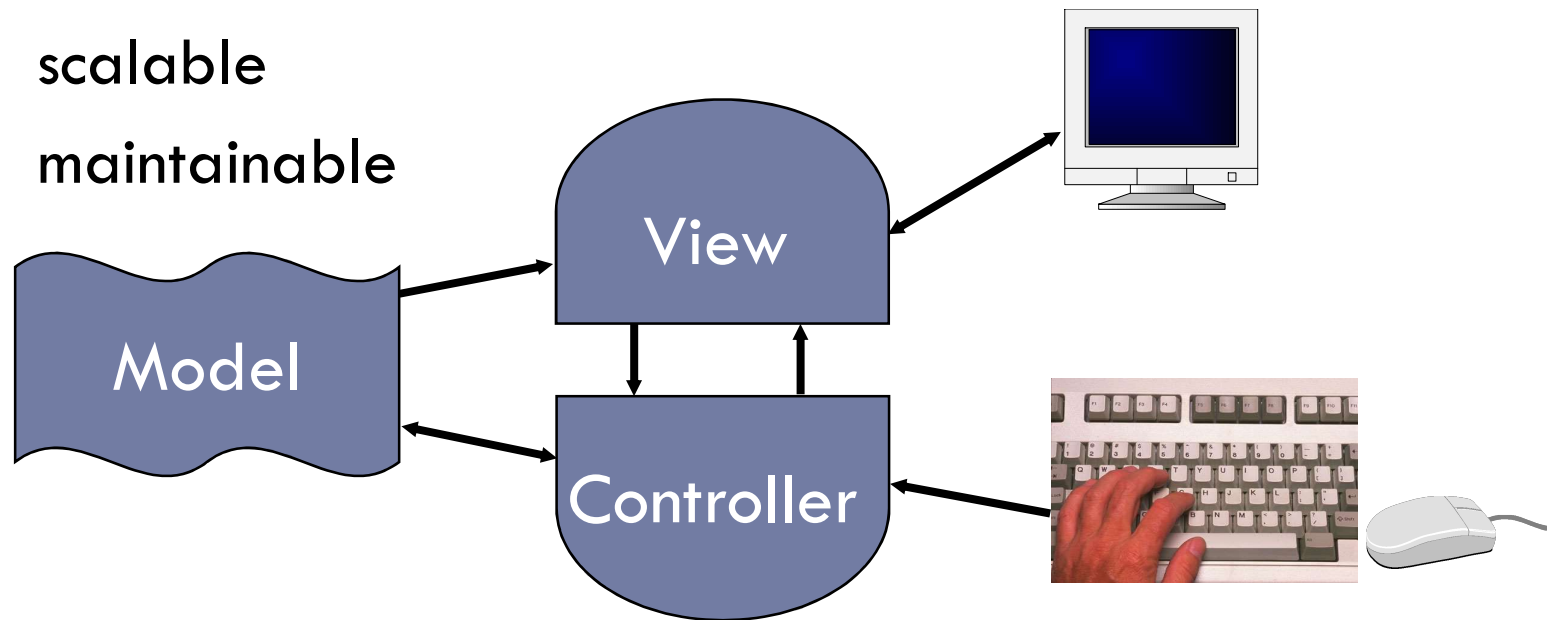# MVC ISSUES AND DETAILS

CMPT 381

Model-View-Controller

- An architecture for interactive applications
  - introduced by Smalltalk developers at PARC

