

## Coming up

- Thursday, March 5:
  - another optional guest lecture 4-5PM CS151
  - testing: “how do you know your test suite is good”
- Monday, March 9:
  - $\alpha$  due (12 noon)
  - Exam review
- Wednesday, March 11,
  - Exam @2:30, in CS 142

## Test Review

- This is how it’s going to work:
- I’ll tell you the topics and give a quick summary of each
- Then I’ll open the floor to questions
- Once we are out of questions, we’ll move on to the day’s lecture

## Exam 1 Topics

- Software development lifecycle
- **Teamwork**
- In-field debugging (guest lecture)
- **Requirements**
- Architecture
- **UML**
- User Interface
- maybe Big Software Data guest lecture

## Thursday guest lecture

### OPTIONAL

**BUT WILL BE A REALLY GOOD TALK**

### Secure and Robust Software Through Testing and Verification

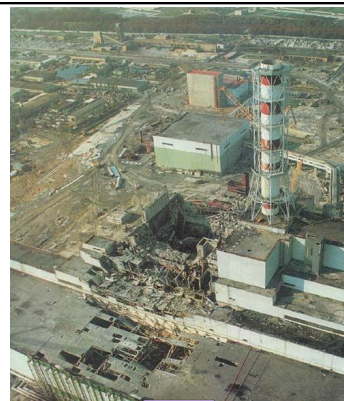
- 4 PM, Thursday 3/5 (tomorrow) CS 151
- Snacks at 3:45 as usual

<http://goo.gl/sjkw7X>

## User Interface



Three Mile Island



Chernobyl

### How do we avoid bad UI?

- Learn from past mistakes
- Build prototypes

### Big questions

- What's the point of prototyping? Should I do it?
  - If so, when should I?
- Should I make my prototype on paper or digitally?
- How do I know whether my UI is good or bad?
  - What are the ways in which a UI quality can be quantified?
  - What are some examples of software you use that have an especially good/bad UI?
  - What do you think makes them good/bad?

### Usability and software design

- **usability**: the effectiveness of users achieving tasks
  - Human-Computer Interaction (HCI).
  - Usability and good UI design are closely related.
  - A bad UI can have serious results...



### Achieving usability

- User testing and field studies
  - having users use the product and gathering data
- Evaluations and reviews by UI experts
- Prototyping
  - Paper prototyping
  - Code prototyping
- Good UI design focuses on the *user* not on the developer, not on the system environment

### Prototyping

- **prototyping**: Creating a scaled-down or incomplete version of a system to demonstrate or test its aspects.
- Reasons to do prototyping:
  - aids UI design
  - provides basis for testing
  - team-building
  - allows interaction with user to ensure satisfaction

### Some prototyping methods

1. UI builders (Visual Studio, ...)
  - draw a GUI visually by dragging/dropping UI controls on screen
2. implementation by hand
  - writing a quick version of your code
3. **paper prototyping**: a paper version of a UI



### Why do paper prototypes?

- much faster to create than code
- can change faster than code
- more visual bandwidth (can see more at once)
- more conducive to working in teams
- can be done by non-technical people
- feels less permanent or final

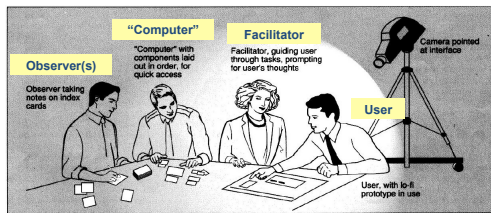
### Where does paper prototyping fit?

When in the software lifecycle is it most useful to do (paper) prototyping?

- Requirements are the **what** and design is the **how**. Which is paper prototyping?
- Prototyping
  - helps uncover requirements and upcoming design issues
  - during or after requirements but before design
  - shows us **what** is in the UI, but also shows us details of **how** the user can achieve goals in the UI

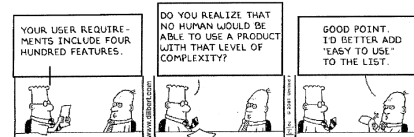
### Paper prototyping usability session

- user gets tasks to perform on a paper prototype
- observed by people and/or recorded
- a developer can "play computer"



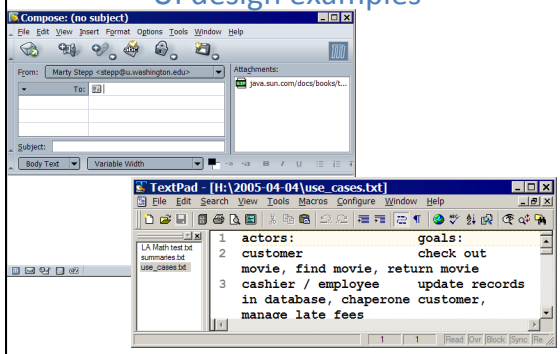
### Schneiderman's 8 Golden Rules

1. Strive for consistency.
2. Give shortcuts to the user.
3. Offer informative feedback.
4. Make each interaction with the user yield a result.
5. Offer simple error handling.
6. Permit easy undo of actions.
7. Let the user be in control.
8. Reduce short-term memory load on the user.



