Human-Computer Interaction IS4300

1

I5 – painting app *due*

- Your mission in this exercise is to implement a very simple Java painting application. The app must support the following functions:
- Draw curves, specified by a mouse drag.
- Draw filled rectangles or ovals, specified by a mouse drag (don't worry about dynamically drawing the shape during the drag - just draw the final shape indicated).
- Shape selection (line, rectangle or oval) selected by a combo box OR menu.
- Color selection using radio buttons OR menu.
- Line thickness using a combo box OR menu.
- A CLEAR button.

3

P4 – Design Sketches
Due next class

- **Interaction Scenarios**
  - Expand each of your activity design scenarios (3+) into full interaction scenarios, thinking about what the user perceives and the actions he/she performs at each major step in the scenario.
- **Design Options**
  - Three options for your most important window or dialog box, and brief rationale for why you selected one over the other two.
- **Preliminary interface design**.
  - One or more sketched windows or dialog boxes, along with the menus and controls that the user manipulates.
- **Storyboards**.
  - For each of your tasks/scenarios, describe how your preliminary interface would be used to perform the task. Use rough sketches to illustrate how the interface would look at important points in the task.

4

Evaluation

Why evaluate?

What needs to be evaluated?

How do we evaluate?

UI Evaluation Why?

- To measure something
  - e.g., usability metrics (learnability etc)
  - To compare metrics to stated criteria
- To compare two or more designs
  - To compare metrics or qualitative feedback from two or more alternatives
- To identify specific design problems that need to be fixed

6

Kinds of Evaluation

- Formative
- Summative

7

UI Evaluation Methods

- Expert/Inspection methods
  - Heuristic evaluation
  - Cognitive walk-through
- User Testing
  - Qualitative methods (interviews, questionnaires, think aloud)
    - ethnography; focus groups
  - Quantitative methods
    - Descriptive studies
    - Experiments (same environment & task with 2 or more alternative designs)
- Modeling

Expert Inspection methods

- Heuristic evaluation
  - Checklist approach
  - Independent evaluators (3-5)
  - Severity rating for problems

    0. No problem
    1. Cosmetic problem
    2. Minor – low priority
    3. Major problem – high priority
    4. Catastrophe – must fix

Problems with Heuristic Eval

- Poor agreement among experts in rating severity of problems.
- You should not evaluate your own designs
  - Difficult to ignore your knowledge of how the system works, the meaning of icons or menu names and so on, and you are likely to give the design the 'benefit of the doubt' or to find obscure flaws which few users will ever happen upon.
- Woolrych and Cockton (2000) conducted a large-scale trial of heuristic evaluation.
  - Many issues identified by the experts were not experienced by people, while some severe difficulties were missed by the inspection against heuristics.
- Conclusion: Use heuristic eval for formative testing only

10

Expert Inspection methods

- Cognitive walkthrough
  - Walk through each step in the task and evaluate:
    1. Is the effect of the action the same as the user's goal at that point?
    2. Will users see that the action is available?
    3. Once users have found the action, will they know it is the one they need?
    4. After the action is taken, will users understand the feedback they get?

Model-based evaluation

- Build and evaluate a formal model
- Simulations of User, System, World
- e.g. GOMS
  - Goals, Operators, Menus, Selection
- e.g. KLM
  - Keyboard Level Model
- How are formal methods used in evaluation?
- Shortcomings?

User testing

- Who?
- Setting?
- Pros / Cons?

13

User testing methods

- Think aloud
- Focus groups
- Usability assessments
- Controlled experiments

Some Terminology

- Reliability vs. Validity
- Ecological Validity
- Within-subjects vs. Between-subjects
- Pilot test

Evaluation Strategy

- What Am I Evaluating? (prototype or ?)
- What Constraints Do I Have?
  - Money
  - Time
  - Availability of usability equipment
  - Availability of participants and the costs of recruiting them
  - Availability of evaluators
- Documenting the Evaluation Strategy

Problem

- Mary has just designed a web site for her daughter's girl scout troup, allowing the public to order cookies. She'd like the site to be as usable as possible. She has no budget, and no access to users, but does have a copy of Nielsen's usability book. What is the most appropriate evaluation method for her situation? Why?