- Partitions application so that it is:
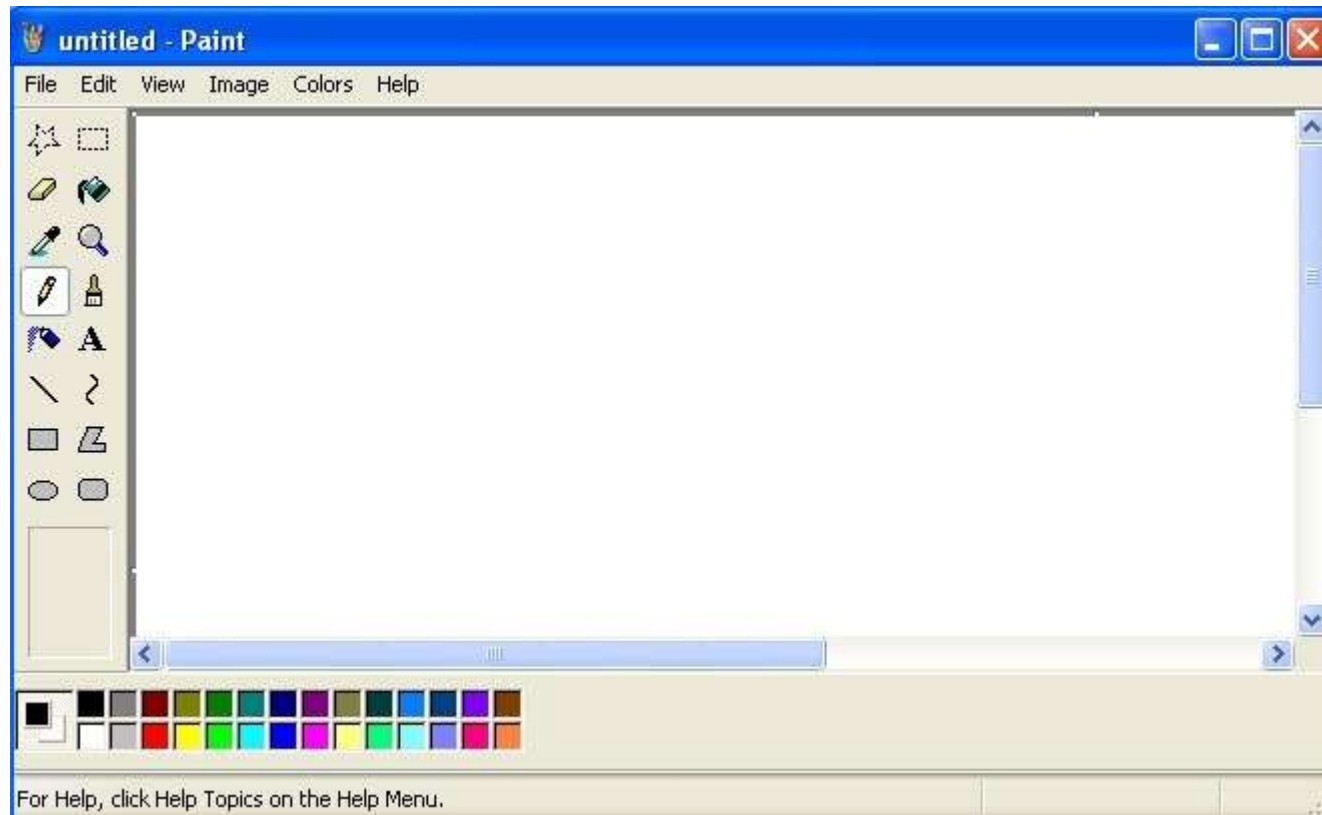
  - scalable
  - maintainable

