

Model-View-Controller

Separation of Concerns

- To make a change: must first figure out where to make the change
- What do you need to know?
- Understand the whole system? Impossible!
- Approach: divide the system into pieces

Responsibilities

- A responsibility is an action, knowledge to be maintained, or a decision to be carried out by a software system or an element of that system. [Bachmann, Bass, Nord]
- Responsibilities are assigned to modules
- But what is the cost of modifying a responsibility?
- Responsibilities can be coupled: a modification to one can result in a modification to the other

Diffuse Responsibilities

- It might be hard to see in a program this small
- But why are calculations for the two sections of the progress bar in the event handler?

```
button.on('click', () => {  
    progress.advance();  
    completedProgressRect.width((progress.amount / progress.max) * stage.width());  
    incompleteProgressRect.x((progress.amount / progress.max) * stage.width());  
    layer.draw();  
    console.log(progress.amount);  
});
```

- Also redundant with initialization

Cost of Changes

- To change the visual layout, also need to change:
 - Event handler
 - Top-level initialization
- Behavior and view are *improperly coupled*.

Coupling

- Cost of modifying module A depends on how tightly-coupled it is to other modules
- Idea: reducing coupling may reduce modification costs
- To reduce coupling:
 - Minimize relationships among elements *not* in the same module
 - Maximize relationships among elements in the same module

Cohesion

- Put related responsibilities in the same module
- To maximize modifiability, maximize cohesion & minimize coupling

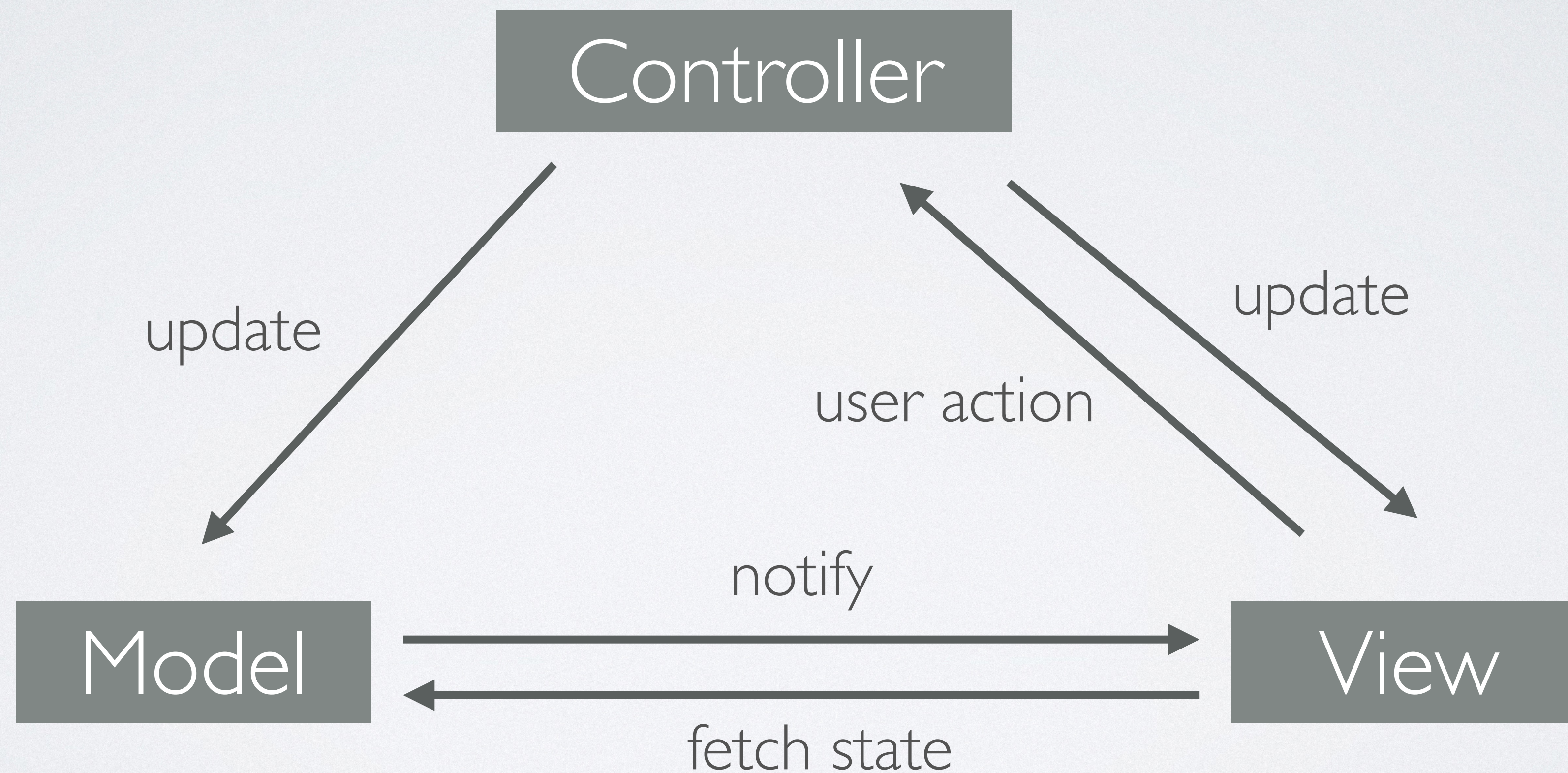
Need a Plan

- Model-View-Controller is classic (SmallTalk-80)
- Other variants: Model-View-Presenter, Model-ViewController
- Basic principle: separate *behavior* from *model* and *view*

Model-View-Controller (MVC)

- Model: represents the state that might be updated
 - In our example: how much progress
 - Could be very complicated (in a spreadsheet, this is the document)
- View: translates the model to pixels
- Controller: implements interaction ("business logic")

Classic Model-View-Controller



Let's Refactor