17

Problem

- Megabuck Inc's R&D department has just designed a new replacement for the desktop mouse that they say will revolutionize computing by cutting time-to-target in half. "Prove it" says the CEO. What is the most appropriate evaluation method? Why?

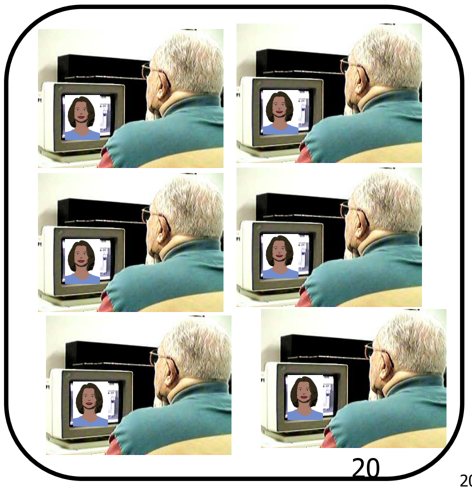
18

Problem

- Startup Industries is thinking of developing a new web portal linking office gossip blogs, and have developed an early prototype, but they're not sure if anyone will want to use it. What kind of evaluation method should they use? Why?

19

Example Living Lab The "Virtual Laboratory"



20

How can we boost engagement?

- "Virtual Laboratory" studies
  - Persistent group of study participants
  - Daily interaction with virtual exercise coach
  - Client-server architecture
  - Ability to make rapid server changes to effect new study conditions, collect new data
  - 4 years, 19,063 interactions

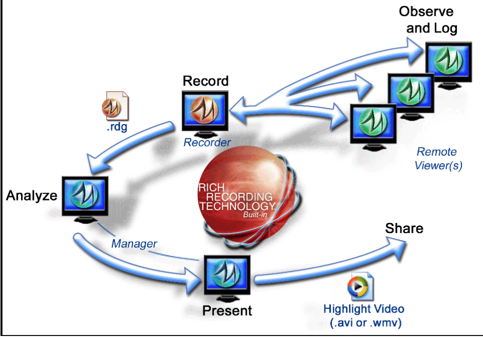
21

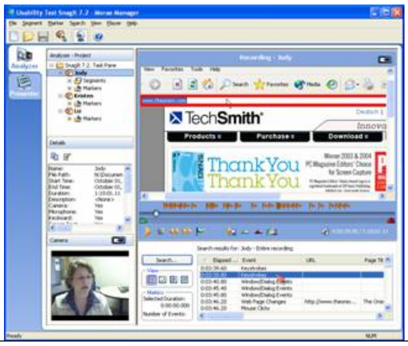
Alternatives you can use

- Mechanical Turk
- Feedback Army
- Loop 11
- etc

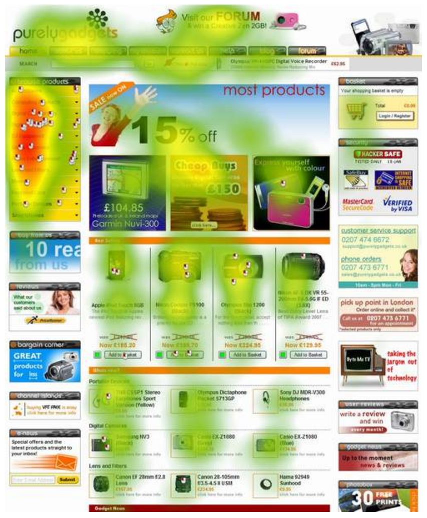
Other ways of measuring usability…

- Auto logging data
- Quantitative behavioral measures
  - Number of frowns per task



