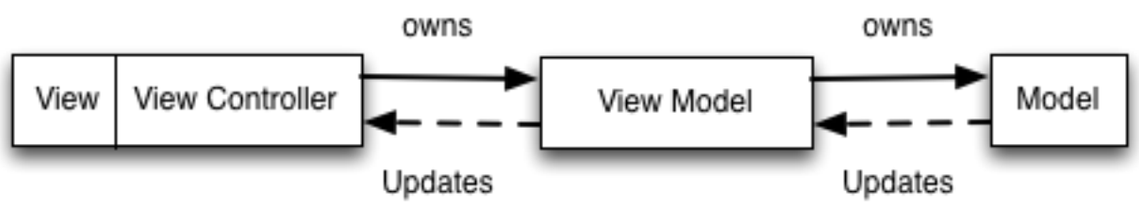
Model View Controller

陈灿 2016.1.5

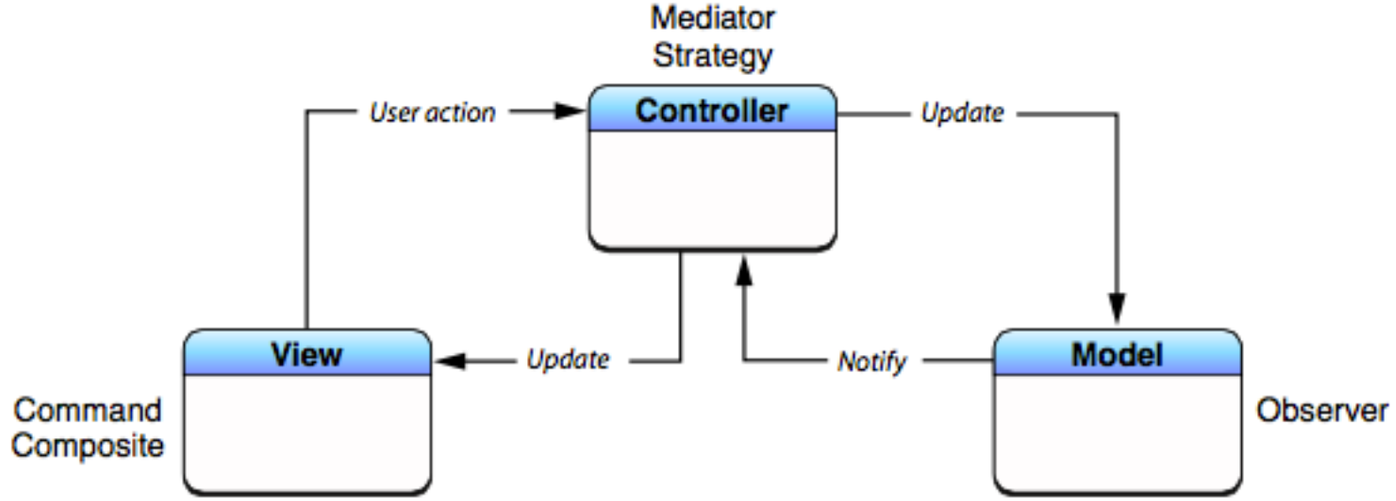
Why MVC ?

What’s UIViewController ?

How about MVVM ?



How MVC works together ?
How to communication ?
Relationship?



Model

Model Objects Encapsulate Data and Basic Behaviors

Model objects represent special knowledge and expertise. They hold an application's data and define the logic that manipulates that data. A well-designed MVC application has all its important data encapsulated in model objects. Any data that is part of the persistent state of the application (whether that persistent state is stored in files or databases) should reside in the model objects once the data is loaded into the application. Because they represent knowledge and expertise related to a specific problem domain, they tend to be reusable.

Ideally, a model object has no explicit connection to the user interface used to present and edit it. For example, if you have a model object that represents a person (say you are writing an address book), you might want to store a birthdate. That's a good thing to store in your Person model object. However, storing a date format string or other information on how that date is to be presented is probably better off somewhere else.

View

View Objects Present Information to the User
&
Pass User Order to Controller

A view object knows how to display, and might allow users to edit, the data from the application's model. ***The view should not be responsible for storing the data it is displaying***. (*This does not mean the view never actually stores data it's displaying, of course. A view can cache data or do similar tricks for performance reasons*). A view object can be in charge of displaying just one part of a model object, or a whole model object, or even many different model objects. Views come in many different varieties.

Controller

Controller Objects Tie the Model to the View

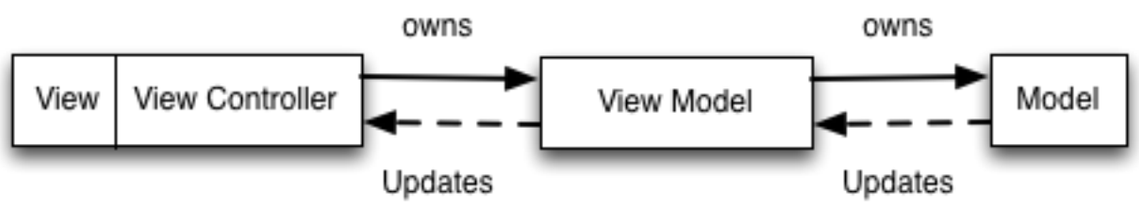
It represent business logic, ***It know how to transform user action to the modification on Model, and how to transform Model to View that user can read it***. A controller object acts as the intermediary between the application's view objects and its model objects.

Controller objects can be either reusable or non-reusable.

Why MVC ?

What’s UIViewController ?

How about MVVM ?



Low coupling
&
High cohesion

Reusable