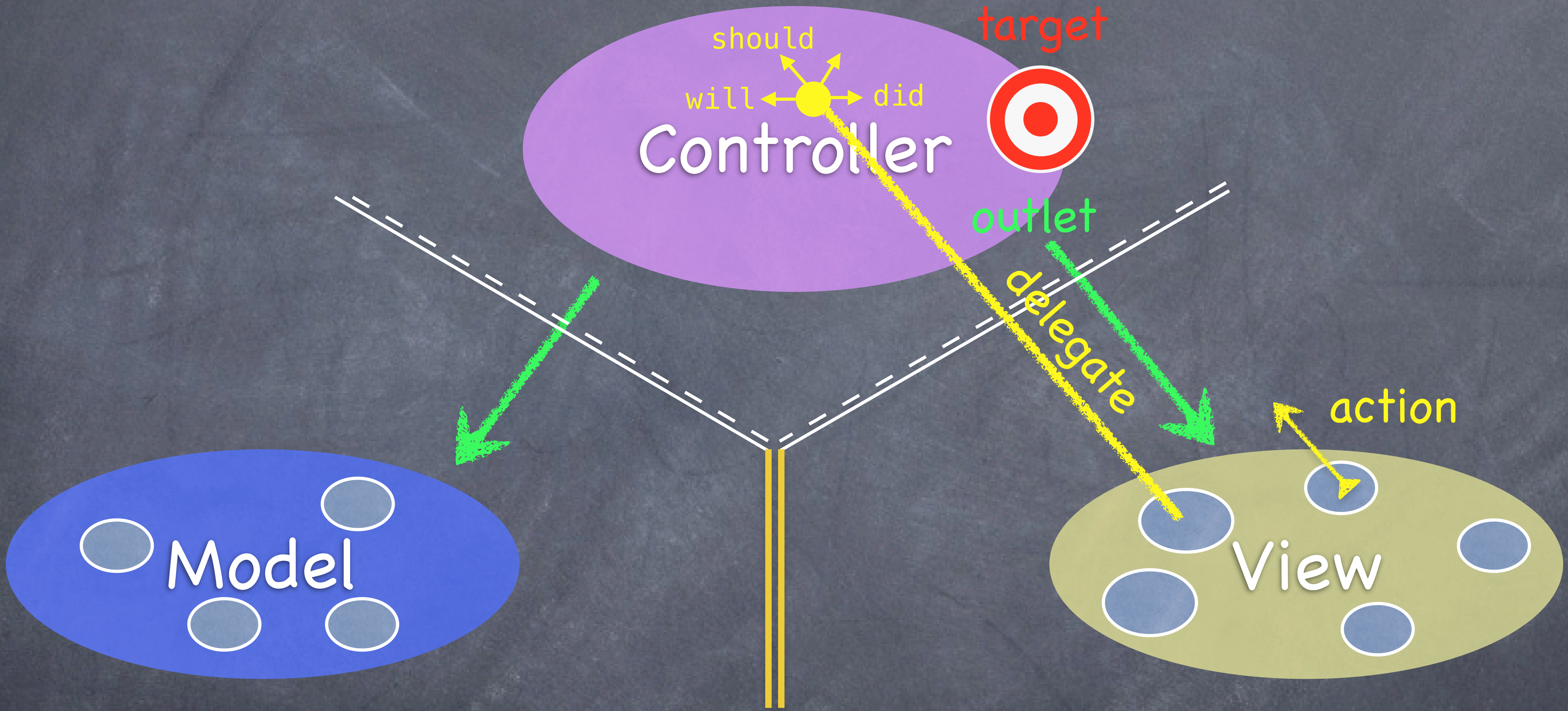
Sometimes the View needs to synchronize with the Controller.

MVC



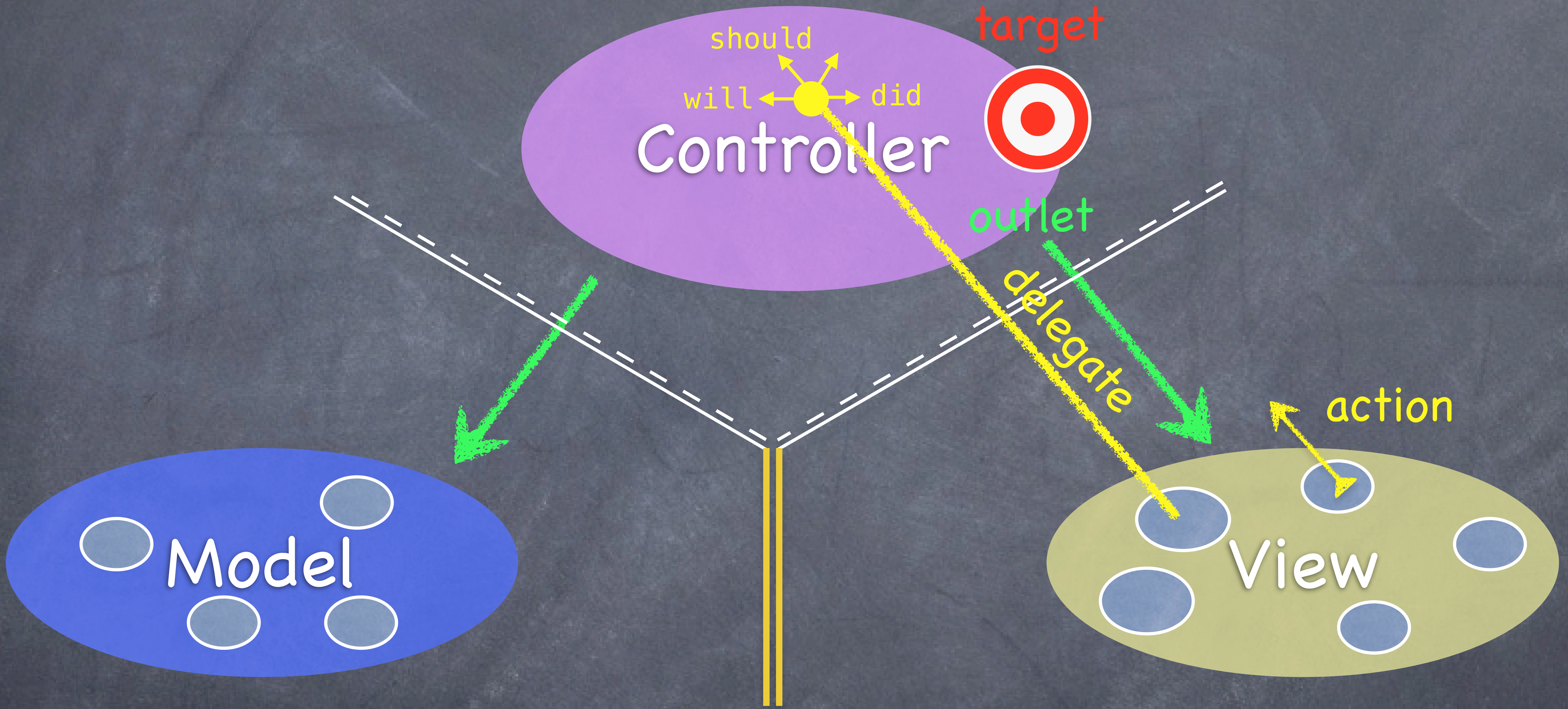
The Controller sets itself as the View's delegate.

MVC



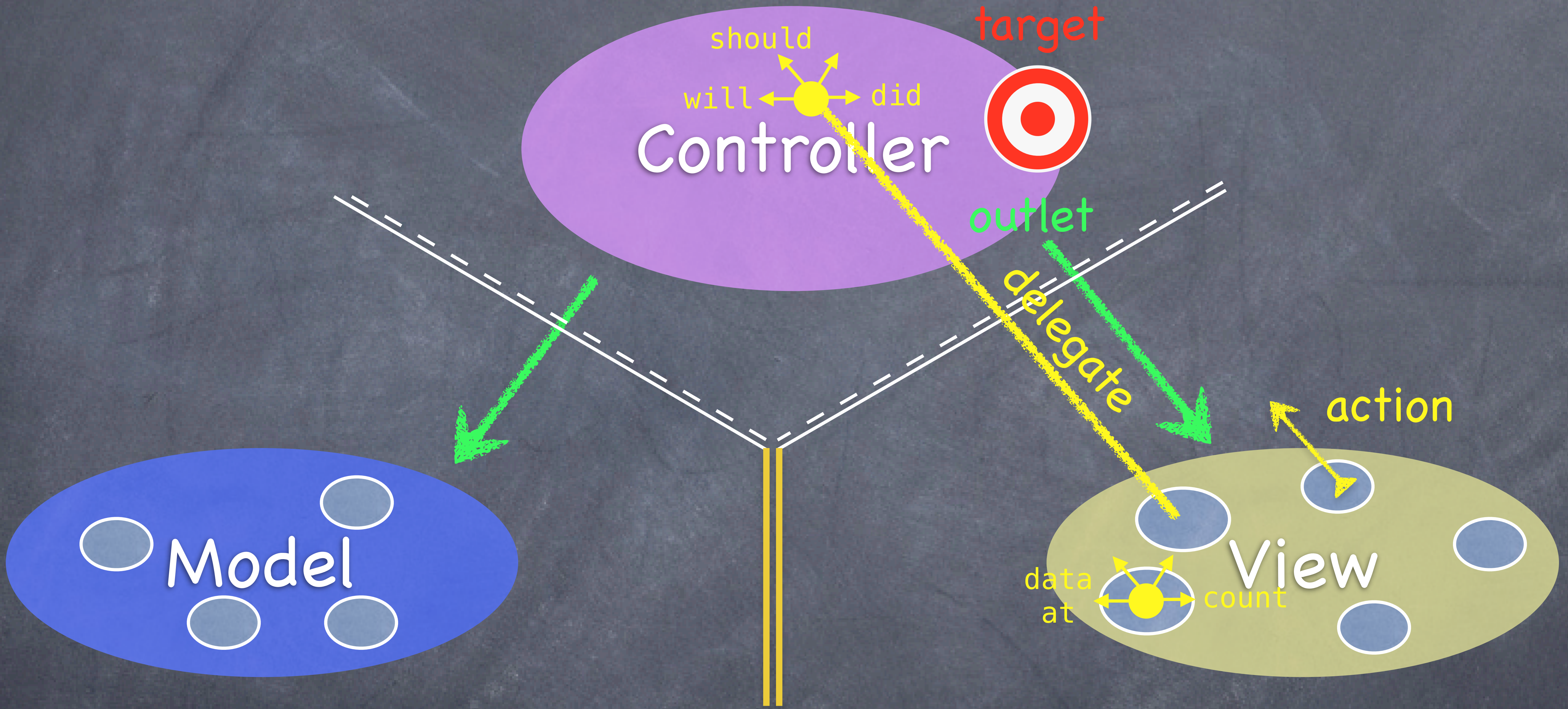
The delegate is set via a protocol (i.e. it's "blind" to class).

MVC



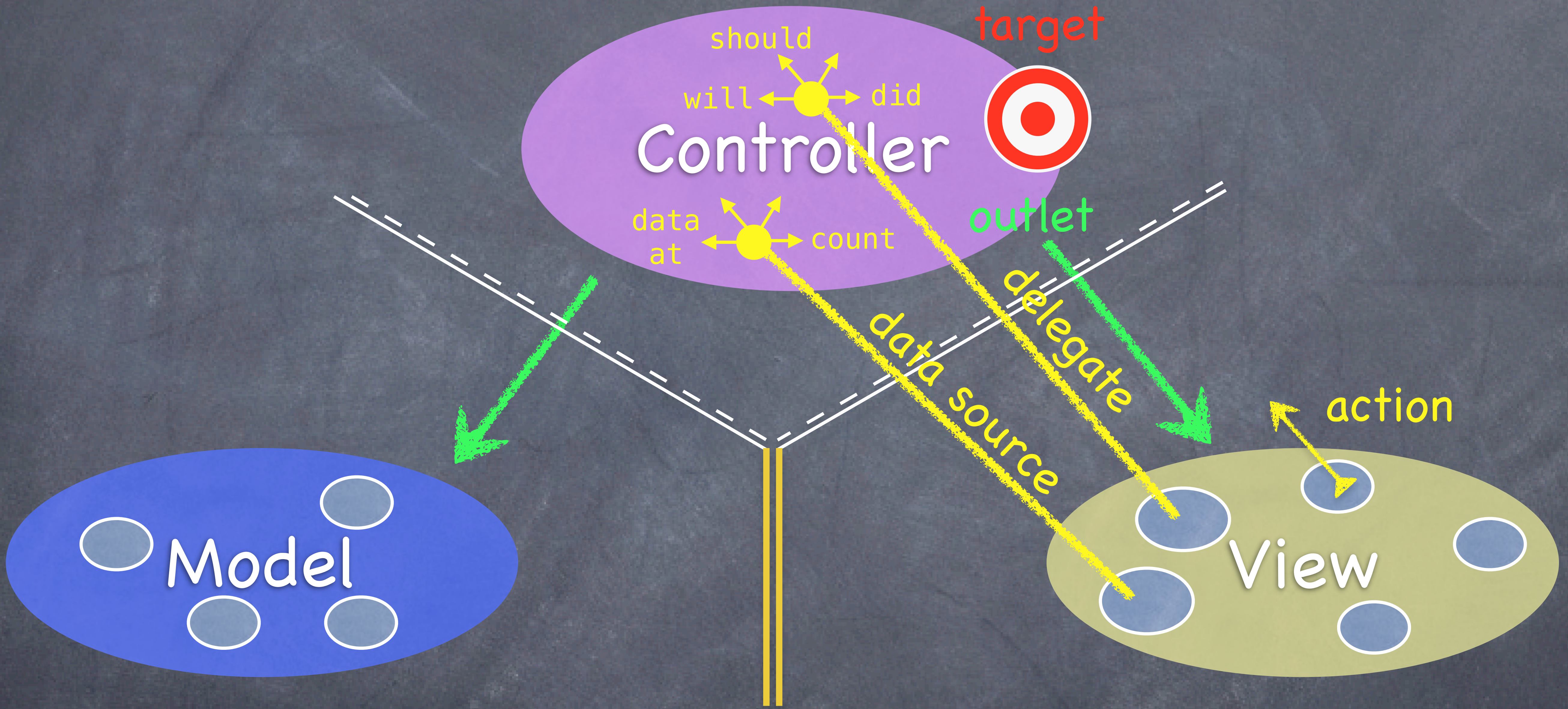
Views do not own the data they display.