Overview

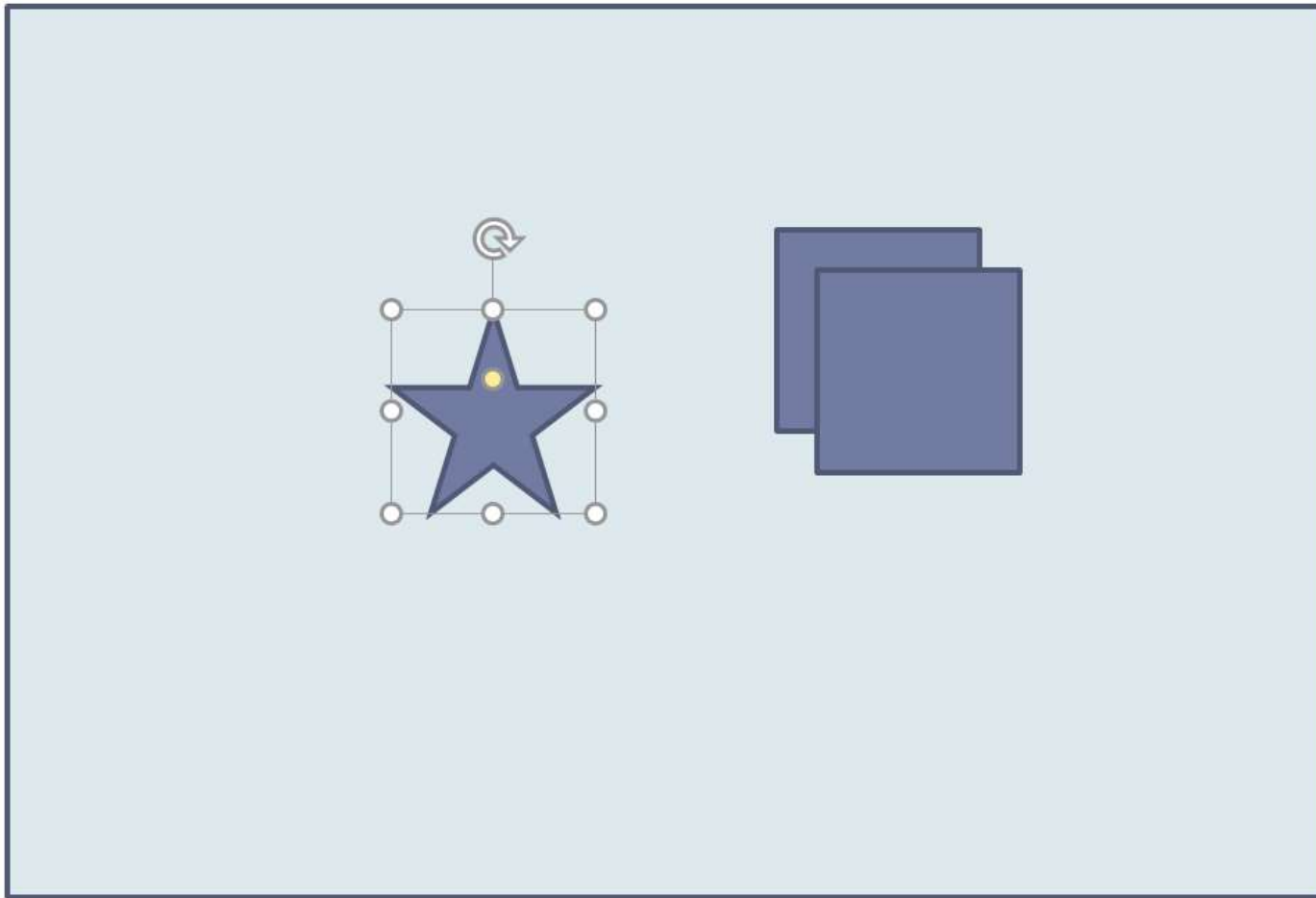View state

Interaction models

Coordination between views

Storing view state

Storing view state

- Some UI actions are direct commands
  - Menu → "Clear All" → model.clearAll()
  - Nothing needs to be stored for these interactions
- Other actions are persistent
  - Involve changes to the state of the view
    - Selecting a current tool or colour
    - Changing the scroll region
  - Where should this information be stored?
    - Model? View? Controller?

