CONTROLLER

Takes user input and figures out what it means to the model.

MODEL

The model holds all the data, state and application logic. The model is oblivious to the view and controller, although it provides an interface to manipulate and retrieve its state and it can send notifications of state changes to observers.

VIEW

Gives you a presentation of the model. The view usually gets the state and data it needs to display directly from the model.

Here's the creamy controller; it lives in the middle.

Controller

① The user did something

② Change your state

③ Change your display

```
class Player {
  play(){}
  rip(){}
  burn(){}
}
```

Model

④ I've changed!

View

⑤ I need your state information

This is the user interface.

Here's the model; it handles all application data and logic.