coming up

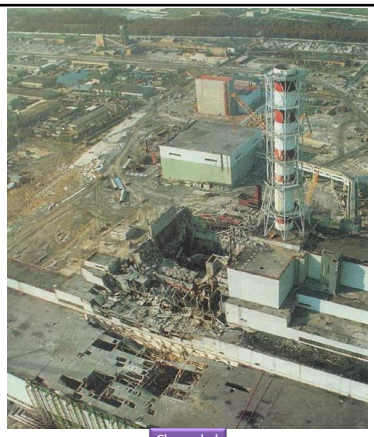
- α assignment due October 25, at noon
- give anonymous class feedback https://forms.gle/A5TSF3wHu3pos5dm7
- reminder: there will be peer evaluation (first one on Nov 3)

1

User Interface

Three Mile Island

2



Chernobyl

3

How do we avoid bad UI?

- Learn from past mistakes
- Build prototypes

4

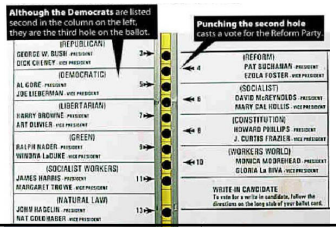
Big questions

- 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?

5

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…



6

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

7

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

8

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

9

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

10

Where does paper prototyping fit?

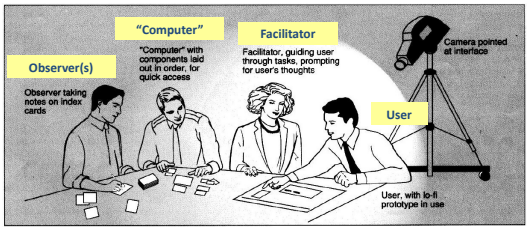
When in the software lifecycle is it most useful to do (paper) prototyping?

- 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

11

Paper prototyping usability session

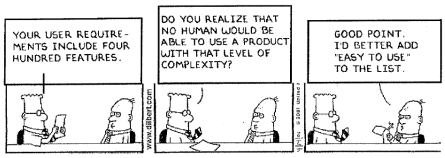
- user gets tasks to perform on a paper prototype
- observed by people and/or recorded
- a developer can "play computer"



12

Schneiderman's 8 Golden Rules

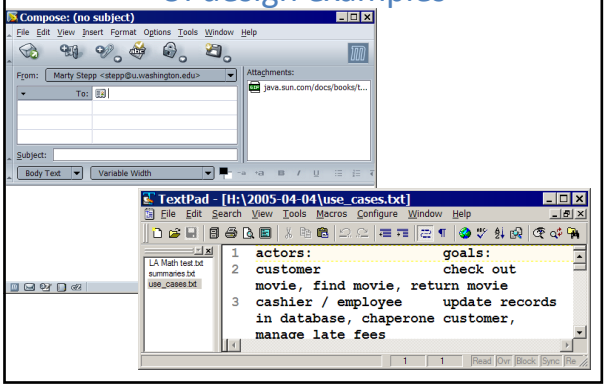
1. Strive for consistency.
2. Give shortcuts to the user.
3. Offer informative feedback.
4. Make each interaction with the user yield a result.
5. Offer simple error handling.
6. Permit easy undo of actions.
7. Let the user be in control.
8. Reduce short-term memory load on the user.



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

13

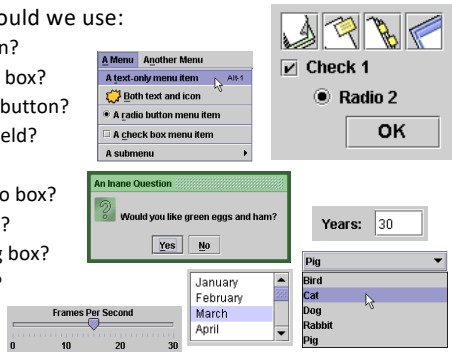
UI design examples



14

UI design, components

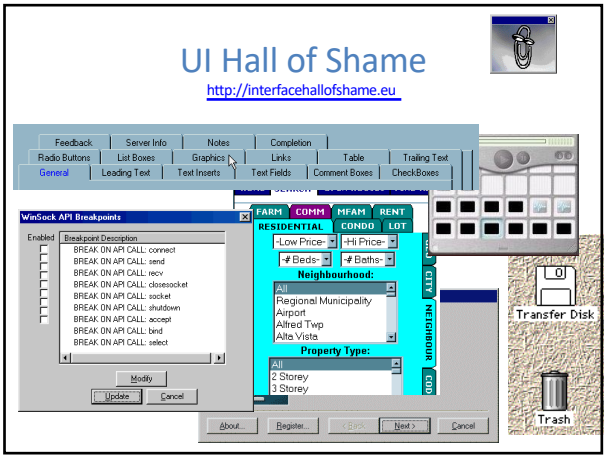
- When should we use:
  - A button?
  - A check box?
  - A radio button?
  - A text field?
  - A list?
  - A combo box?
  - A menu?
  - A dialog box?
  - Other..?