MVC



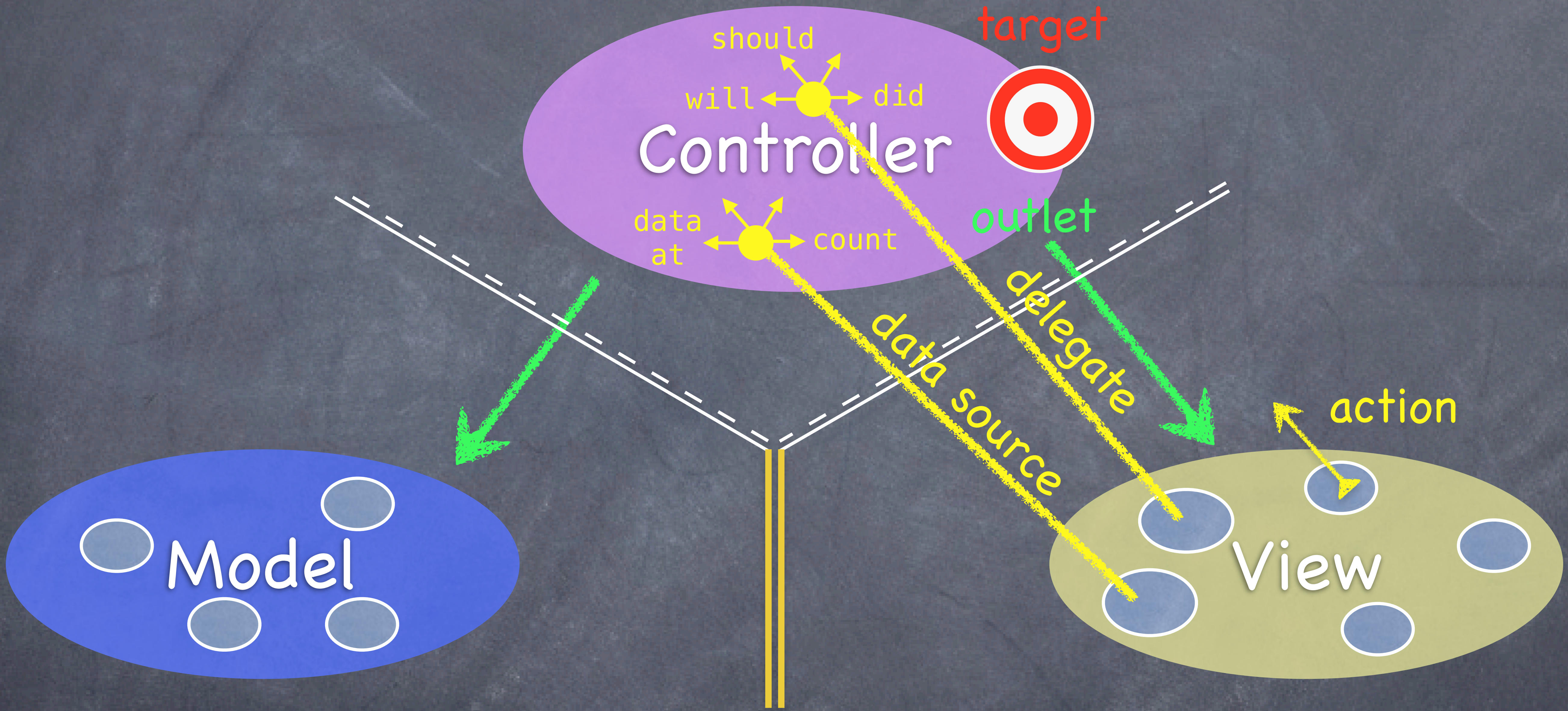
So, if needed, they have a protocol to acquire it.

MVC



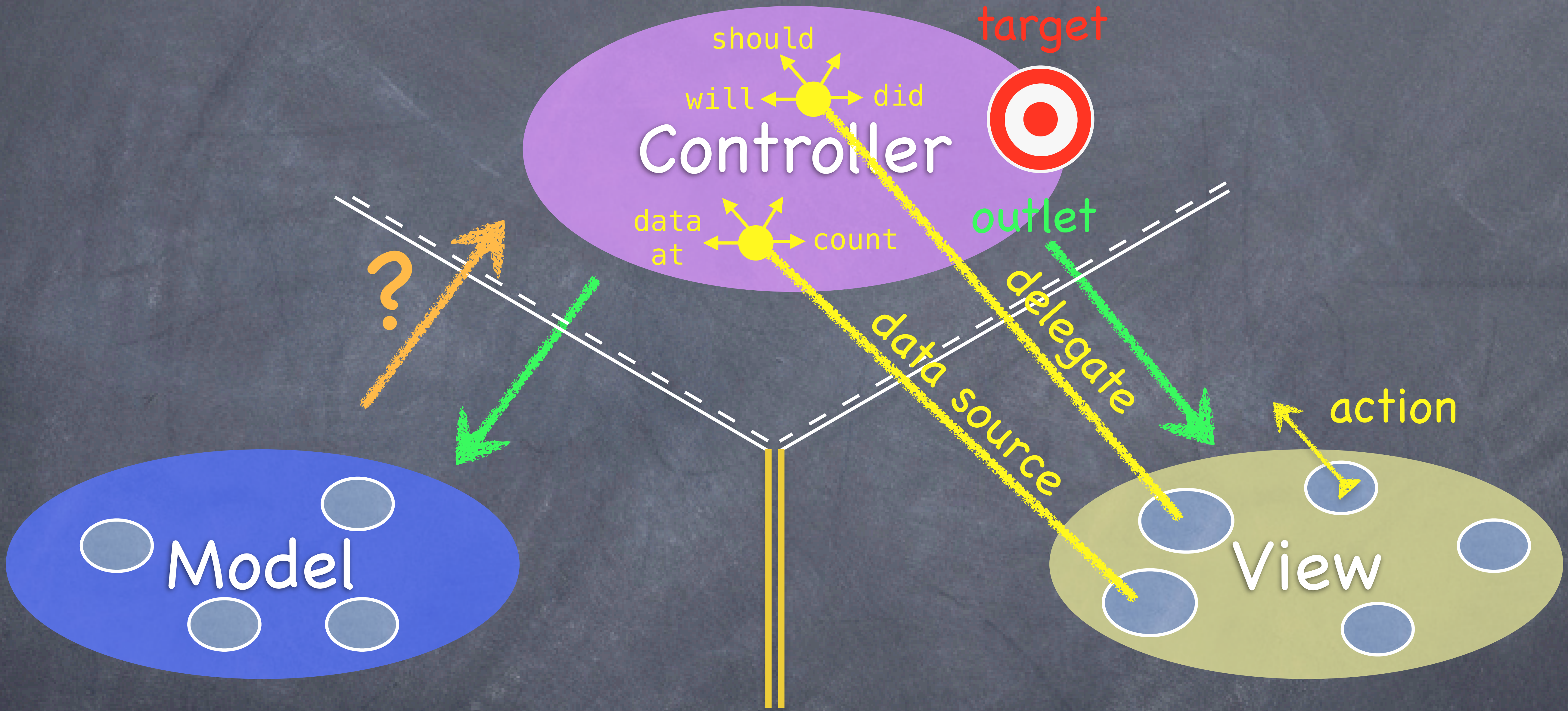
Controllers are almost always that data source (not Model!).

MVC



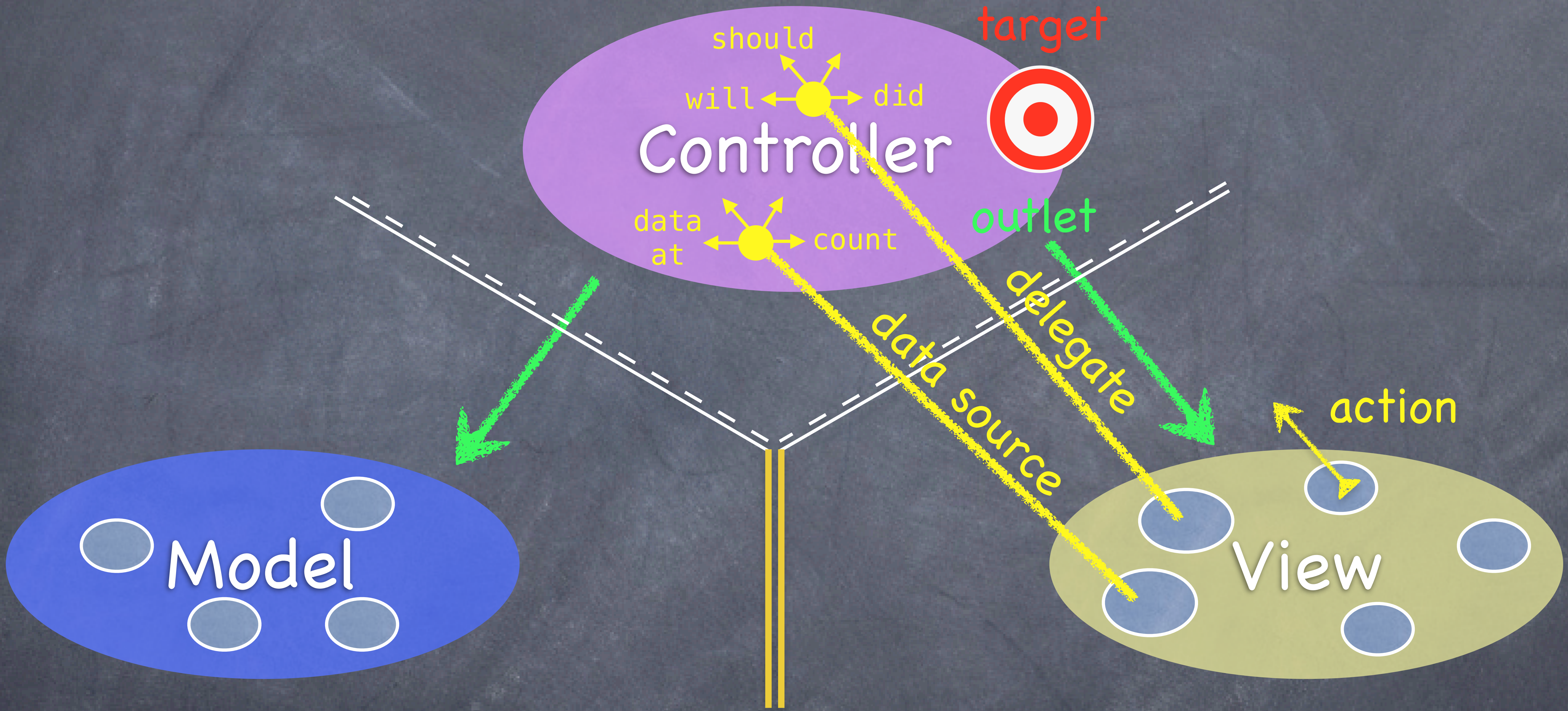
Controllers interpret/format Model information for the View.

MVC



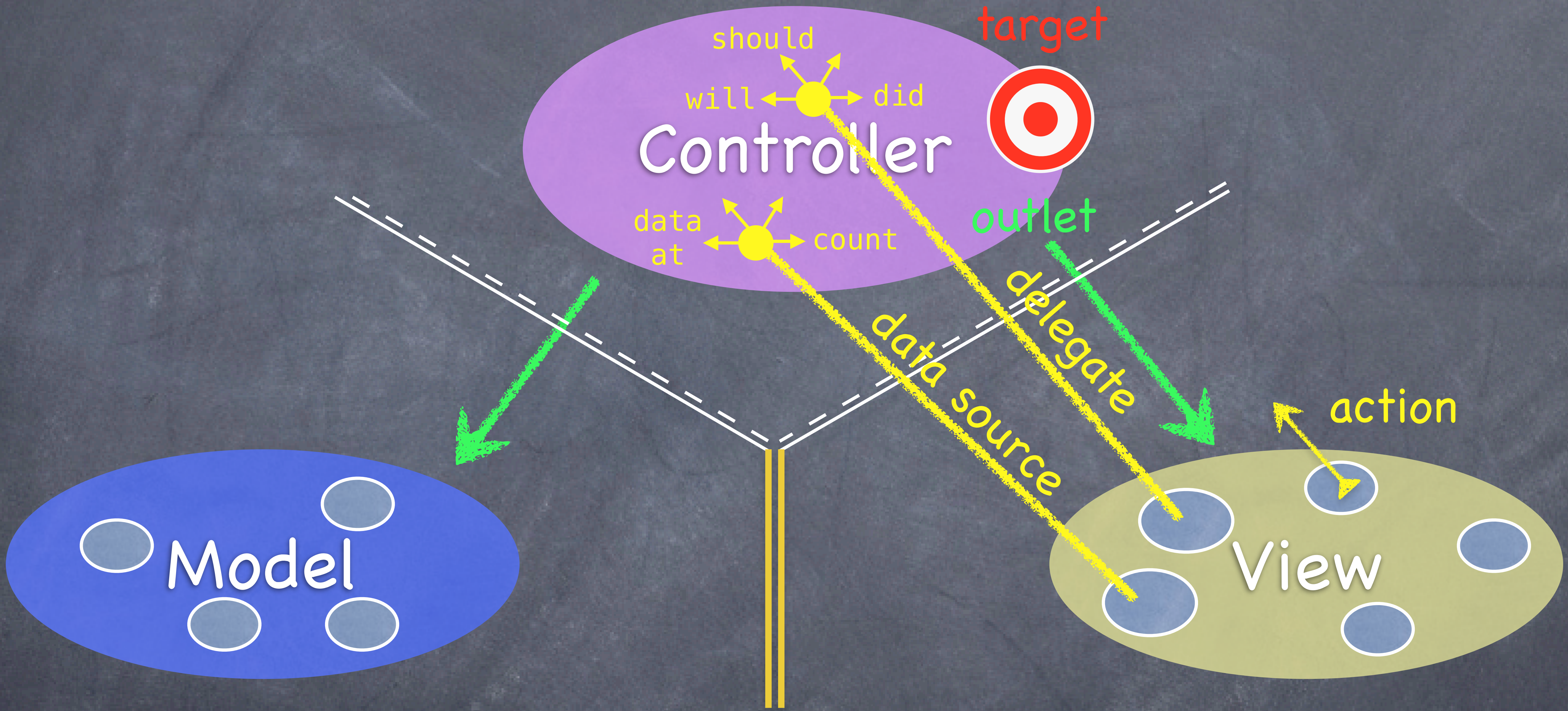
Can the Model talk directly to the Controller?

