

# CS160 Discussion Section

Matthew Kam  
Apr 7, 2003



## Outline

- Design Patterns
- Model-View-Controller
- Heuristic Evaluation Assignment
- Administrivia

## Motivation for Design Patterns

- Most examples from UI literature are critiques
  - Norman, Nielsen, etc.
- Design is about finding solutions
- Unfortunately, designers often reinvent
  - hard to know how things were done before
  - hard to reuse specific solutions
- Design patterns are a solution
  - reuse existing knowledge of what works well



## Pattern Format

1. Pattern Title
2. Background Information
3. Problem Statement
4. Solution
5. Solution Sketch
6. Other Patterns to Consider

## Home Page Design Rules

- Problem
  - without a compelling home page (H/P), no one will ever go on to the rest of your site
  - surveys show millions of visitors leave after H/P
    - most will never come back -> lost sales, etc.



## Six Ways to Make a Good Home Page



- Lure visitors to return
  - with fresh content
    - keep it updated so there is a reason to come back
  - by seducing with text
    - you have only seconds
      - lively, sparkling, precise

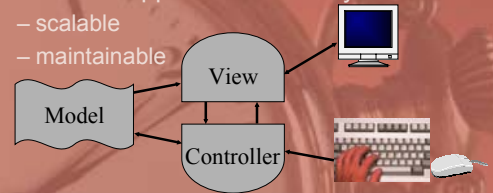
## Design Patterns

- Next used in software engineering [Gamma, et. al.]
  - communicate design problems & solutions
- Proxy
  - surrogate for another object to control access to it
- Observer
  - when one object changes state, its dependents are notified

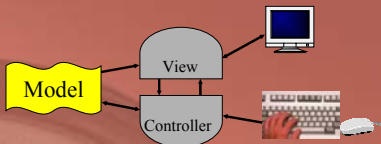


## Model-View-Controller

- Architecture for interactive apps
  - introduced by Smalltalk developers at PARC
- Partitions application in a way that is
  - scalable
  - maintainable

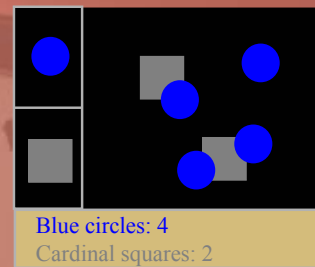


## Model



- Information the app is trying to manipulate
- Representation of real world objects
  - circuit for a CAD program
    - logic gates and wires connecting them
  - shapes in a drawing program
    - geometry and color

## Example Application



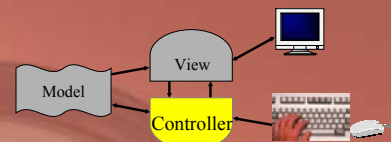
## Example Code (Model)

```
class Model {
    class Shape {
        ...
        int type; // 0 - square, 1 - circle
        int x, y;
        Color color;
        Shape(int type, int x, int y, Color c) {...};
    }

    Shape shapes[MAX_SHAPES]; // array of shapes
    View views[MAX_VIEWS]; // array of Views

    void addCircle(Shape circle) {
        shapes.addElement(circle);
        for each view do
            view.refresh();
    }
}
```

## Controller



- Receives all input events from the user
- Decides what they mean and what to do
  - communicates with view to determine which objects are being manipulated (e.g., selection)
  - calls model methods to make changes on objects
    - model makes change and notifies views to update

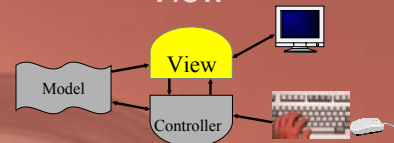
## Example Code (Controller)

```
// declaration in corresponding Model
class Shape {
    ...
    int type; // 0 - square, 1 - circle
    int x, y;
    Color color;
    Shape(int type, int x, int y, Color c);
}

// code in corresponding Model
void addCircle(Shape circle) {
    ...
}

// Controller code
void onMouseClick(MouseEvent e) {
    addCircle(
        new Shape(Shape.CIRCLE, e.getX(),
                 e.getY(), Color.BLUE));
}
```

## View



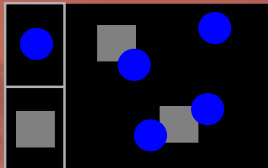
- Implements a visual display of the model
- May have multiple views
  - e.g., shape view and numerical view
- Any time the model is changed, each view must be notified so that it can change *later*
  - e.g., adding a new shape