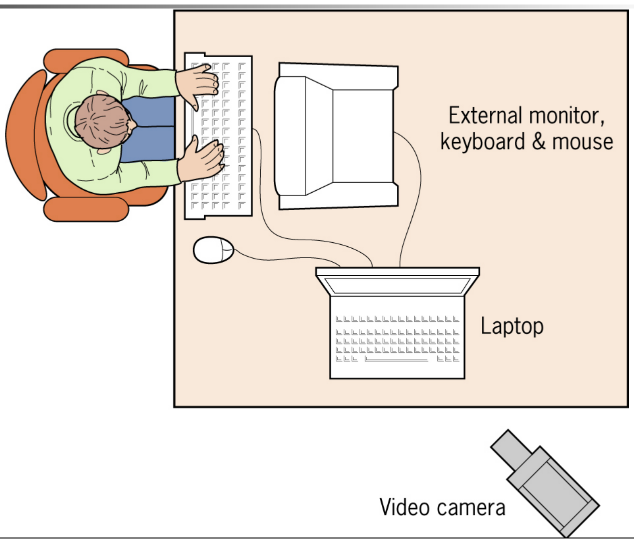


Other ways of measuring usability…



25

Low cost video mixing



26

Usability Test Research Plan

- Experiments can take a huge amount of time to plan and prepare.
- Extreme example: medical clinical trials

- **What kinds of things do you need to worry about when planning a study?**

27

Sample Research Plan

Embodied Conversational Agents to Promote Health Literacy for Older Adults

Sample Research Plan

A. SPECIFIC AIMS

B. BACKGROUND AND SIGNIFICANCE

C. PRELIMINARY STUDIES

D. RESEARCH DESIGN AND METHODS

E. HUMAN SUBJECTS

29

Sample Research Plan
Hypotheses

- H1. Immediate and distal knowledge gains and glycemic control will be significantly improved when the current standard of care is augmented with a brief "virtual consultation" with an embodied conversational agent, compared to the current standard of care alone.
- H2. Patient satisfaction will be greater when the current standard of care is augmented with an embodied conversational agent, compared to the current standard of care alone.

30

Sample Plan Research Model

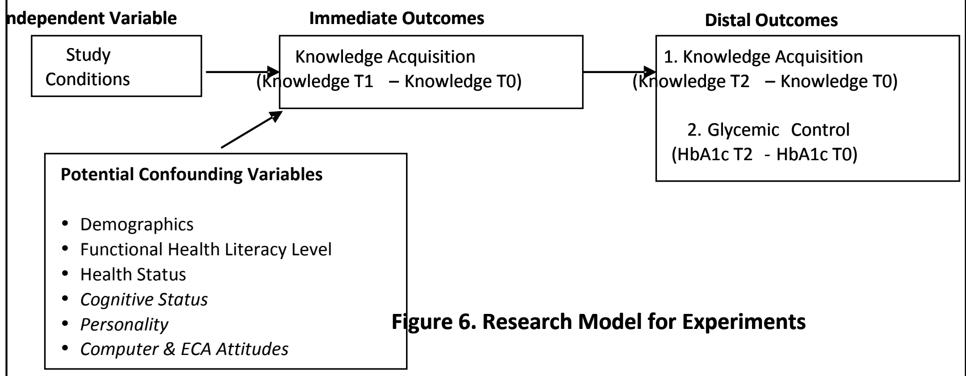

ndependent Variable
Immediate Outcomes
Distal Outcomes
Study Conditions
Knowledge Acquisition (Knowledge T1 – Knowledge T0)
1. Knowledge Acquisition (Knowledge T2 – Knowledge T0)
2. Glycemic Control (HbA1c T2 - HbA1c T0)
Potential Confounding Variables
Demographics
Functional Health Literacy Level
Health Status
Cognitive Status
Personality
Computer & ECA Attitudes

Figure 6. Research Model for Experiments

31

Measures

- D.3.1.1 Independent Variables
  - Agent vs. Standard of Care
- D.3.1.2 Primary Dependent Variables
  - Diabetes Knowledge
  - HA1c
  - Satisfaction
- D.3.1.3 Potential Confounding Variables
  - Sociodemographics
  - Health literacy
  - Cognitive status
  - Etc.