Storing view state – selection
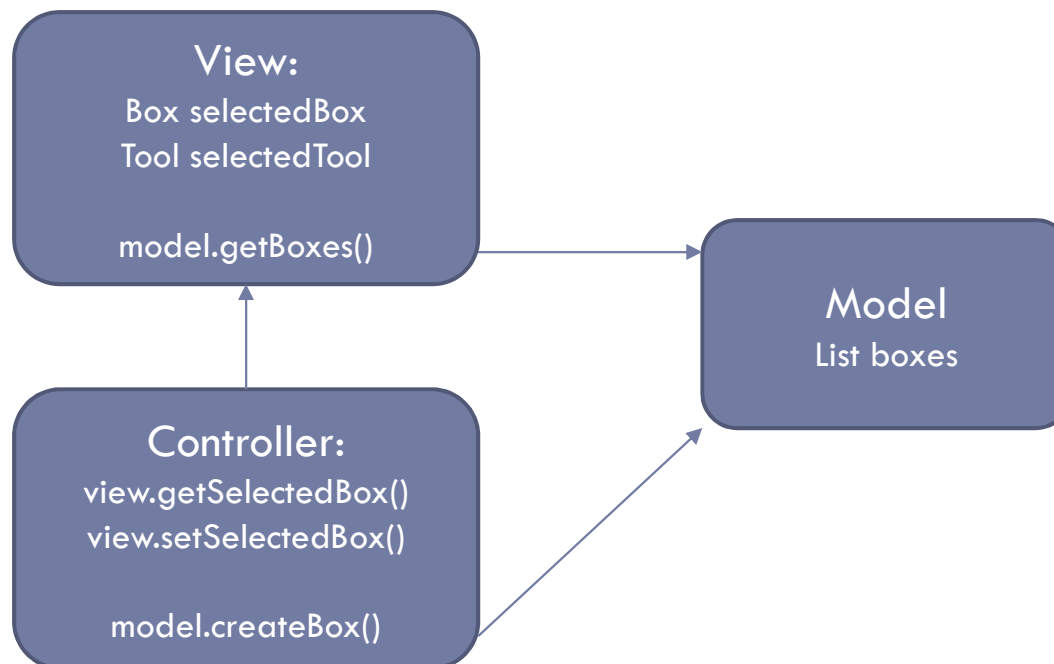
Who needs to know about view state?

- Selection:
  - View displays selected object differently
  - Controller needs to know which object to act on
    - E.g., when delete key pressed, what gets deleted?
- Tool (mode):
  - View: specific cursor, highlighting in toolbar
  - Controller: what happens on action
    - E.g., when mouse pressed, draw or erase?
- Viewport
  - View: what region of the workspace to draw
  - Controller: handle scroll events
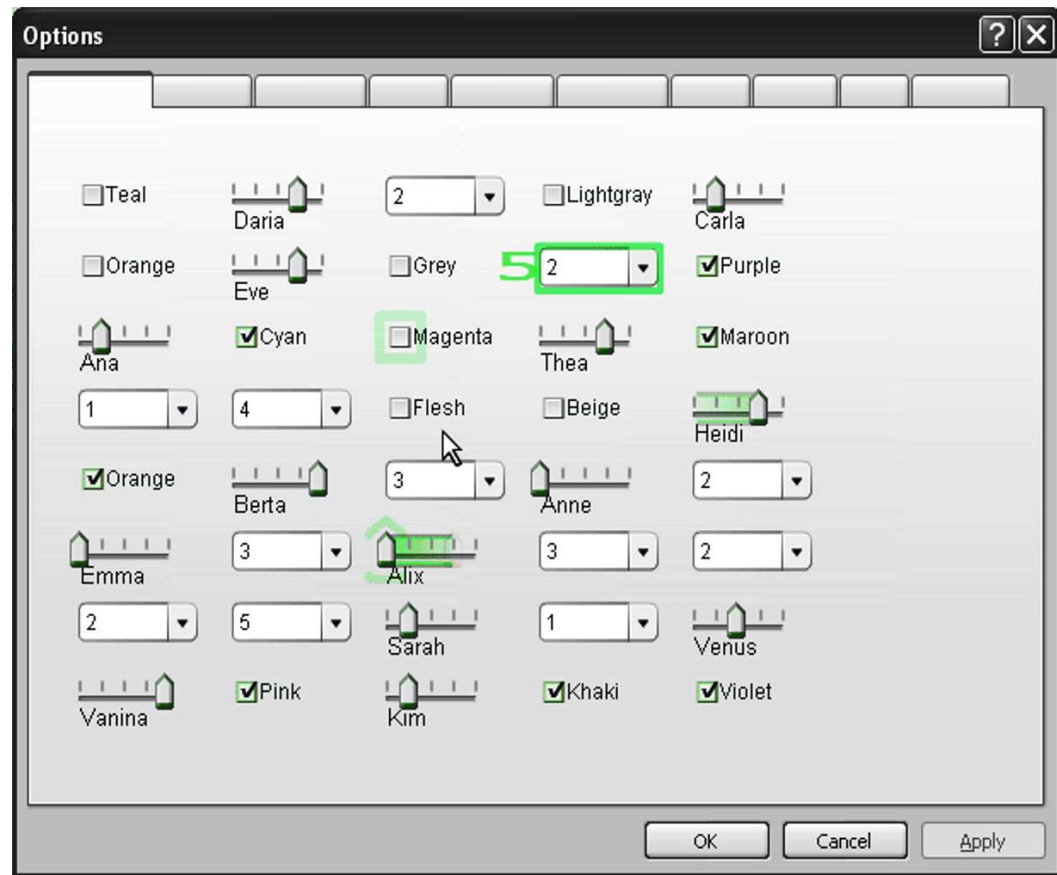
Where to store view state?

