Lecture 21
MVC

- **Model**
  - Encapsulates application state
  - Responds to state queries
  - Exposes application functionality
  - Notifies views of changes

- **View**
  - Renders the models
  - Requests updates from models
  - Sends user gestures to controller
  - Allows controller to select view

- **Controller**
  - Defines application behavior
  - Maps user actions to model updates
  - Selects view for response
  - One for each functionality

**State Query**

**View Selection**

**User Gestures**

**Method Invocations**

**Events**
State Diagram

• Controller can be modeled as a state diagram.

• States
  – Think of “state of the world”
  – Characterized by “how things are”
    • Because this is a “model”, we just want to know about how things that matter are.
  – Define a specific state by a set of assertions
    • Truth statements about the program
States Graphically

- canvas 1 picture here
State Transitions

• Something must trigger a change in state.
  – Intuitively: if nothing happens, then the state of the world remains the same.

• State diagram must account for each possible state changing interaction.
  – Given the state you are in, must identify all possible events that cause a change in state.
    • Identify any actions and or side-effects associated with the event.
      – In MVC parlance: interactions with model and view.
    • Identify the new state.
      – Can be the same as the old state.
State Transitions Graphically

• canvas 2 picture here
Diagram Notes

• In interest of making diagram easier to read:
  – May hide assertion description and just use state name to label states.
  – May hide actions and side-effect descriptions and just use event name to label state transitions.
What Makes Two States Different

• Either of following conditions
  – Assertions associated with the two states are different.
  – Effect of possible events are different
    • In other words, state transitions are different
    • In our calculator example: if effect of pushing a button is not the same, then must be in a different state.
State Diagram Completeness

• A complete state diagram is *complete* if every transition from every state is accounted for.
  – Think of it as a game to account for every possible sequence in the fewest states.
Calculator Initial State

• What’s the calculator’s initial state?
  – What assertions can we make about the view and model of the calculator.
    • Init only canvas
Other States

• States Only Canvas
State Transitions

• From our initial state, what events might occur?
  – Press a button
    • 0-9, +/-, decimal point, C, CE, operation (+,-,*,/)

• What are the effects of these buttons?
  – 0, +/-, C, CE, =
    • Does nothing. All the assertions remain true and effect of all buttons remains the same.
  – 1-9
    • Display changed to match digit.
    • Effect of other buttons changes (e.g., 0)
  – +,-,*,/
    • Operation is known
  – .
    • Display changes to “0.”
    • Effect of other buttons changes (e.g., 0 and .)

• Init Canvas
Are we complete?

• No
• Have to account for all events possible for all states.
• Go through rest of states.
Calculator v2

- Code controller to match state diagram.
- Compare to first version of code.