MVC



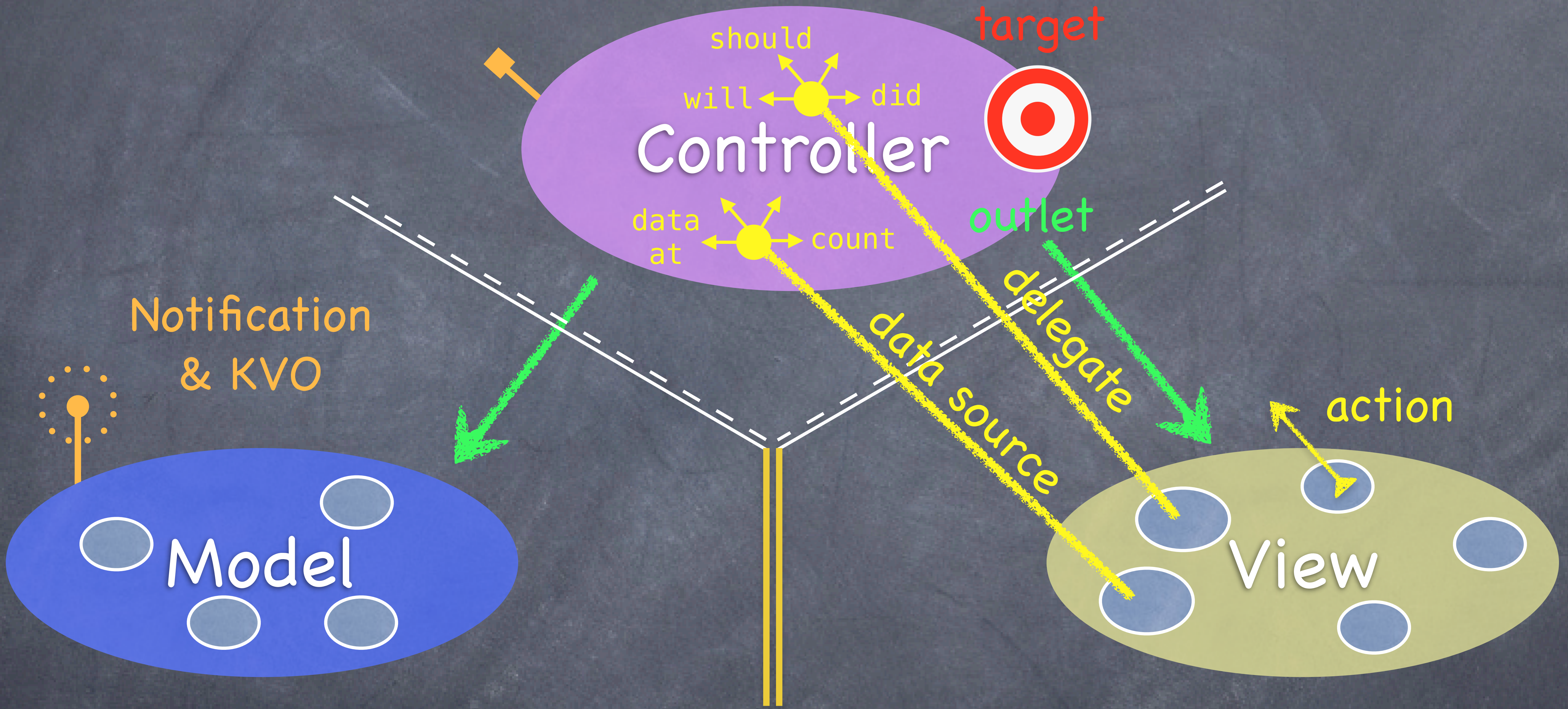
No. The Model is (should be) UI independent.

MVC



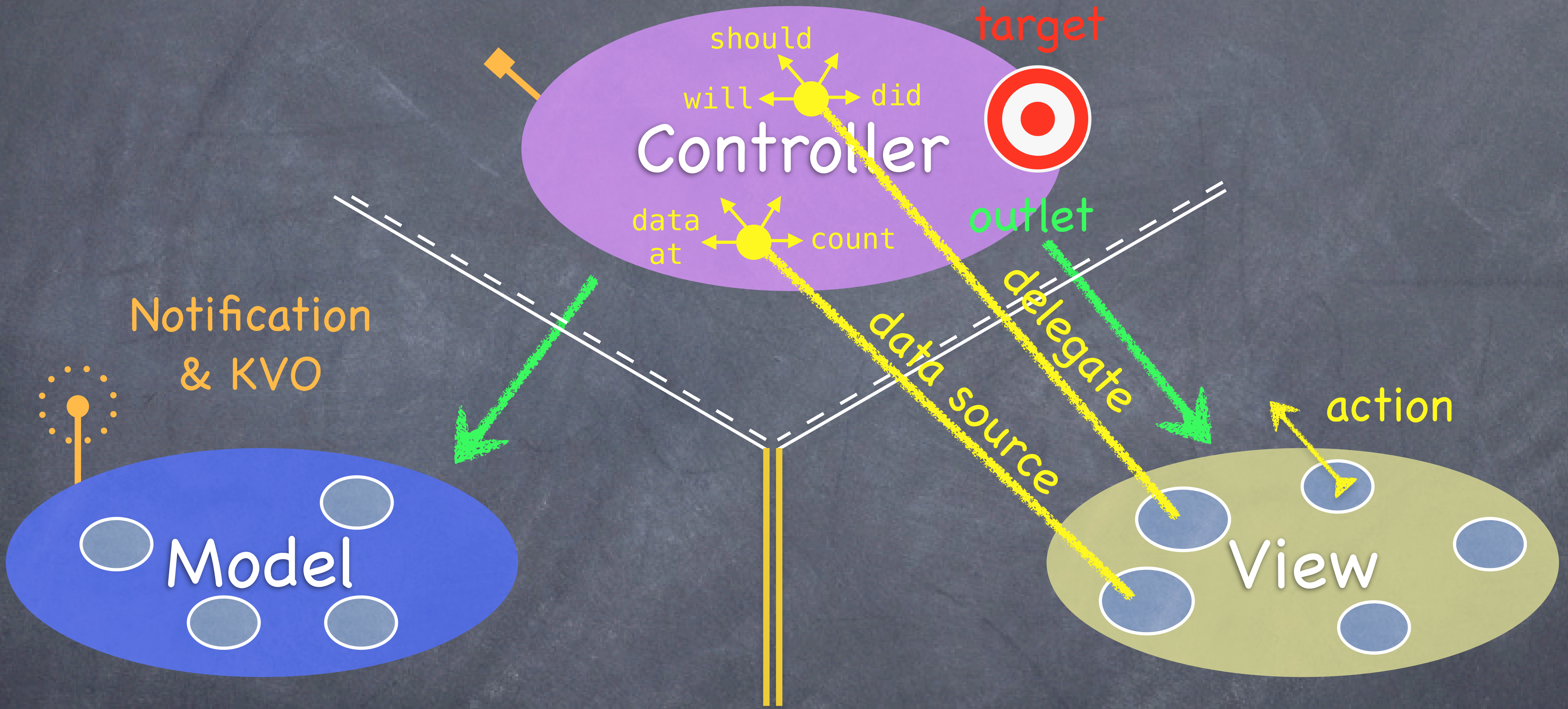
So what if the Model has information to update or something?

MVC



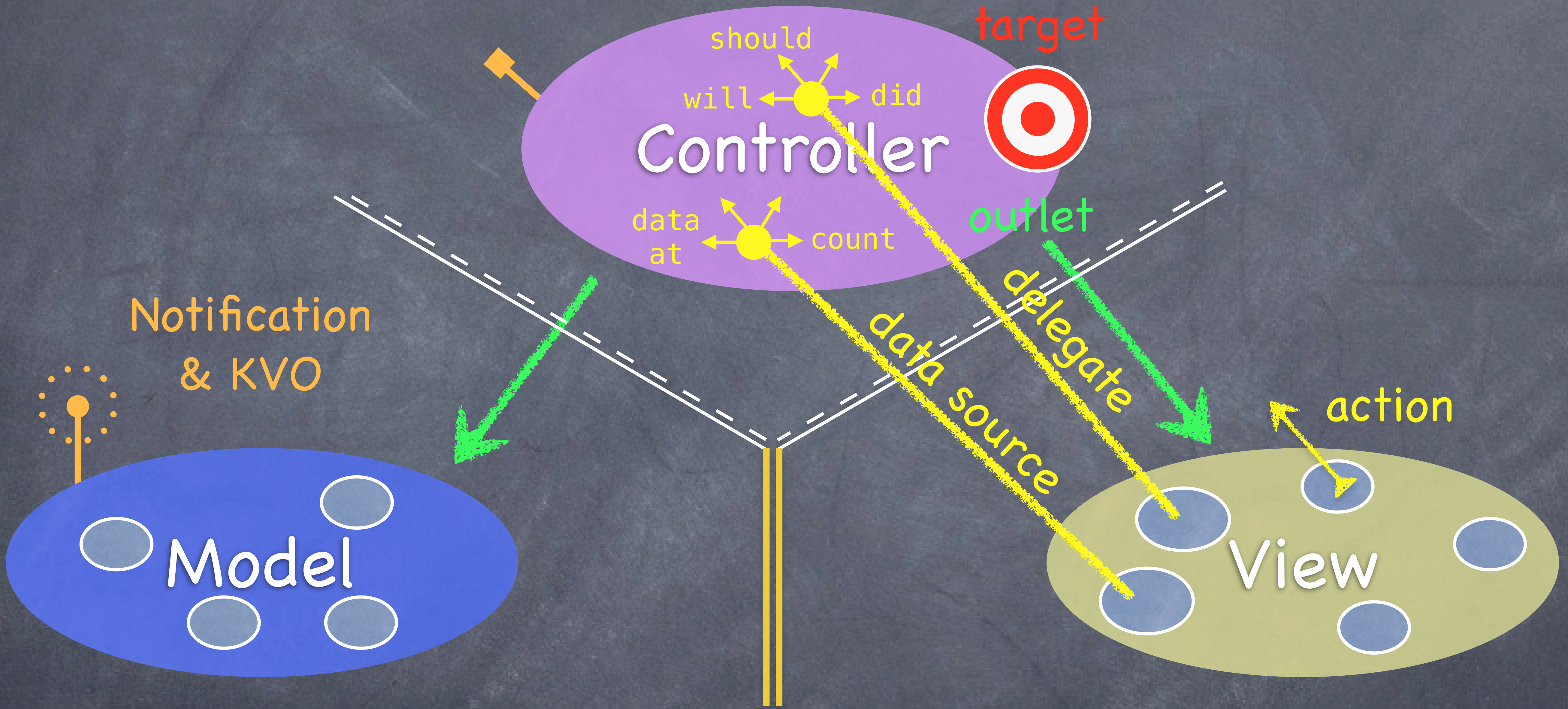
It uses a "radio station"-like broadcast mechanism.

MVC



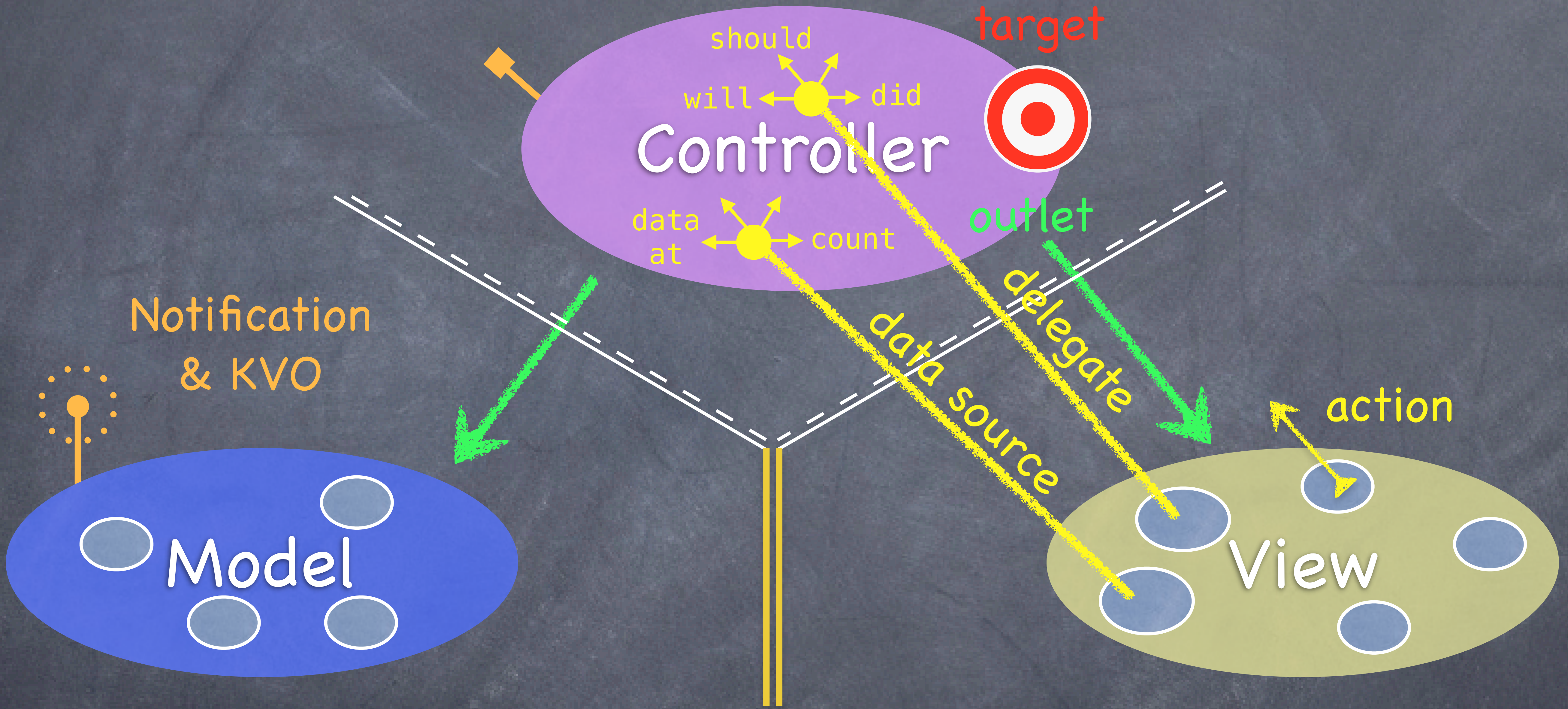
Controllers (or other Model) "tune in" to interesting stuff.

MVC



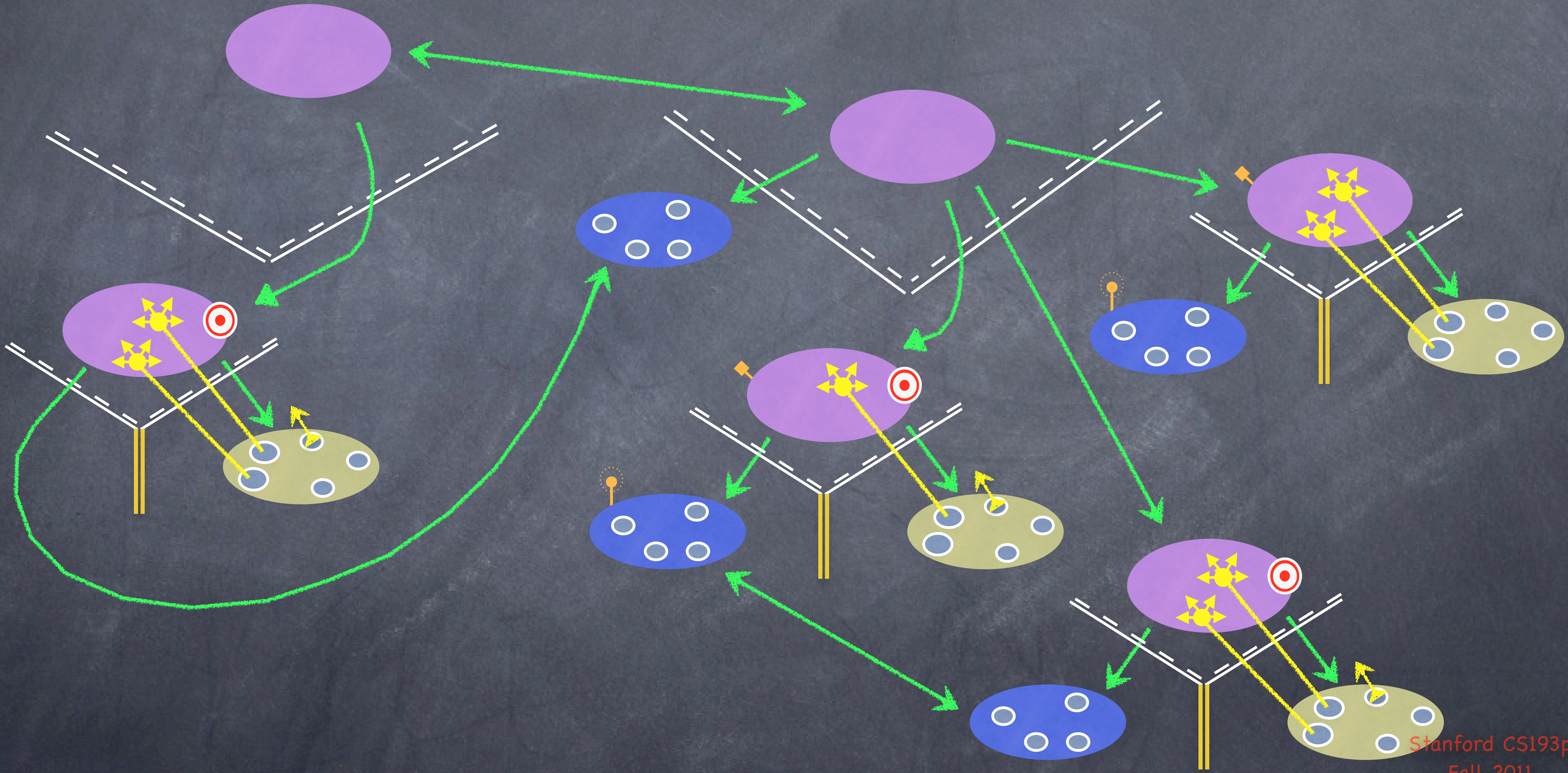
A View might "tune in," but probably not to a Model's "station."