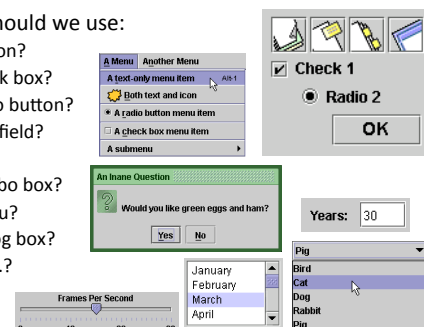
(from Designing the User Interface, by Ben Schneiderman of UMD, noted HCI and UI design expert)

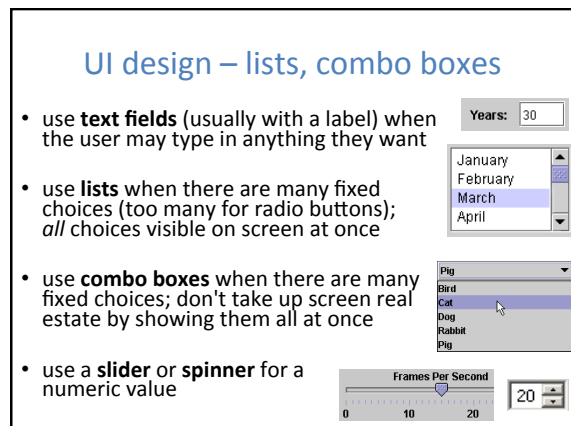
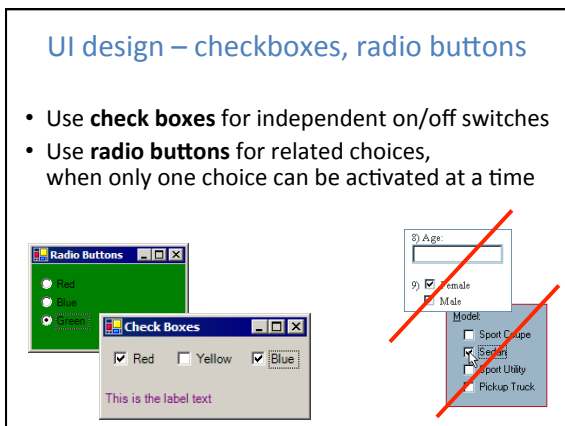
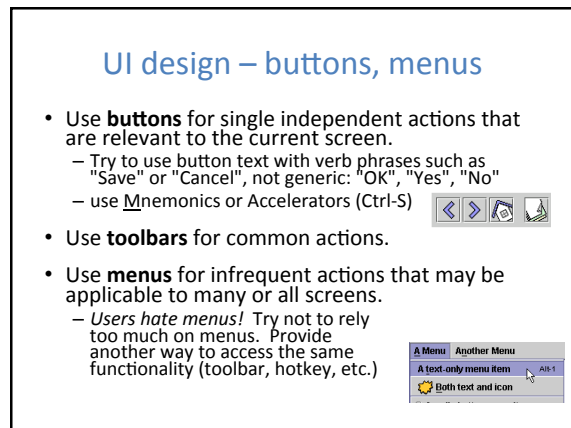
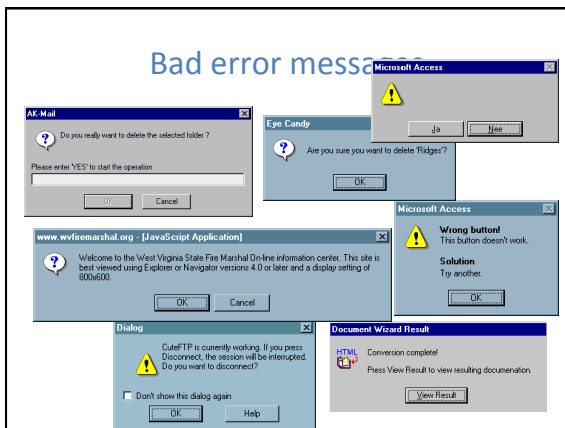
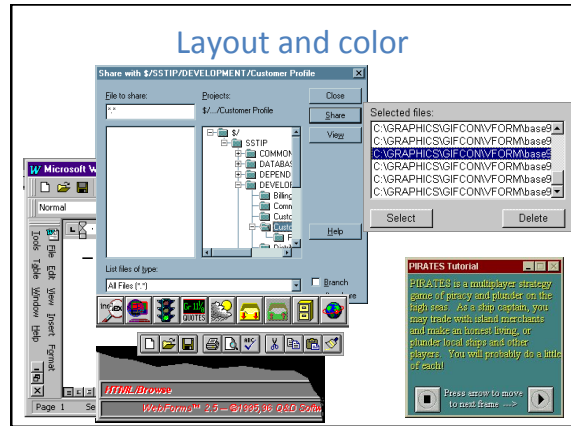
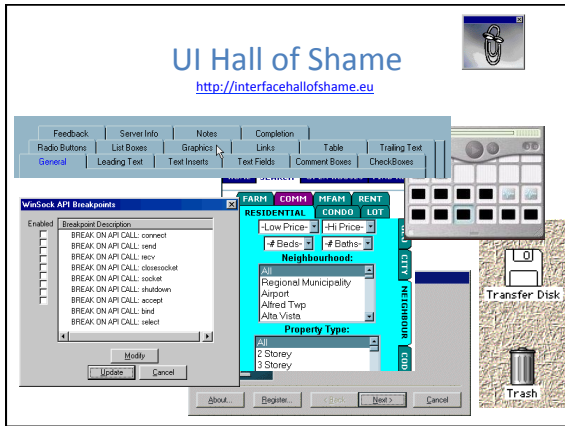
### UI design examples