15

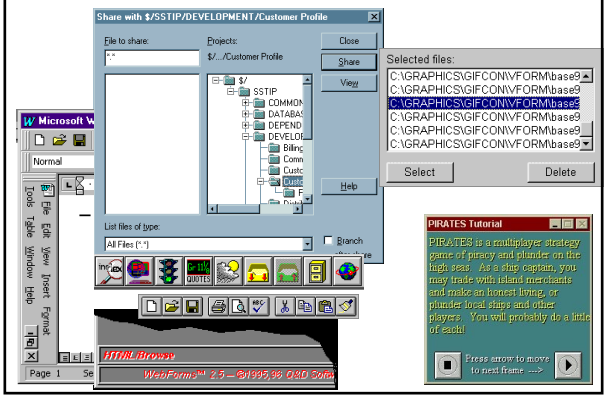
UI Hall of Shame

http://interfacehallofshame.eu



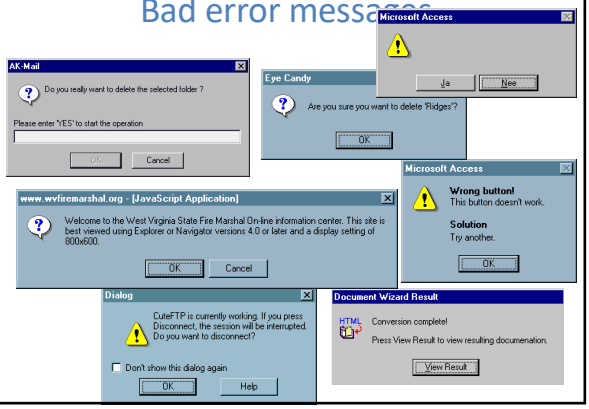
16

Layout and color



17

Bad error messages

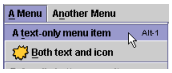


18

UI design – buttons, menus

- Use **buttons** for single independent actions that are relevant to the current screen.
  - Try to use button text with verb phrases such as "Save" or "Cancel", not generic: "OK", "Yes", "No"
  - use Mnemonics or Accelerators (Ctrl-S)

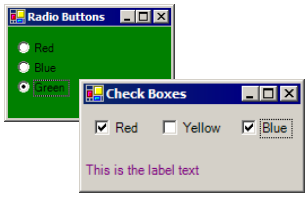
- Use **toolbars** for common actions.
- Use **menus** for infrequent actions that may be applicable to many or all screens.
  - *Users hate menus!* Try not to rely too much on menus. Provide another way to access the same functionality (toolbar, hotkey, etc.)

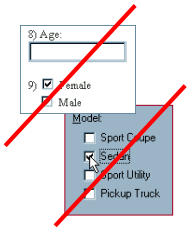


19

UI design – checkboxes, radio buttons

- Use **check boxes** for independent on/off switches
- Use **radio buttons** for related choices, when only one choice can be activated at a time



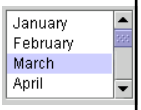


20

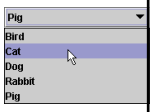
UI design – lists, combo boxes

- use **text fields** (usually with a label) when the user may type in anything they want

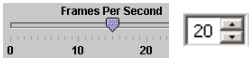
- use **lists** when there are many fixed choices (too many for radio buttons); *all* choices visible on screen at once



- use **combo boxes** when there are many fixed choices; don't take up screen real estate by showing them all at once



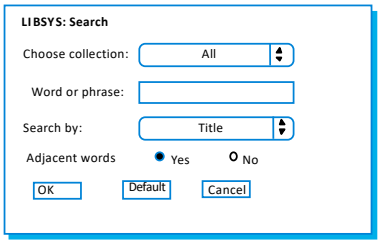
- use a **slider** or **spinner** for a numeric value



21

An example UI

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

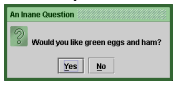


22

UI design – multiple screens

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

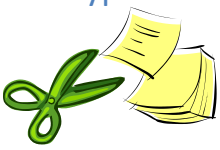
- use **dialog boxes** or **option panes** to present temporary screens or options



23

Creating a paper prototype

- gather materials
  - paper, pencils/pens
  - tape, scissors
  - highlighters, transparencies

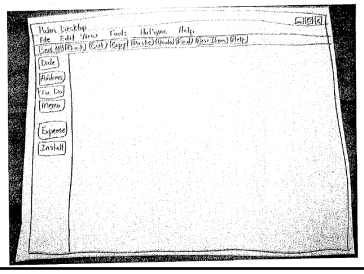


- identify the screens in your UI
  - consider use cases, inputs and outputs to user
- think about how to get from one screen to next
  - this will help choose between tabs, dialogs, etc.

24

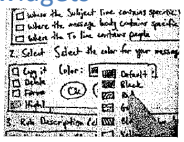
Application backgrounds

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



25

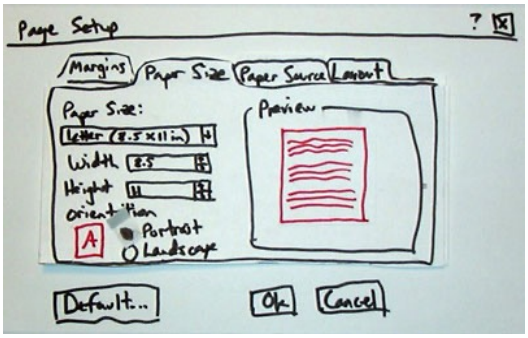
Representing interactive widgets



- 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"

26

Example paper prototype screen



27