Model-View-Controller

Context

Interactive applications with a flexible human-computer interface.

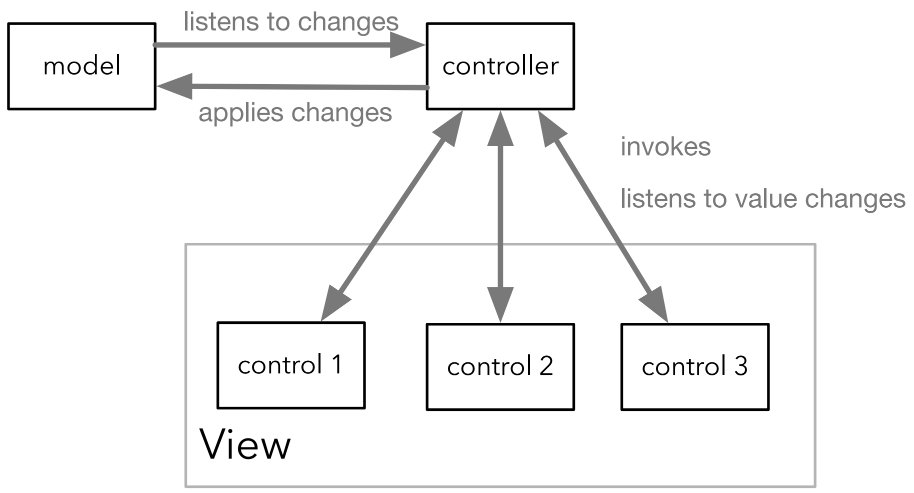
Discussion

- The *model* contains the logic about the application. It exposes attributes or properties that may be interdependent.
- The *view* consists of multiple user interface controls. Controls are isolated from each other and contain no logic. The view is not aware of the model.
- The *controller* bridges between the view and the model. It receives user interface events and pushes changes into the model. It receives model updates and pushes them into the controls.

The view should express user-interface concepts, not domain concepts.
The model should express domain concepts, not user-interface concepts.

Forces

- The same information appears in multiple controls.
- Controls should reflect changes immediately.
- Changes to the structure of the interface should be easy and may be done dynamically.
- Different 'look-and-feel' standards should not affect the logic of the application.



Question: Which frameworks correctly use the MVC pattern?
Which do not follow the pattern?