32

Sample Measure

- Diabetes Knowledge.
  - Diabetes knowledge will be assessed using the Diabetes Knowledge (DKN) Scales, three separate 15-item multiple choice questionnaires that measure general diabetes knowledge. Reliability for the items in the scales (Cronbach's alpha) was 0.92, indicating high internal consistency. Validity was assessed by determining that 219 participants who participated in a 1-1/2 day class on diabetes scored significantly higher posttest on the measures compared to pretest (11.27 vs. 7.61, p<.001).
  - We will administer the DKN immediately before the educational intervention (T0), immediately following the intervention (T1), and at three months follow up (T2).

33

Study Population

- D.3.2 Study Population
- D.3.2.1 Study Setting: The Geriatric Ambulatory Practice
- D.3.2.3 Eligibility and Exclusion Criteria
  - Eligibility criteria include:
    - Age 60 years or greater,
    - Have Type 2 diabetes mellitus, with or without complications (ICD-9 codes 250.00-250.90)
  - Exclusionary criteria include:
    - Patients with significant cognitive disability …

34

Sample Plan

- D.3.3 Sample Size and Power Considerations

35

Sample Plan

- D.3.4 Recruitment and Data Collection Procedures
  - D.3.4.1 Study Subjects
  - D.3.4.2 Recruitment and Initial Telephone Interview
  - D.3.4.3 Initial Clinic Visit (T0, T1)
  - D.3.4.4 Follow-up Clinic Visit (T2)

36

Sample Plan

- D.3.5 Analysis
  - D.3.5.1 The Analysis Plan

37

Types of Quantitative User Study Designs

- Quantitative
  - Descriptive
  - Correlational
  - Demonstrative
  - Experimental
    - Between-subjects
      - Single factor, two-level
      - Single factor, N-level (for N>2)
      - Two factor, N-level (for N>=2)
    - Within-subjects
      - Single factor, two-level

38

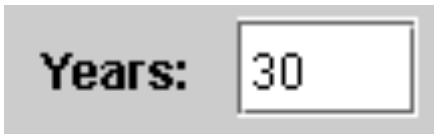
Exercise

- Project teams
- Sketch two test plans
  1. Evaluate design alternatives for interface sketches (P4)?
  2. Evaluate your final project?

Swing Layout Managers

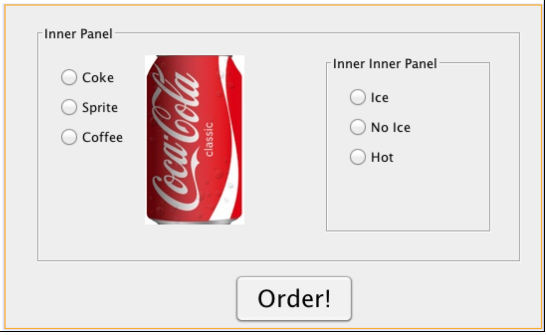
OO Toolkit Concepts
#1 Specialization via Subclassing

```
java.lang.Object
      java.awt.Component
          java.awt.Container
              javax.swing.JComponent
                  javax.swing.text.JTextComponent
                      javax.swing.JTextField
                      javax.swing.JTextArea
```



OO Toolkit Concepts
#2 Composition

- Put together interactive objects at larger scale than atomic interactors
- Container objects

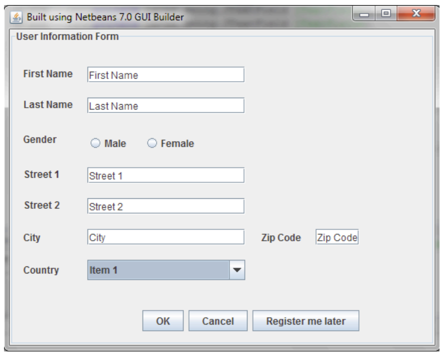


OO Toolkit Concepts #4 Event Handling

1. When anything happens in the UI
   - Mouse clicked, Window moved, Key pressed, etc
2. Windowing System creates a record
3. The event record in added to a UI event queue
4. The application (or toolkit) pulls events from the queue and acts on them in order

OO Toolkit Concepts #3 Layout

- How a container organizes its widgets within itself.