### UI design, components

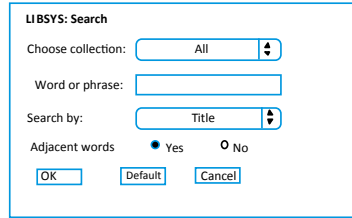
- When should we use:
  - A button?
  - A check box?
  - A radio button?
  - A text field?
  - A list?
  - A combo box?
  - A menu?
  - A dialog box?
  - Other..?





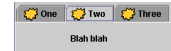
### An example UI

- Good UI dialog?  
Did the designer choose the right components?  
assume there are 20 collections and 3 ways to search

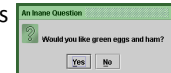


### UI design – multiple screens

- use a **tabbed pane** when there are many screens that the user may want to switch between at any moment



- use **dialog boxes** or **option panes** to present temporary screens or options



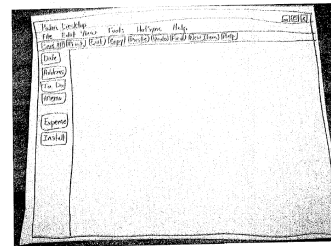
### Creating a paper prototype

- gather materials
  - paper, pencils/pens
  - tape, scissors
  - highlighters, transparencies
- identify the screens in your UI
  - consider use cases, inputs and outputs to user
- think about how to get from one screen to next
  - this will help choose between tabs, dialogs, etc.



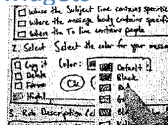
### Application backgrounds

- draw the app background (parts that matter for the prototyping) on its own, then lay the various subscreens on top of it

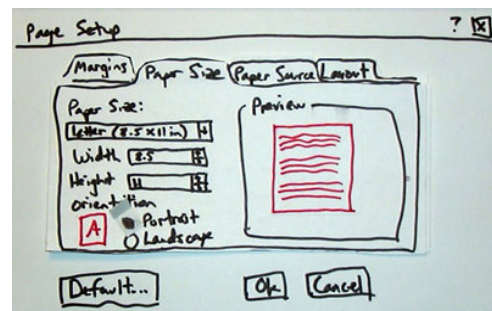


### Representing interactive widgets

- buttons / check boxes: tape
- tabs, dialog boxes: index cards
- text fields: removable tape
- combo boxes: put the choices on a separate piece of paper that pops up when they click
- selections: a highlighted piece of tape or transparency
- disabled widgets: make a gray version that can sit on top of the normal enabled version
- computer beeps: say "beep"



### Example paper prototype screen



## Let's talk about presentations

- Practice, practice, practice

## How to give a good presentation

- Practice with your team
- Practice with people outside your team
  - Your audience won't be our teammates who've been working on the project nonstop
- Aim your presentation at the right audience
- If you had never heard about the product, what kinds of things do you need to hear?

## Audience

- Who is your audience?

Your customer is your audience.

- Before you begin:
  - List the things you want to convey to your customer
  - Figure out the most effective way to convey them
  - Structure the presentation around that

**PRACTICE!**

## Prototyping exercise

- In your project groups, draw a rough prototype for a music player (e.g., WinAmp or iTunes).
  - Assume that the program lets you store, organize, and play songs and music videos.
  - Draw the main player UI and whatever widgets are required to do a **search for a song or video**.
  - After the prototypes are done, we'll try walking through each UI together.
- Things to think about:
  - How many clicks are needed? What controls to use?
  - Could your parents figure it out without guidance?

34

## Thursday guest lecture

OPTIONAL

**BUT WILL BE A REALLY GOOD TALK**

**Secure and Robust Software Through Testing and Verification**

- 4 PM, Thursday 3/5 (tomorrow) CS 151
- Snacks at 3:45 as usual

<http://goo.gl/sjkw7X>