MVC

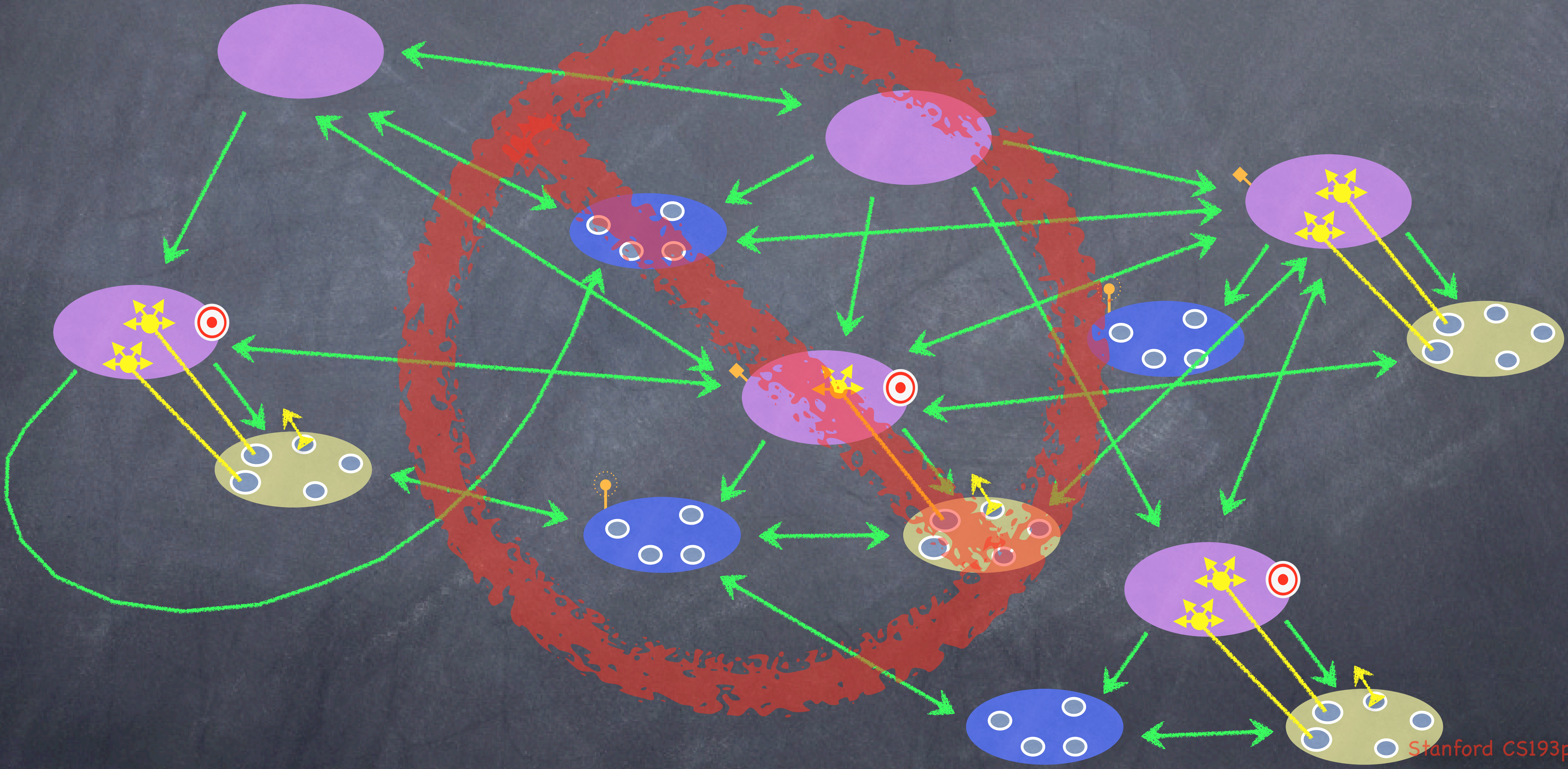


Now combine MVC groups to make complicated programs ...

MVCs working together

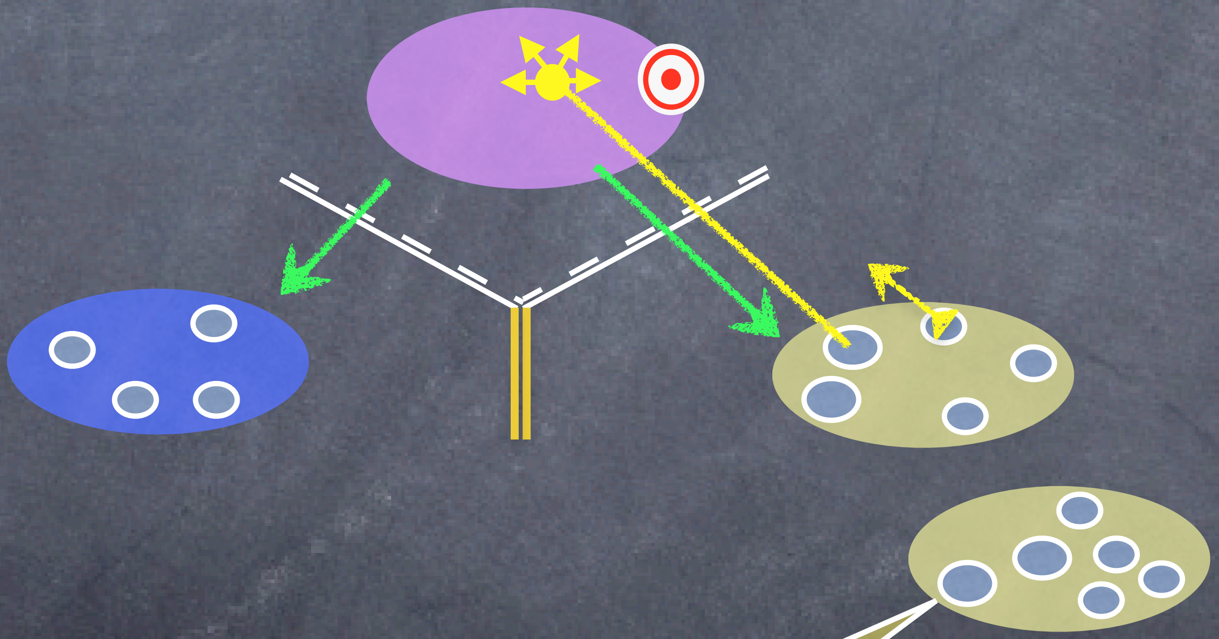


MVCs not working together



MVCs working together

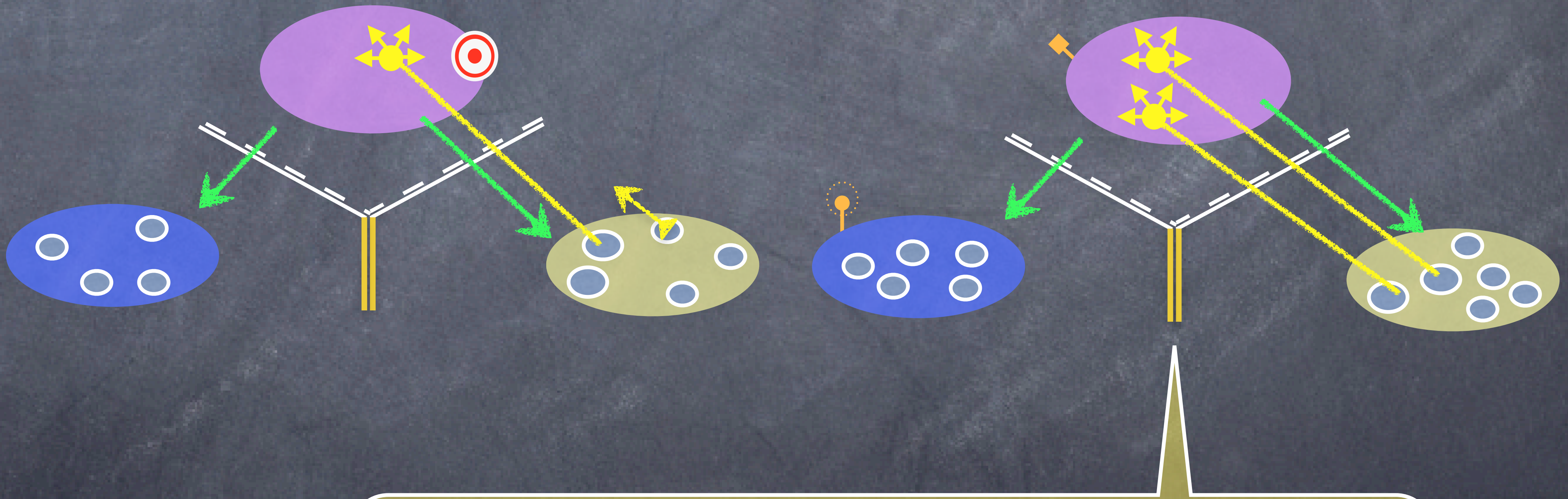
What happens when your application gets more features?



Now all of your UI can't fit in one MVC's view.

MVCs working together

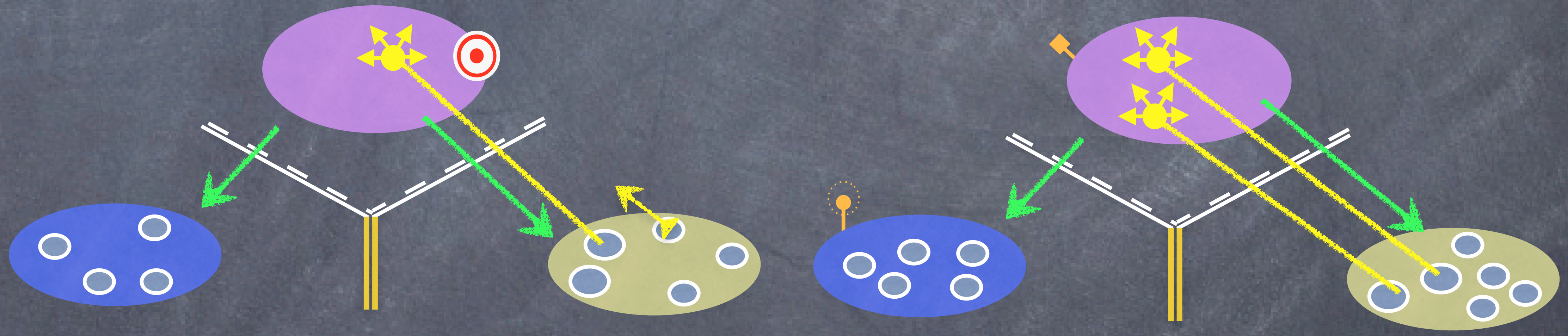
What happens when your application gets more features?



We never have an MVC's view span across screens.
So we'll have to create a new MVC for these new features.

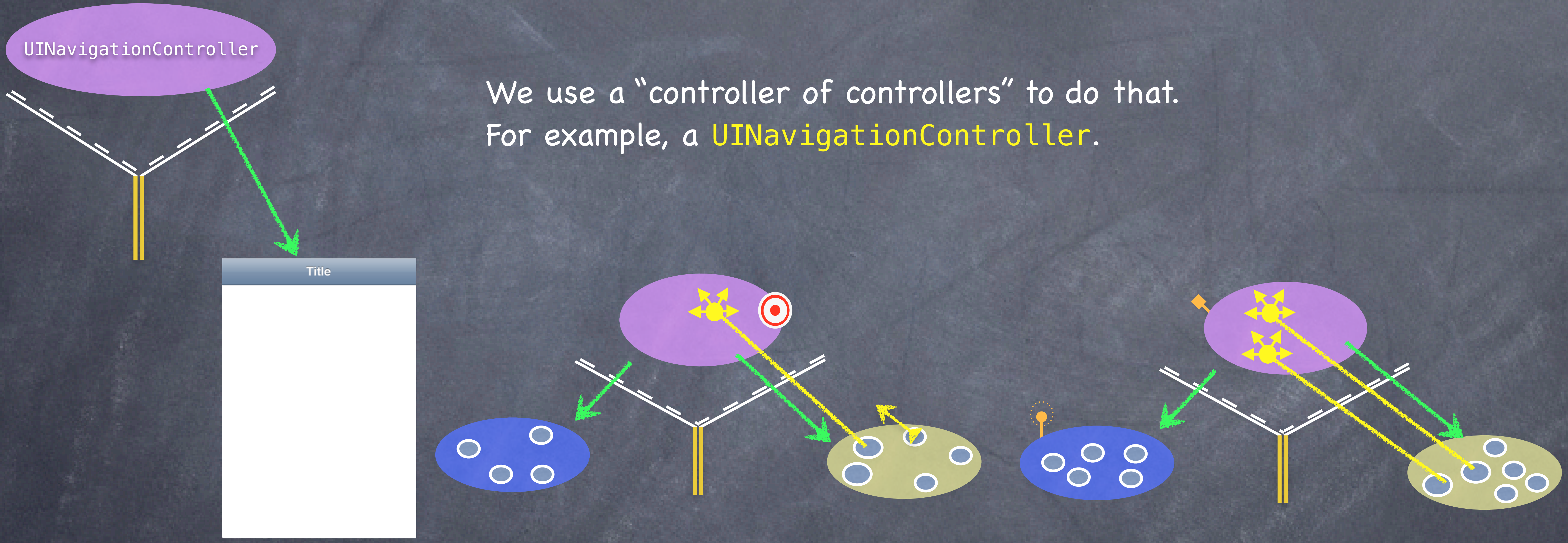
MVCs working together

So how do we switch the screen to show this other MVC?