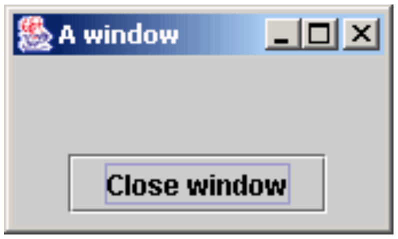

Built using Netbeans 7.0 GUI Builder

User Information Form

First Name	First Name		
Last Name	Last Name		
Gender	Male / Female		
Street 1	Street 1		
Street 2	Street 2		
City	City	Zip Code	Zip Code
Country	Item 1		

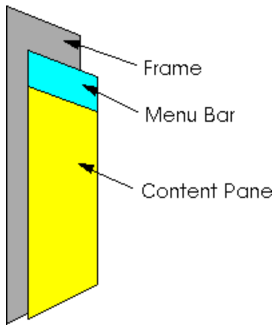
JFrame

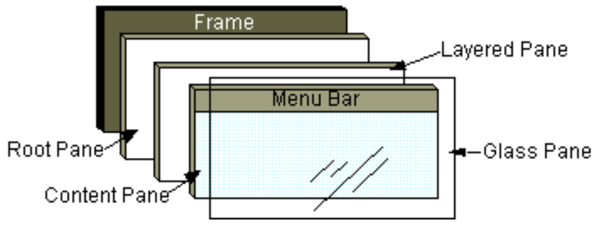
- A stand-alone window



JFrame guts

- We're just going to focus on the Content Pane





Creating a JFrame

```
class MyFrame extends JFrame {
    public MyFrame() {
      super("My Example");
       setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
      //populate the content pane: getContentPane() …
      pack();
    }
    public static void main(String args[]) {
        java.awt.EventQueue.invokeLater(new Runnable() {
            public void run() {
                new MyFrame().setVisible(true);
            }
        });
    }
}
```

JDialog

- Just like a JFrame except you can make it *modal*
- *Note:* Use JOptionPane for simple, standard alert & informational message dialogs.
- JColorChooser, JFileChooser – built in, special-purpose dialogs.

Layout Managers

- Decide how to display the Components within a Container.
- To use a layout manager:
  - Construct an instance of the manager.
  - Assign the instance to the container using:
    setLayout(LayoutManager)
  - Each Container can only have one layout manager.
- Or in NetBeans:
  R-click on component and choose "Set Layout…"

FlowLayout

- The default for JPanel
- Strategy:
  - Keeps components at their preferred size. Place components in rows, left-to-right. When a row fills up, a new row is started.
  - Rows can be centered, left or right justified.

Example FlowLayout

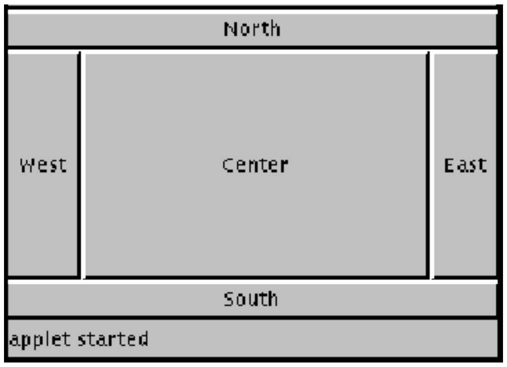
```
class FlowLayoutExample extends JFrame {
    public FlowLayoutExample() {
       super("Flow Layout Example");
       setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
       Container frame=getContentPane();
       frame.setLayout(new FlowLayout(FlowLayout.LEFT,10,10));
       frame.add(new Button("Button 1"));
       frame.add(new Button("Button 2"));
       frame.add(new Button("Button 3"));
       frame.add(new Button("Button 4"));
       frame.add(new Button("Button 5"));
       pack();
}
```