## Example Code (View)

```
// code in corresponding model
void addCircle(Shape circle) {
    shapes.addElement(circle);
    for each view do
        view.refresh();
}

// for graphical View
void refresh() {
    for each shape do
        shape.repaint();
}

// for text View
void refresh() {
    print("Blue circles:" + shapes.count(Shape.CIRCLE);
    print("Cardinal squares:" + shapes.count(Shape.SQUARE);
}
```

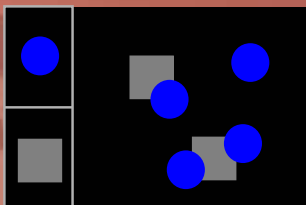


Blue circles: 4  
Cardinal squares: 2

## Why MVC?

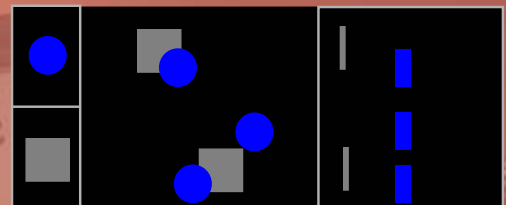
- Combining MVC into one class or using global variables will not scale
  - model may have more than one view
    - each is different and needs update when model changes
- Separation eases maintenance
  - easy to add a new view later
  - new model info may be needed, but old views still work
  - can change a view later, e.g., draw shapes in 3-d (recall, view handles selection)

## Multiple Views



Blue circles: 4  
Cardinal squares: 2

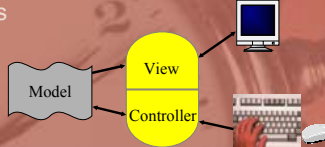
## Adding Views Later



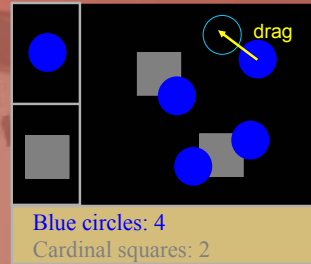
Blue circles: 4  
Cardinal squares: 2

## Combining View & Controller

- View and controller are tightly intertwined
  - lots of communication between the two
- Almost always occur in pairs
  - i.e., for each view, need a separate controller
- Many architectures combine into a single class

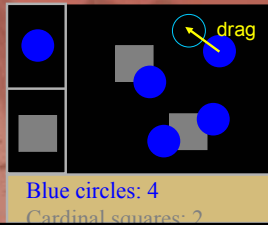


## Combining V-C for Efficiency



## Example Code

```
void onMouseDragged(MouseEvent e) {  
    if (view1.inDraggedMode() {  
        view1.getDraggedObject.setX(e.getX());  
        view1.getDraggedObject.setY(e.getY());  
        view1.refresh();  
    }  
}
```



## Back to Patterns

- What are the problems that MVC attempts to solve?
- What is the overall solution in the MVC approach?

## HE Assignment

- Report outline
  - Problem (one sentence)
  - Violations found (as needed)
  - Summary of violations (¼ page)
    - Total # of violations in user interface
    - # of violations for each heuristic
    - # of violations for each severity rating
  - Overall recommendations (¼ page)

## Grading Criteria

- Completeness
  - Does HE cover entire interface?
    - Indicate clearly if no violation found on particular screens
  - HE must include unimplemented portions
    - Indicate clearly if no sketches given for unimplemented portions
    - Explain if unimplemented portions could not be evaluated
  - Were there *major* violations that were not found?
    - That's why HE needs 3-5 evaluators

## Grading Criteria (Cont'd)

- Clarity of violation descriptions
  - Are the stated violations valid?
  - Limit to 30 *major* violations if >30 found
  - Report multiple instances as single violation
- Quality of recommendations
  - Usually, one recommendation per heuristic
    - Unless each heuristic's violation fall under >1 category
  - For each heuristic, give recommendation for violation with highest severity

## Administrivia

- Heuristic evaluation assignment due Apr 9, 2003 (this Wed) in lecture
  - Turn in two copies
  - Post online on Swiki for respective groups to start making hi-fi changes

## Administrivia

- Scribe notes from HCI Panel posted on section homepage under "Announcements"
  - Skills needed for HCI careers
  - Career opportunities in HCI
  - Workplace-related issues
- Ethics involving human subjects
  - Anonymize online reports if haven't already done so