- If only a few pieces of information:
  - Store in View, make visible in Controller

Where to store view state?

- If only a few pieces of information:
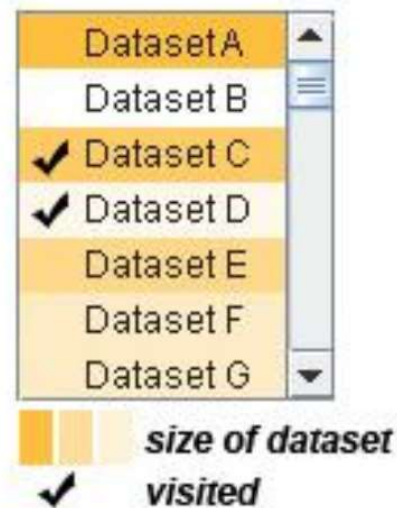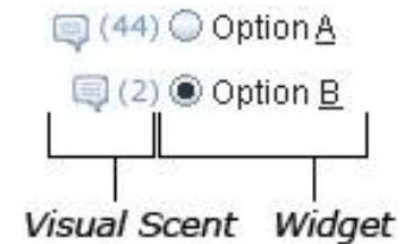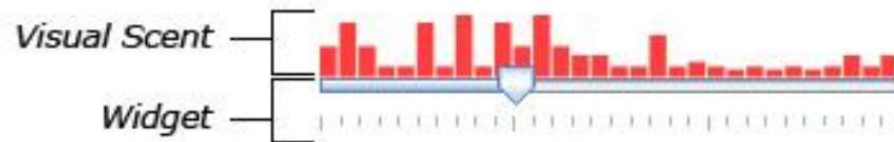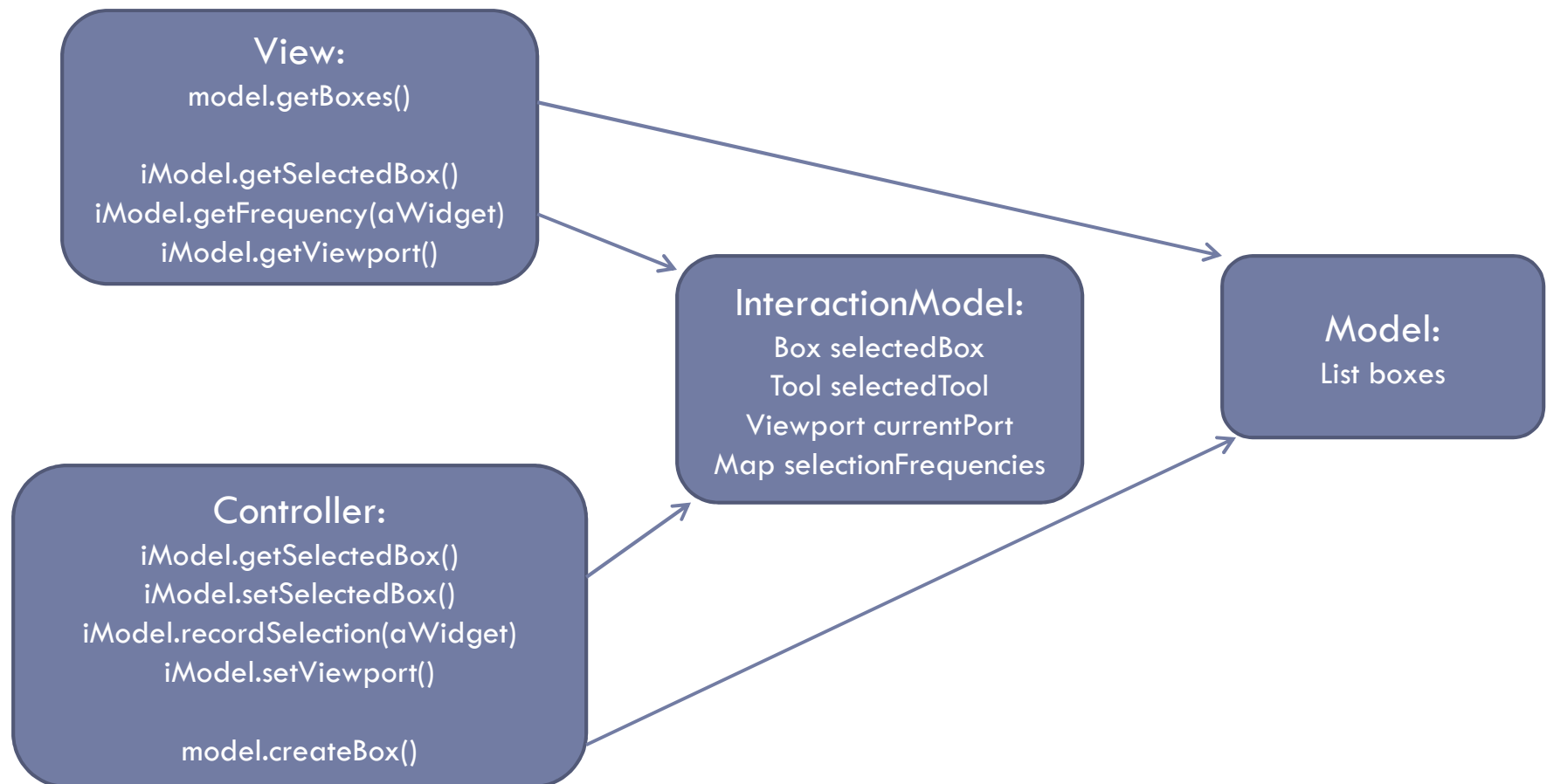 - Store in View, make visible in Controller

View:
Box selectedBox
Tool selectedTool

model.getBoxes()

Model
List boxes

Controller:
view.getSelectedBox()
view.setSelectedBox()

model.createBox()

More complex view state

- Phosphor widgets (Baudisch)

More complex view state

- "Scented widgets" (Willet, Heer, Agrawala)

Where to store complex view state?

View:
model.getBoxes()

iModel.getSelectedBox()
iModel.getFrequency(aWidget)
iModel.getViewport()

InteractionModel:
Box selectedBox
Tool selectedTool
Viewport currentPort
Map selectionFrequencies

Model:
List boxes

Controller:
iModel.getSelectedBox()
iModel.setSelectedBox()
iModel.recordSelection(aWidget)
iModel.setViewport()

model.createBox()

Synchronization between views

Synchronization between views

Synchronization between views

How to handle this?

- Store in View
  - Views need to inform other Views of selection changes
- Store in InteractionModel
  - Works well for selection
    - (assuming one selection across all views)
- What about View-specific information?
  - E.g., viewport, cursor location
  - Can register View with InteractionModel
  - Valuable for groupware applications

Multiple distributed views in groupware