There is both a “U” and an “I” in UI

Big questions

- What’s the point of prototyping? Should I do it?
  - If so, when in the overall process or “lifecycle” 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’s “quality” can be quantified?
  - What are some examples of software you use that have especially good/bad UIs? What do you think makes them good/bad?

Usability and software design

- usability: the effectiveness with which users can achieve tasks in one software environment
  - Studying and improving usability is part of Human-Computer Interaction (HCI)
  - Usability and good UI design are closely related
  - A bad UI can have unfortunate results...

Achieving usability

- Some methods to achieve good usability:
  - User testing / field studies – having users use the product and gathering data
  - Evaluations and reviews by UI experts
  - Card sorting
    - Show users various UI menus and ask them to group the ones that are similar, to see what UI tasks are seen as being related by users.
  - Prototyping
    - Paper prototyping
    - Code prototyping
  - Good UI design focuses on the user – not on the developer or on the system environment
Prototyping

• prototyping: Creating a scaled-down or incomplete version of a system to demonstrate or test aspects of it
• Balzer et al.: Prototyping is “building to learn”
• Reasons to do prototyping:
  – aids UI design
  – provides basis for testing
  – team-building
  – allows interaction with user to ensure satisfaction

Some prototyping methods

• UI builders (Visual Studio, ...)
  – draw a GUI visually by dragging/dropping UI controls on screen
• implementation by hand
  – writing a “quick” version of your code
• paper prototyping: a paper version of a UI
  – Why not just code up a working prototype?

Benefits of paper prototype

• 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?

• At what point in the software lifecycle should we do (paper) prototyping? When would it be most useful to do it? Why?
• If we talk about requirements being about “what” and design being about “how,” then which is paper prototyping?
  – PP helps us uncover requirements and also upcoming design issues
  – Do PP during or after requirements; before design
  – “what” vs. “how”: PP shows us “what” is in the UI, but it also shows us details of “how” the user can achieve their goals in the UI
• The UI is surely a connection to the “world”, in Jackson’s terms
Paper prototyping usability session

- user is given tasks to perform using paper prototype
- session can be observed by people or camera
- one developer can "play computer"

Schneiderman's 8 Golden Rules

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

(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..?

Apple Mac user interfaces
UI Hall of Shame


Layout and color

Bad error messages

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)
Checkboxes, radio buttons

- Use check boxes for on/off switches, when any one switch can be toggled irrespective of the others (often correspond to boolean values).
- Use radio buttons for related choices, when only one choice can be activated at a time (often corresponds to enum/constant values).

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 to be practical) and you want all choices visible on screen at once
- use combo boxes when there are many fixed choices, but you don’t want to take up screen real estate by showing them all at once
- use a slider or spinner for a numeric value

An example UI

- What can we say about this UI dialog? Did the designer choose the right components?
  - Let’s assume there are 20 collections and 3 ways to search (by title, author, relevancy)

A few comments

- Some of the combo boxes might be better as lists.
- The Adjacent Words should be a checkbox.
- The button labels (especially “Default”, which could be “Reset to Default”) are poor.
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 (the parts that matter for the prototyping) on its own, then lay the various subscreens on top of it

Representing a changing UI

- layers of UI can be placed on top of background as user clicks various options
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: ?

Example full paper prototype

Prototyping exercise you can try

- Draw a rough prototype for a music player (e.g. 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.
  - Walk through the UI with another person or group
- Things to think about:
  - How many clicks are needed? What controls to use?
  - Could your parents figure it out without guidance?