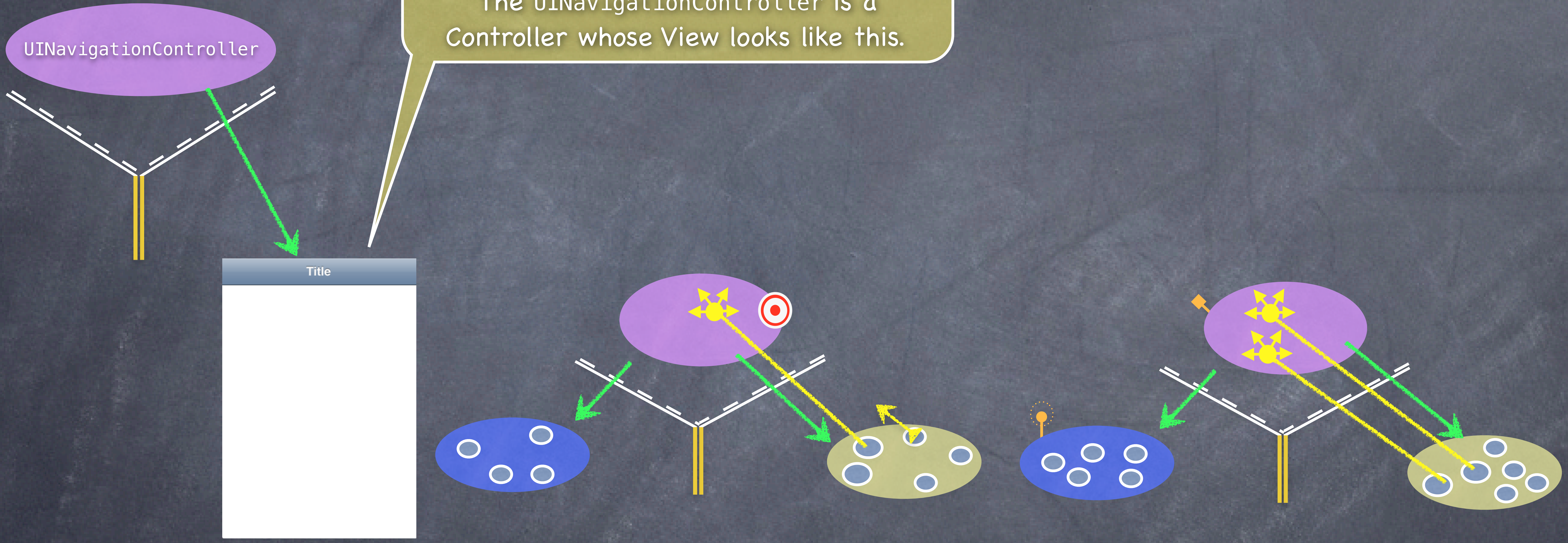
MVCs working together

We use a "controller of controllers" to do that.
For example, a UINavigationController.

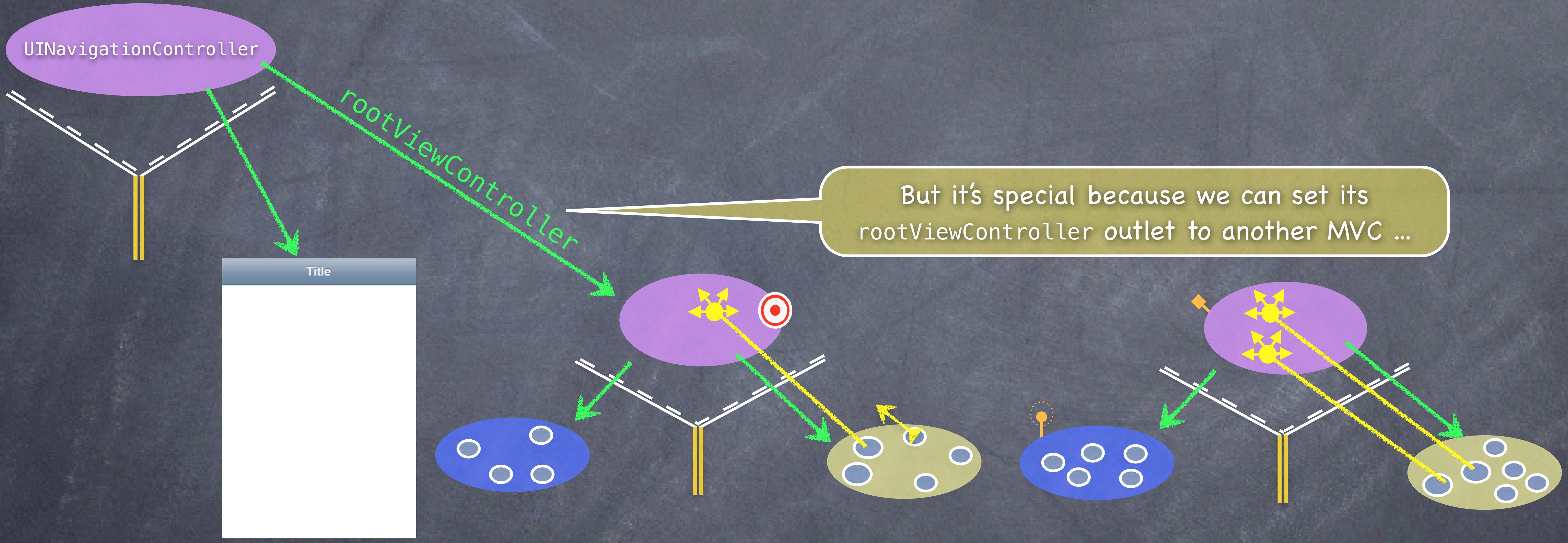


MVCs working together

The UINavigationController is a Controller whose View looks like this.

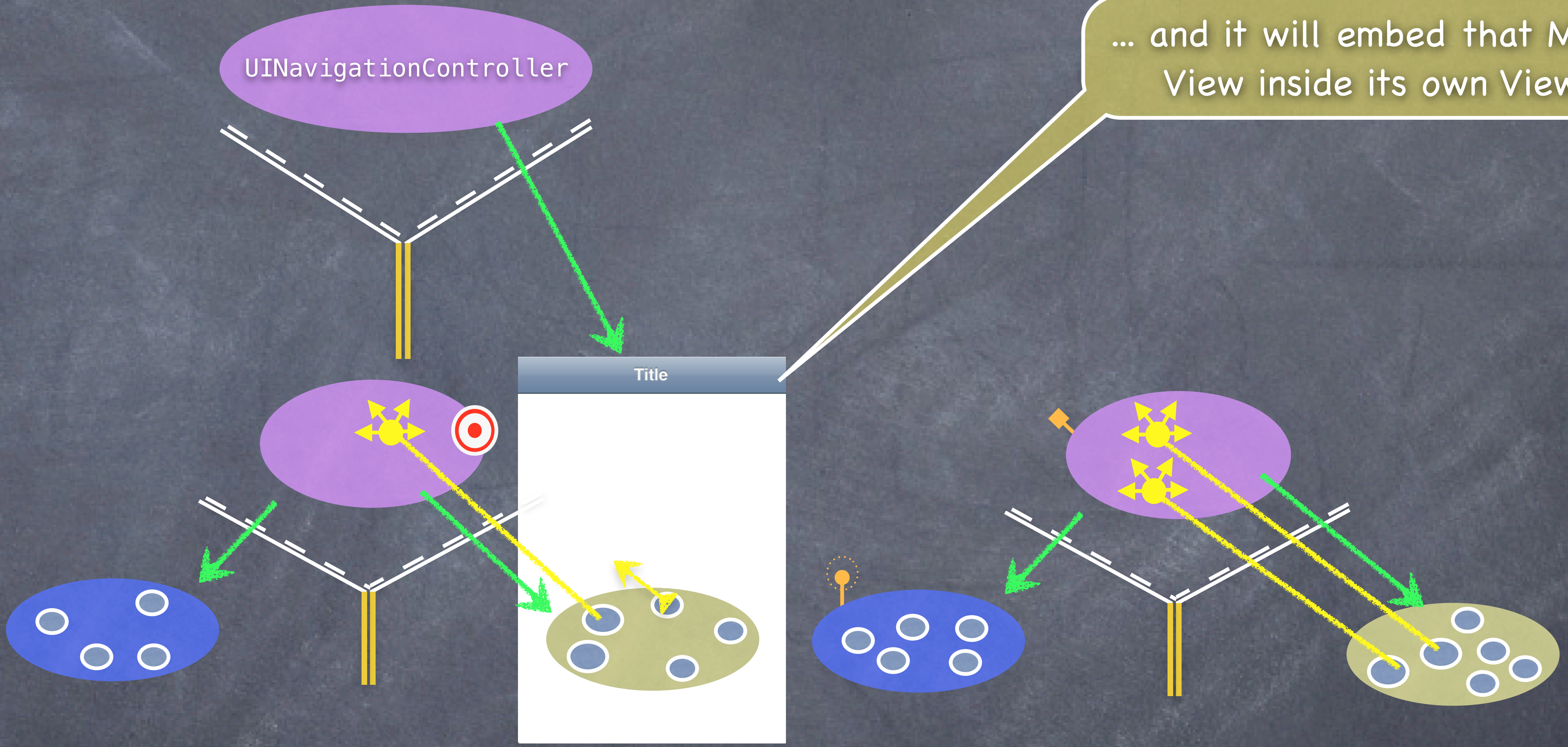


MVCs working together



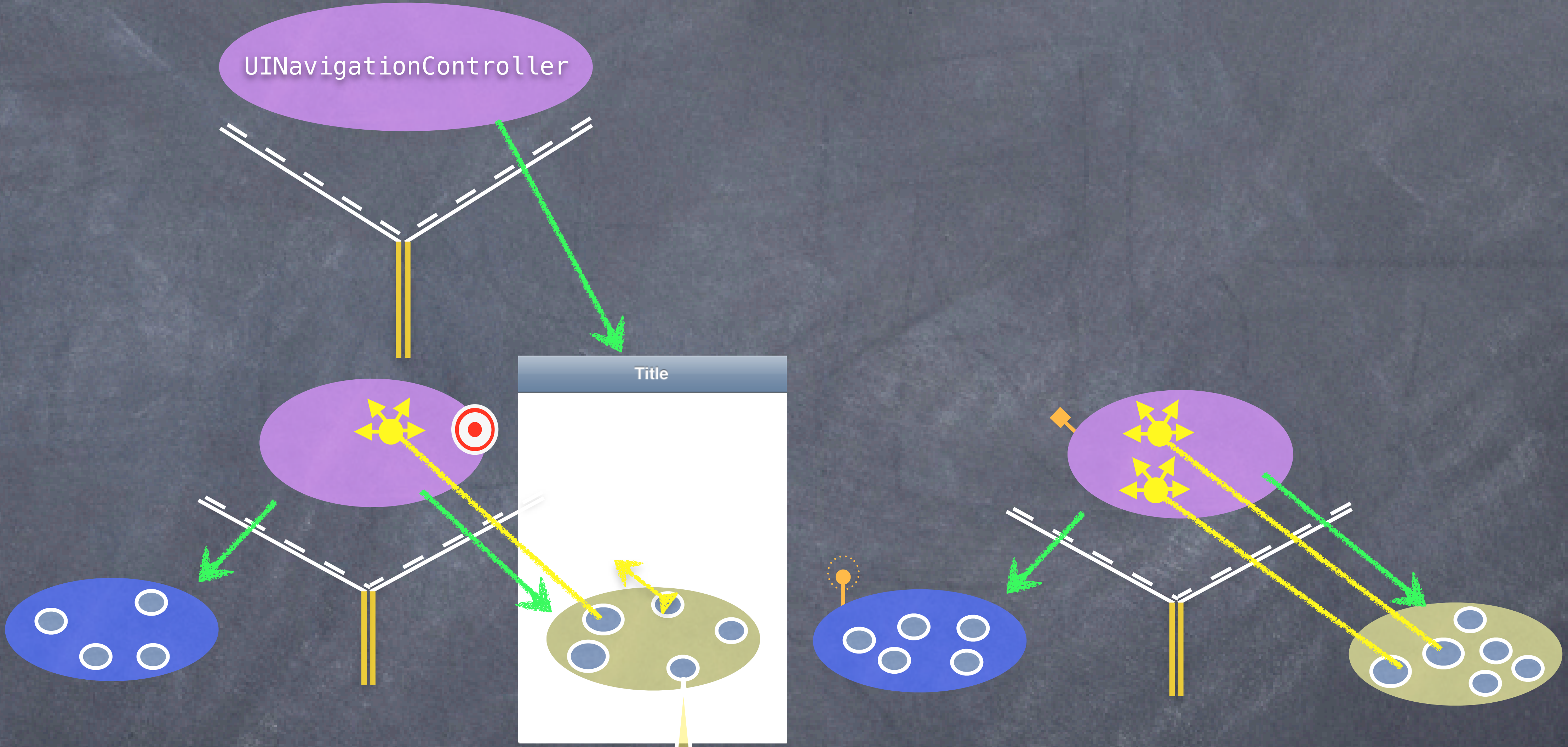
But it's special because we can set its `rootViewController` outlet to another MVC ...