Demo

BorderLayout

- Partitions the layout space into regions
  - You specify which region you want to place Components into by name
  - At most one component can go into each region

`add(Component,<where>)`



BorderLayout Example

```
class BorderLayoutExample extends JFrame {
    public BorderLayoutExample() {
      super("Border Layout Example");
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        Container frame=getContentPane();
      frame.setLayout(new BorderLayout());
      frame.add(new Button("Button 1"),BorderLayout.NORTH);
      frame.add(new Button("Button 2"),BorderLayout.SOUTH);
      frame.add(new Button("Button 3"),BorderLayout.EAST);
      frame.add(new Button("Button 4"),BorderLayout.WEST);
      frame.add(new Button("Button 5"),BorderLayout.CENTER);
      pack();
    }
```

Demo

GridLayout

- Forms a rectangular grid of rows and columns
  - You specify the number of rows, columns, or both
  - Components are forced into the same shape for **every** cell.
  - Grid is filled left-to-right, top-down
- Constructor

  `GridLayout(int rows,int cols)`
  - Value of zero denotes undefined

GridLayout Example

```
class GridLayoutExample extends JFrame {
    public GridLayoutExample() {
      super("Grid Layout Example");
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        Container frame=getContentPane();
      frame.setLayout(new GridLayout(0,2));
      frame.add(new Button("Button 1"));
      frame.add(new Button("Button 2"));
      frame.add(new Button("Button 3"));
      frame.add(new Button("Button 4"));
      frame.add(new Button("Button 5"));
      pack();
    }
```

Demo

CardLayout

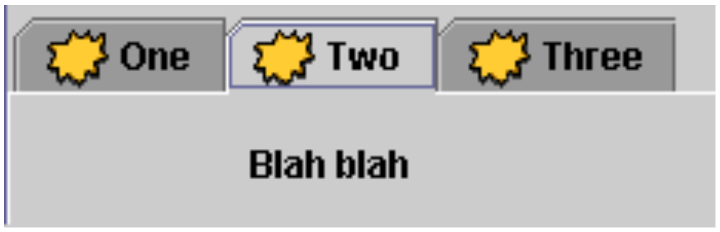
- Swaps among each of its components
- Each component can be named:

  `add(“name”,Component)`
- First component displayed initially
- To swap among components

```
CardLayout.next(Container parent)
CardLayout.first(Container parent)
CardLayout.last(Container parent)
CardLayout.show(Container parent,”name”)
```

JTabbedPane

- Acts like a JPanel with a CardLayout



Hierarchical Example

```
class HierarchyExample extends JFrame {
    public HierarchyExample() {
        super("Hierarchy Layout Example");
         setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
         Container frame=getContentPane();
        JPanel P1=new JPanel();
        JPanel P2=new JPanel();
        JPanel P3=new JPanel();
        P1.setLayout(new BorderLayout());
        P1.add(new TextArea(5,15),BorderLayout.CENTER);
        P1.add(new Button("Clear"),BorderLayout.SOUTH);
        P2.setLayout(new GridLayout(0,1));
        P2.add(new Button("Option 1"));
        P2.add(new Button("Option 2"));
        P2.add(new Button("Option 3"));
        P3.setLayout(new FlowLayout());
        P3.add(new Button("OK"));
        P3.add(new Button("Cancel"));
        frame.setLayout(new BorderLayout());
        frame.add(P1,BorderLayout.EAST);
        frame.add(P2,BorderLayout.WEST);
        frame.add(P3,BorderLayout.SOUTH);
        pack();
    }
```

Exercise – what will this look like?

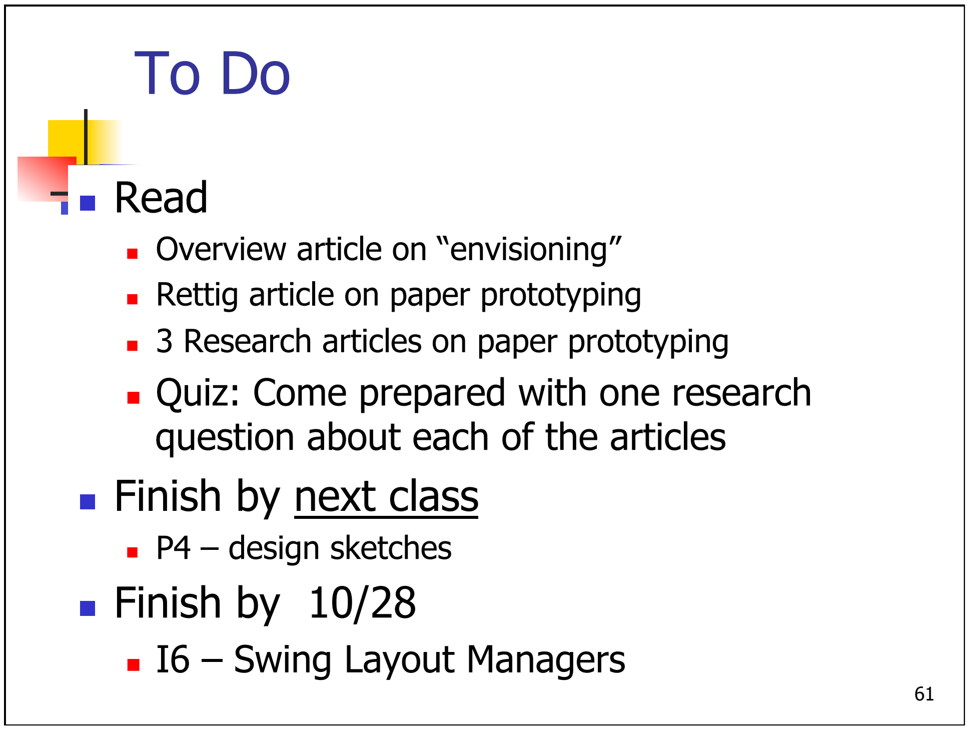
```
P1.setLayout(new BorderLayout());
P1.add(new TextArea(5,15),BorderLayout.CENTER);
P1.add(new Button("Clear"),BorderLayout.SOUTH);
P2.setLayout(new GridLayout(0,1));
P2.add(new Button("Option 1"));
P2.add(new Button("Option 2"));
P2.add(new Button("Option 3"));
P3.setLayout(new FlowLayout());
P3.add(new Button("OK"));
P3.add(new Button("Cancel"));
frame.setLayout(new BorderLayout());
frame.add(P1,BorderLayout.EAST);
frame.add(P2,BorderLayout.WEST);
frame.add(P3,BorderLayout.SOUTH);
```

Demo

Homework I6

- Your objective in this assignment is to get some experience with Frames, Dialogs and layout managers in Swing. Your mission is to create your own (ideally project-related) application with the following minimum requirements:
  - A JFrame and a (non-modal) JDialog.
  - A JTabbedPane and JScrollPane.
  - Nested JPanels including the following layout managers: GridLayout, FlowLayout, BorderLayout
  - Some interaction widgets (JButton, etc.) on every JPanel and tab.
  - Reasonable behavior when the JFrame is resized.
- NOTE: You may not use GridBagLayout, Free Design, Box, Overlay, Null or Absolute Layout anywhere in the project.

60



To Do

- Read
  - Overview article on “envisioning”
  - Rettig article on paper prototyping
  - 3 Research articles on paper prototyping
  - Quiz: Come prepared with one research question about each of the articles
- Finish by <u>next class</u>
  - P4 – design sketches
- Finish by 10/28
  - I6 – Swing Layout Managers

61