MVCs working together



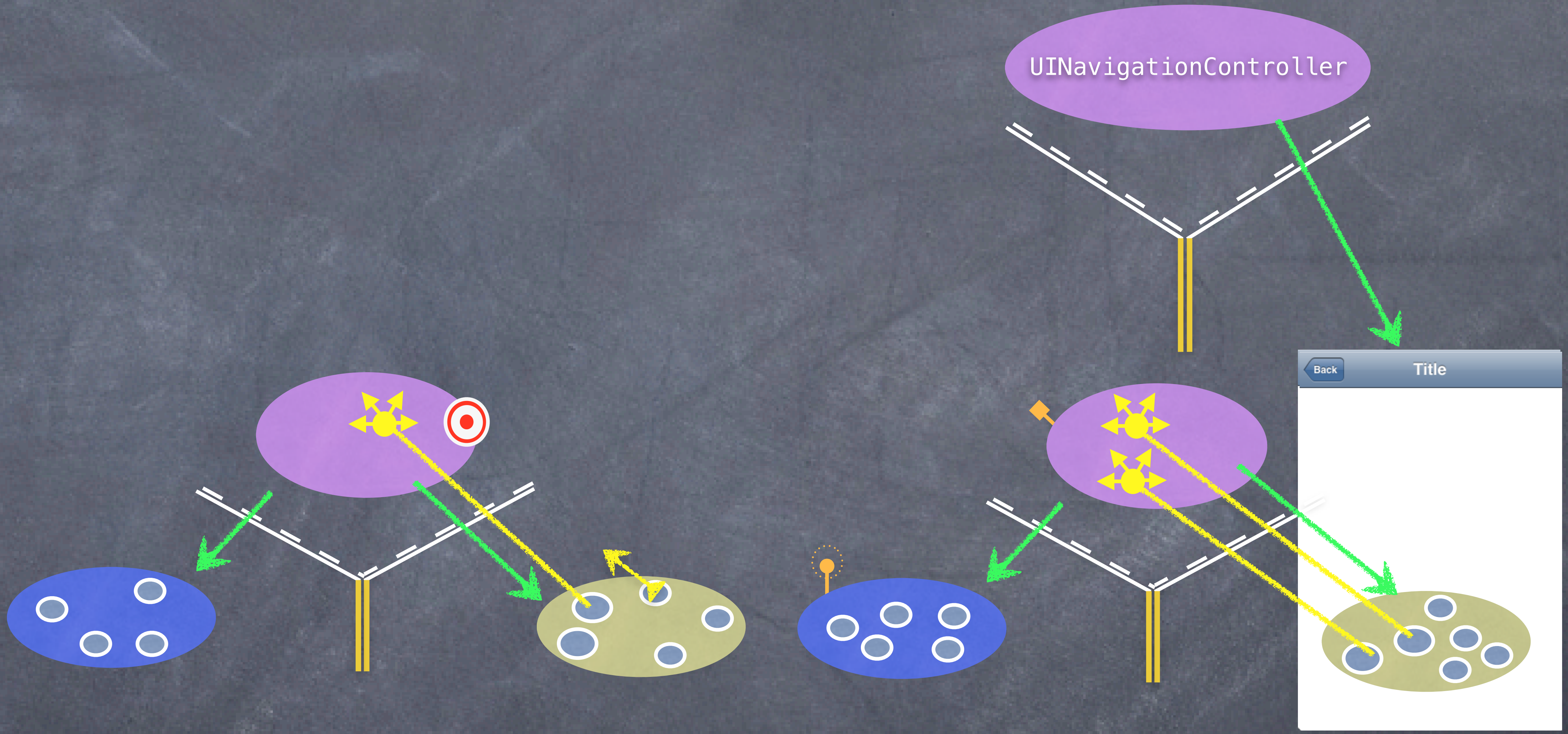
... and it will embed that MVC's View inside its own View.

MVCs working together

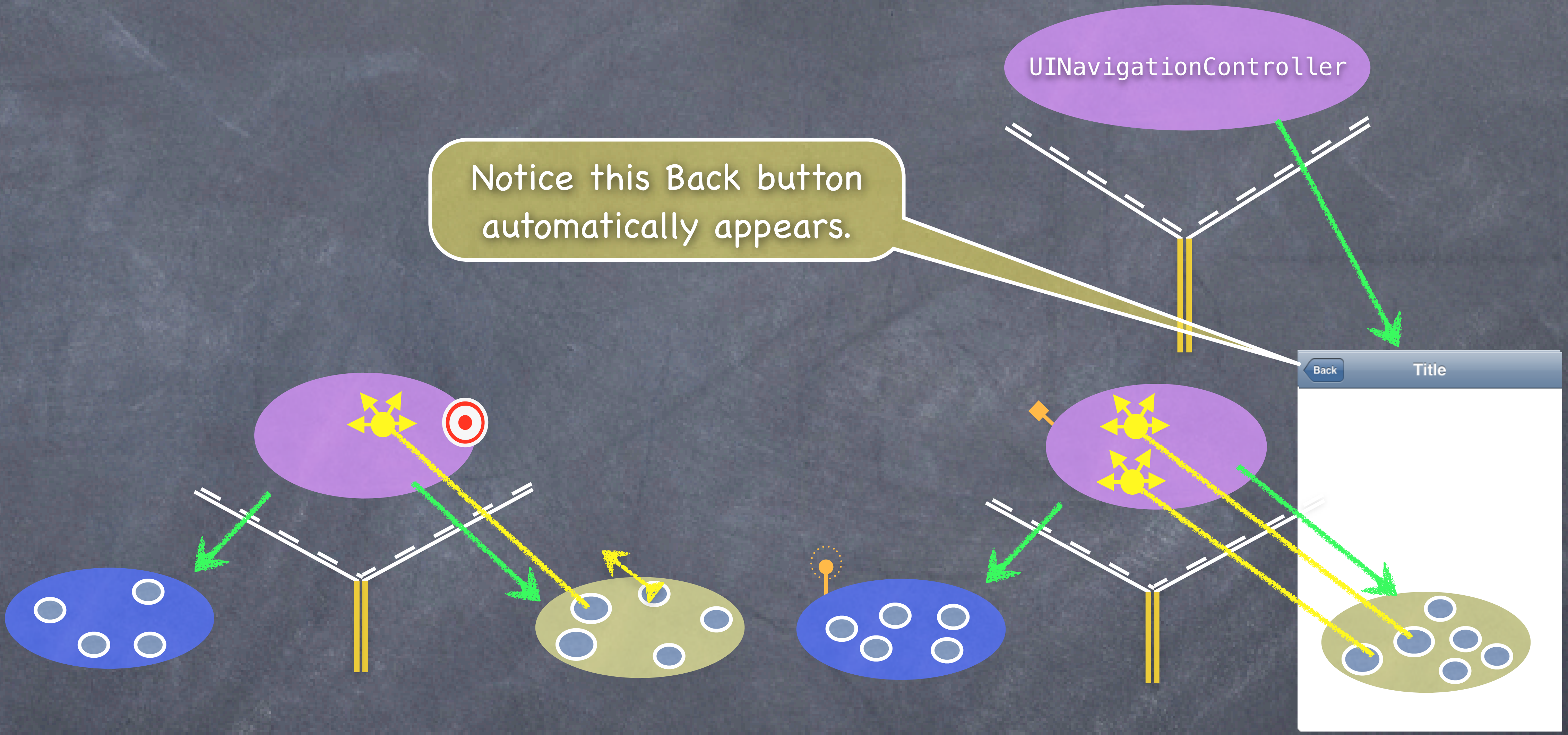


Then a UI element in this View (e.g. a `UIButton`) can segue to the other MVC and its View will now appear in the `UINavigationController`.

MVCs working together

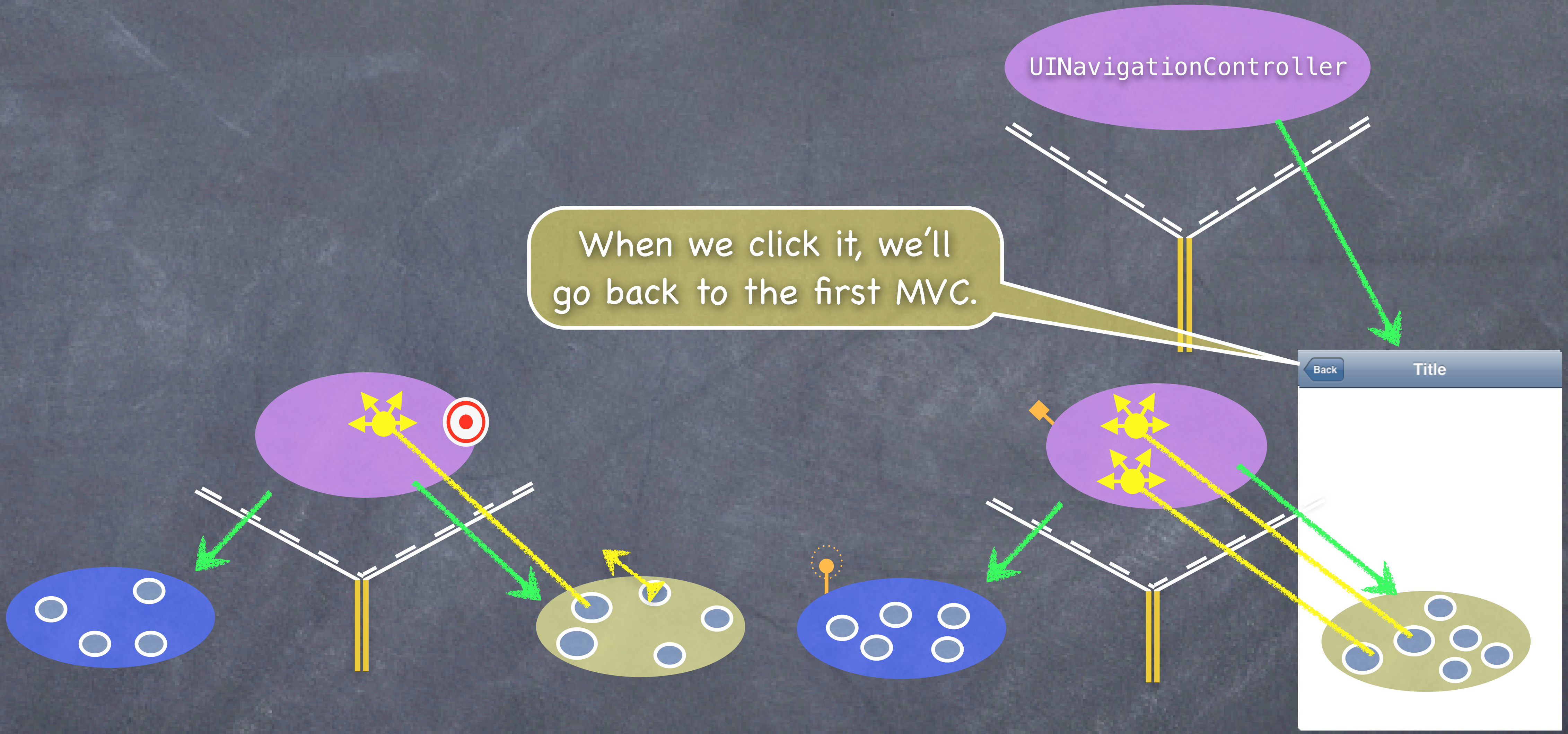


MVCs working together



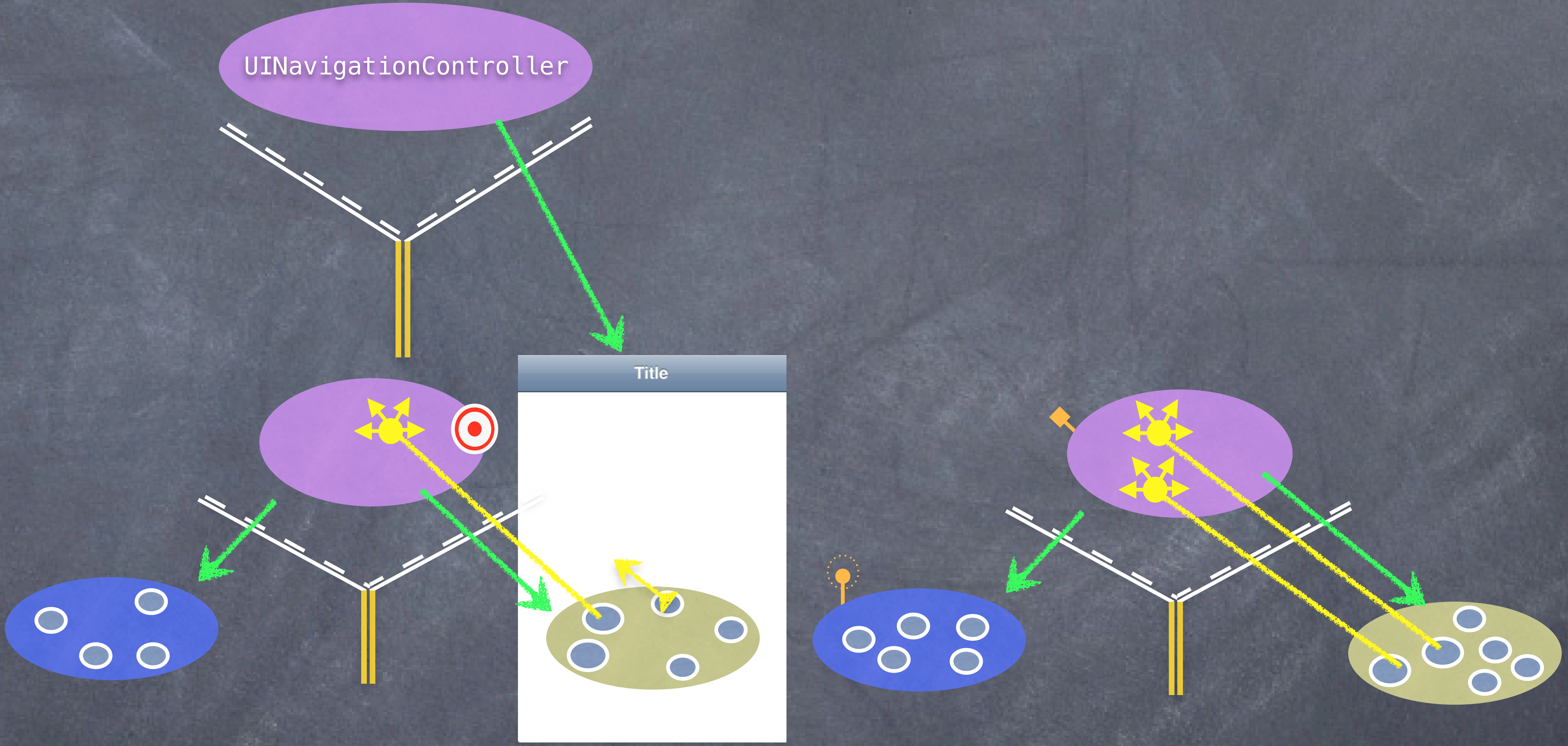
Notice this Back button automatically appears.

MVCs working together

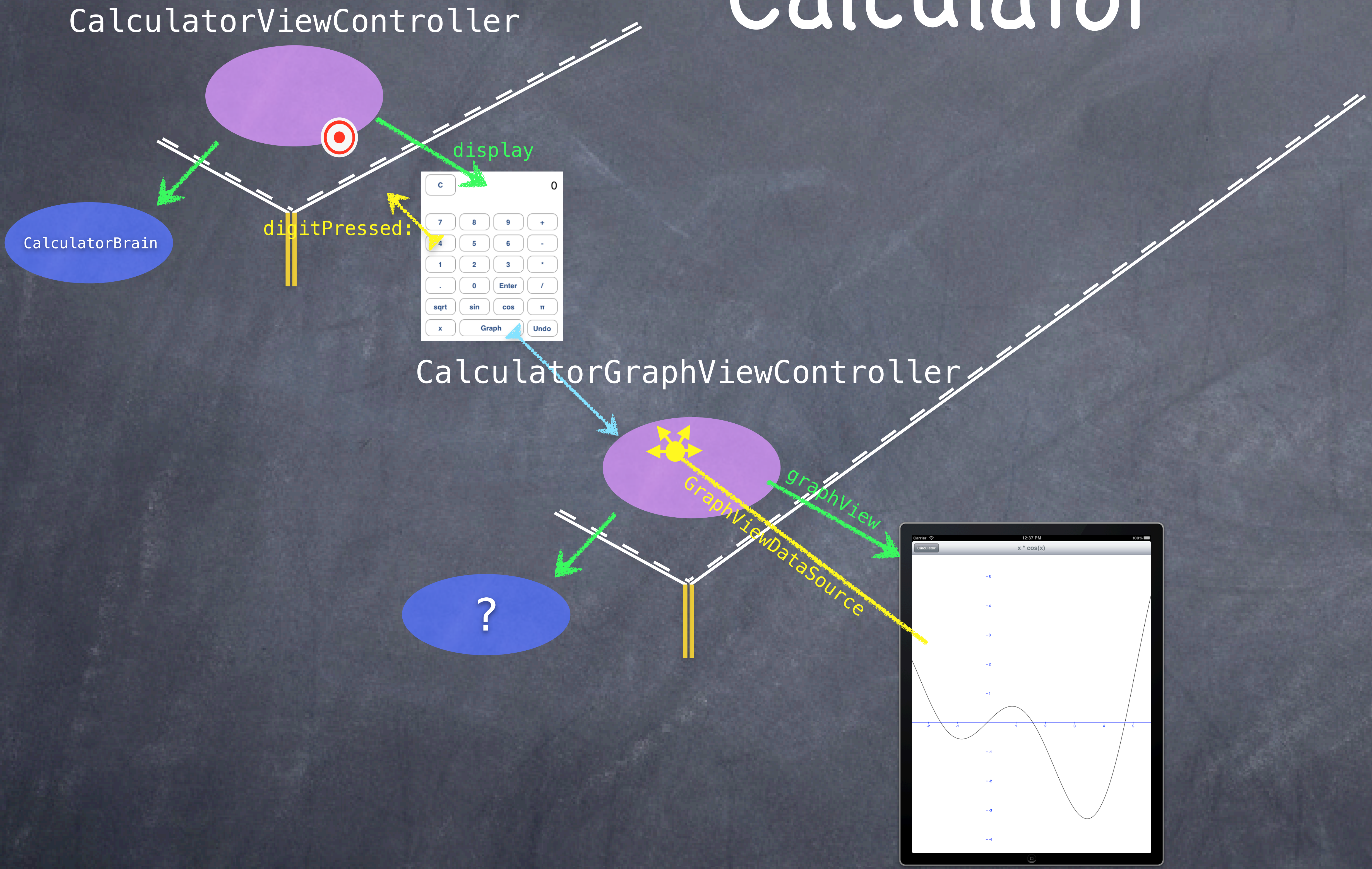


When we click it, we'll go back to the first MVC.

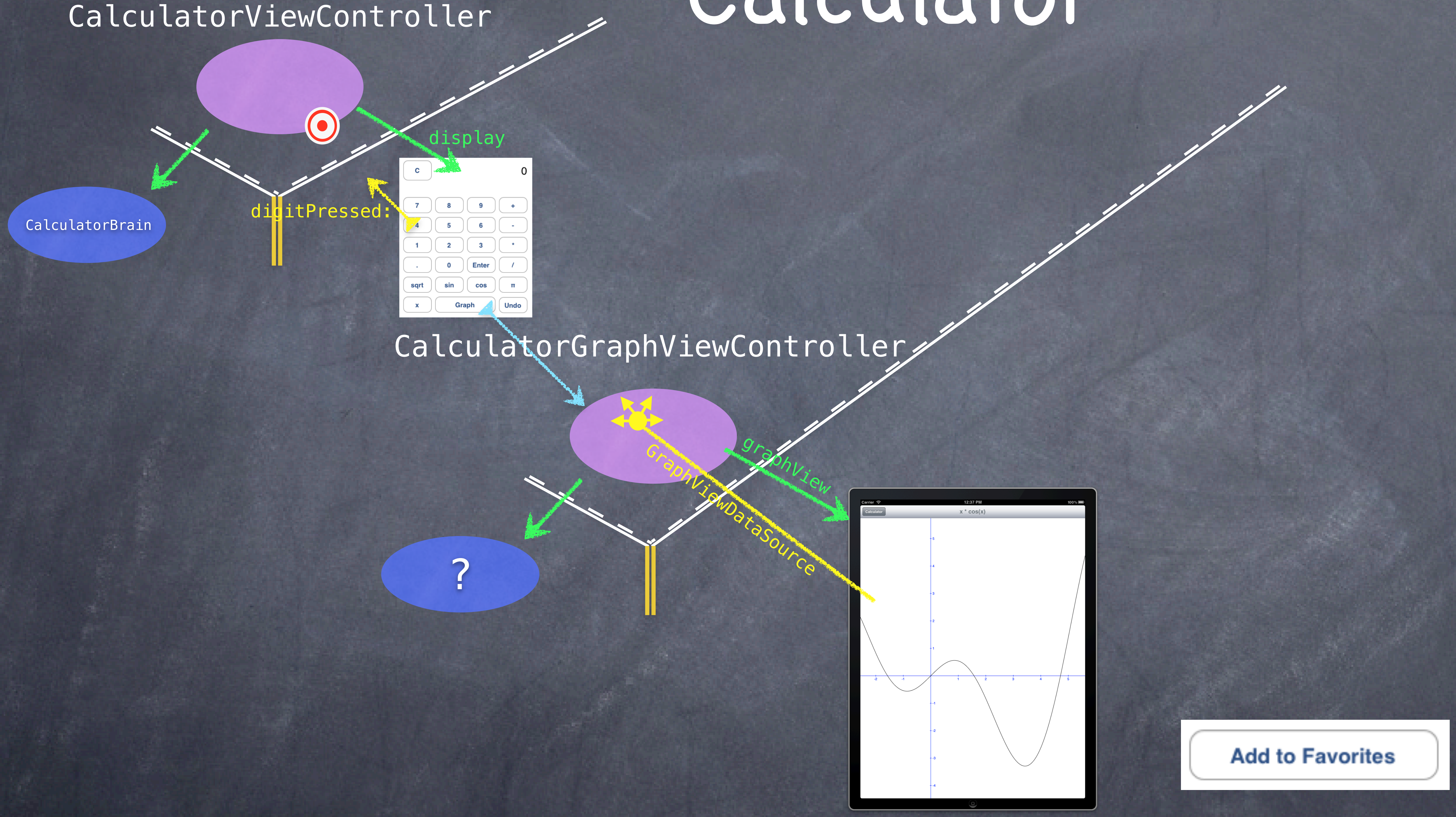
MVCs working together



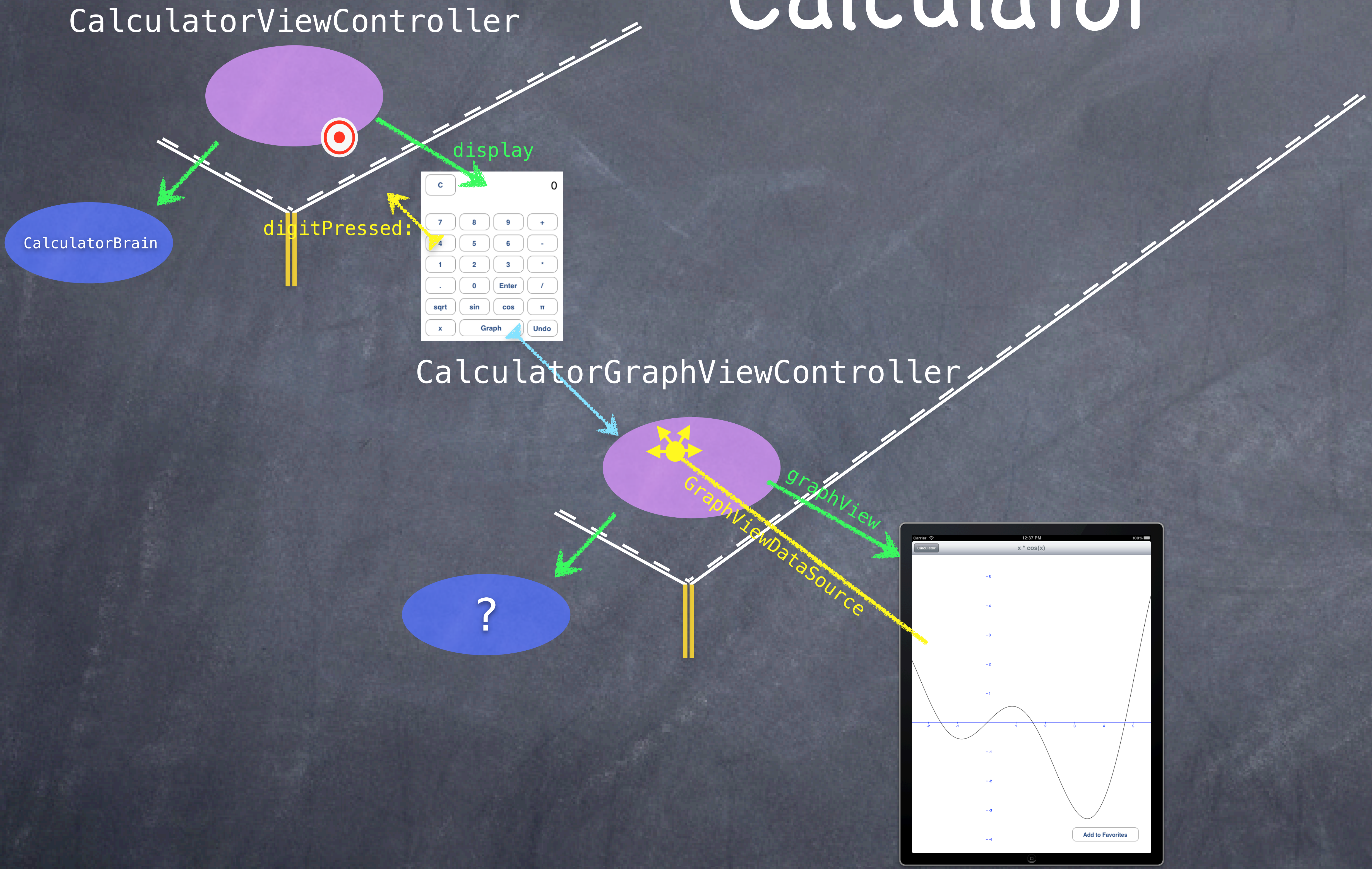
Calculator


CalculatorViewController
display
digitPressed:
CalculatorBrain
C
0
7 8 9 +
4 5 6 -
1 2 3 *
. 0 Enter /
sqrt sin cos π
x Graph Undo
CalculatorGraphViewController
graphView
GraphViewDataSource
?
Carrier
12:37 PM
100%
Calculator
x * cos(x)

Calculator

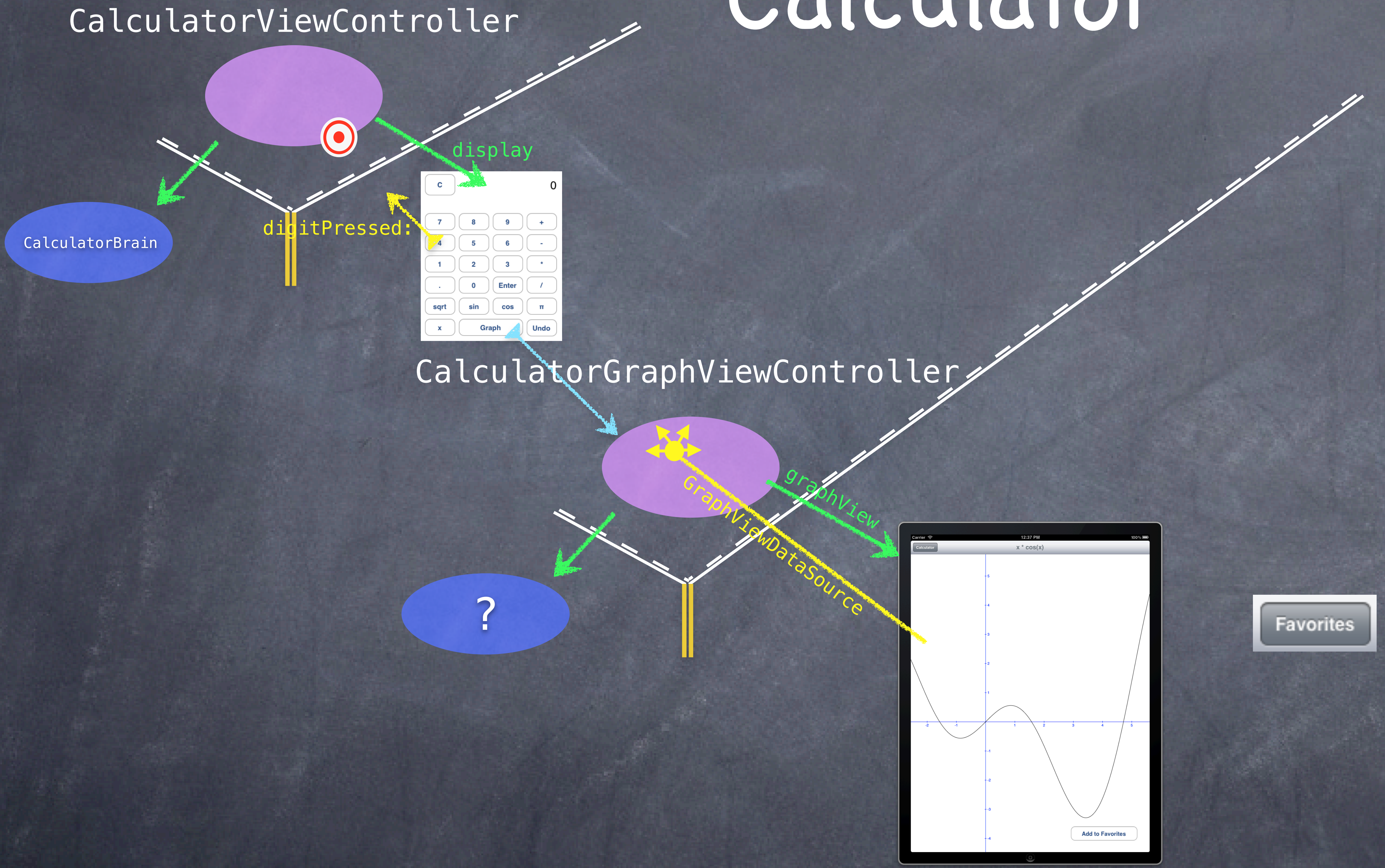

CalculatorViewController
CalculatorBrain
display
digitPressed:
C
0
7 8 9 +
4 5 6 -
1 2 3 *
. 0 Enter /
sqrt sin cos π
x Graph Undo
CalculatorGraphViewController
graphView
GraphViewDataSource
?
Calculator
x * cos(x)
Add to Favorites

Calculator

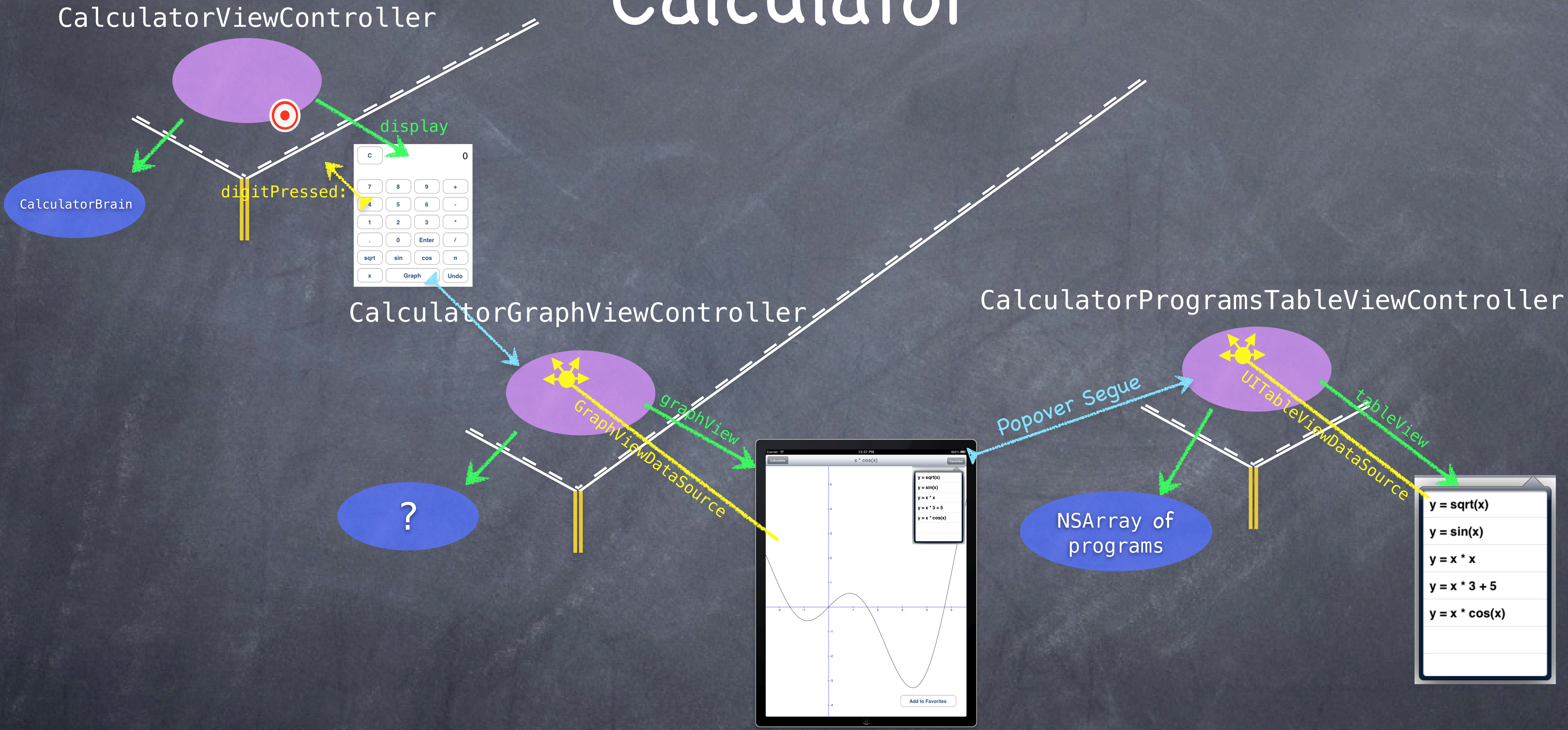


CalculatorViewController
display
digitPressed:
CalculatorBrain
C
0
7
8
9
+
4
5
6
-
1
2
3
*
.
0
Enter
/
sqrt
sin
cos
π
x
Graph
Undo
CalculatorGraphViewController
graphView
GraphViewDataSource
?
Calculator
x * cos(x)
Add to Favorites

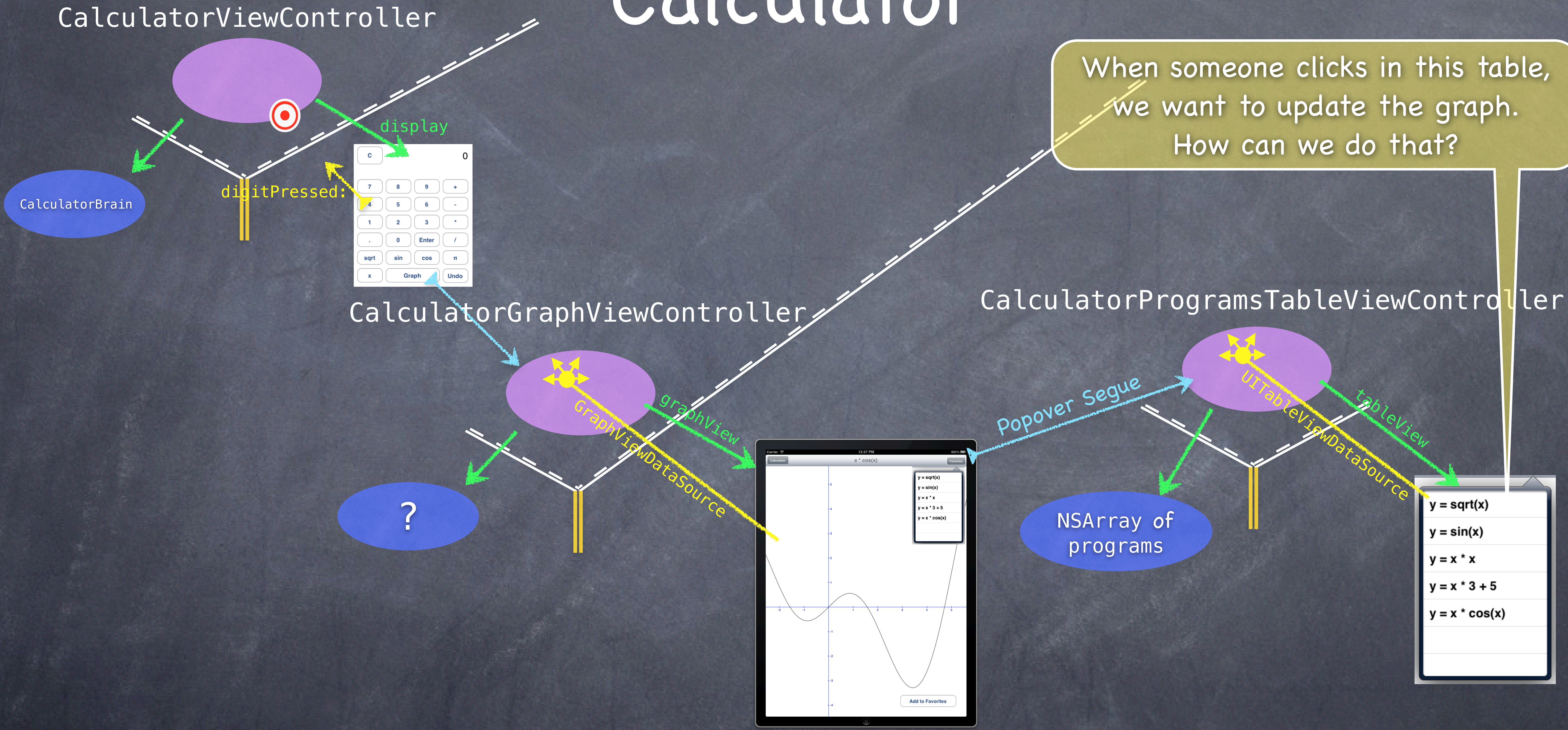
Calculator


CalculatorViewController
display
digitPressed:
CalculatorBrain
C
0
7
8
9
+
4
5
6
-
1
2
3
*
.
Enter
/
sqrt
sin
cos
π
x
Graph
Undo
CalculatorGraphViewController
graphView
GraphViewDataSource
?
Calculator
x * cos(x)
Add to Favorites
Favorites

Calculator


CalculatorViewController
display
digitPressed:
CalculatorBrain
C
0
7
8
9
+
4
5
6
-
1
2
3
*
.
Enter
/
sqrt
sin
cos
π
x
Graph
Undo
CalculatorGraphViewController
GraphViewDataSource
graphView
?
x * cos(x)
Calculator
Favorites
Add to Favorites
Popover Segue
CalculatorProgramsTableViewController
UITableViewDataSource
tableView
NSArray of programs
y = sqrt(x)
y = sin(x)
y = x * x
y = x * 3 + 5
y = x * cos(x)

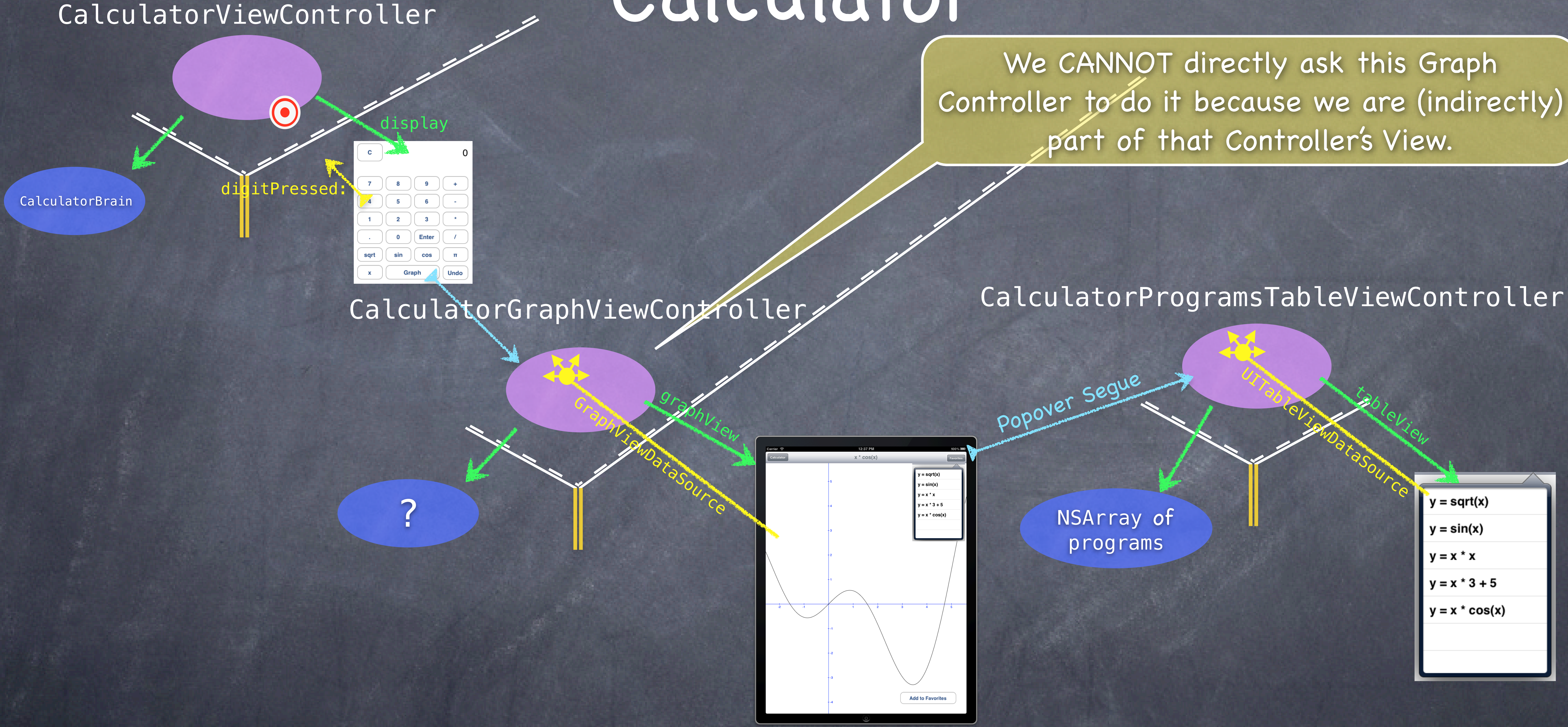
Calculator



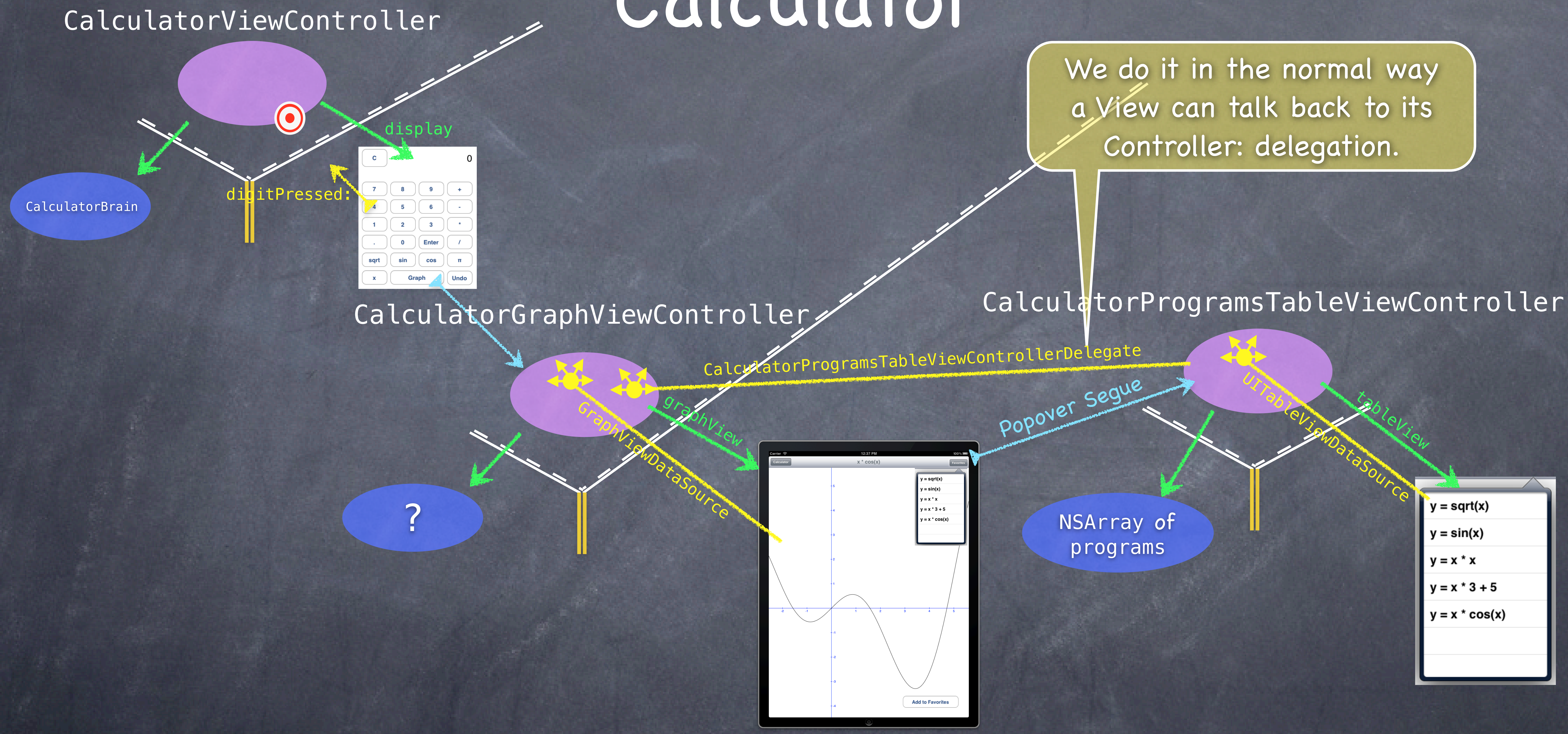
When someone clicks in this table, we want to update the graph. How can we do that?

Calculator

We CANNOT directly ask this Graph Controller to do it because we are (indirectly) part of that Controller's View.



Calculator



We do it in the normal way a View can talk back to its Controller: delegation.