User Interfaces Design

Dr. Mohammad Ahmad

Lecture 7

Key Definitions

- The user interface defines how the system will interact with external entities

- The system interfaces define how systems exchange information with other systems

Key Definitions

- The **navigation mechanism** provides the way for users to tell the system what to do

- The **input mechanism** defines the way the system captures information

- The **output mechanism** defines the way the system provides information to users or other systems

Key Definitions

- **Graphical user interface (GUI)** is the most common type of interfaces.

The principles of user interface design-1

- *The structure principle*: Design should organize the user interface purposefully, in meaningful and useful ways based on clear, consistent models that are apparent and recognizable to users, putting related things together and separating unrelated things, differentiating dissimilar things and making similar things resemble one another. The structure principle is concerned with overall user interface architecture.

- *The simplicity principle*: The design should make simple, common tasks easy, communicating clearly and simply in the user's own language, and providing good shortcuts that are meaningfully related to longer procedures.

- *The visibility principle*: The design should make all needed options and materials for a given task visible without distracting the user with extraneous or redundant information. Good designs don't overwhelm users with alternatives or confuse with unneeded information.

The principles of user interface design-2

- ***The feedback principle***: The design should keep users informed of actions or interpretations, changes of state or condition, and errors or exceptions that are relevant and of interest to the user through clear, concise, and unambiguous language familiar to users.

- ***The tolerance principle***: The design should be flexible and tolerant, reducing the cost of mistakes and misuse by allowing undoing and redoing, while also preventing errors wherever possible by tolerating varied inputs and sequences and by interpreting all reasonable actions.

- ***The reuse principle***: The design should reuse internal and external components and behaviors, maintaining consistency with purpose rather than merely arbitrary consistency, thus reducing the need for users to rethink and remember.

More Principles for User Interface Design

- Layout

- Content awareness

- Aesthetics

- User experience

- Consistency

- Minimal user effort

Layout Concepts

- The screen is often divided into three boxes
    - Navigation area (top)
    - Status area (bottom)
    - Work area (middle)
- Information can be presented in multiple areas
- Like areas should be grouped together

More Layout Concepts

- Areas and information should minimize user movement from one to another

- Ideally, areas will remain consistent in
  - Size
  - Shape
  - Placement for entering data
  - Reports presenting retrieved data

Layout with Multiple Navigation Areas

Content Awareness

- All interfaces should have titles
- Menus should show
  - where you are
  - where you came from to get there
- It should be clear what information is within each area
- Fields and field labels should be selected carefully
- Use dates and version numbers to aid system users

Aesthetics

- Interfaces need to be functional and inviting to use

- Avoid squeezing in too much, particularly for novice users

- Design text carefully
  - Be aware of font and size
  - Avoid using all capital letters

More Aesthetics

- Colors and patterns should be used carefully
  - Test quality of colors by trying the interface on a black/white monitor
  - Use colors to separate or categorize items

User Experience

- How easy is the program to learn?

- How easy is the program to use for the expert?

- Consider adding shortcuts for the expert

- Where there is low employee turnover, some training can lessen the impact of less precise interfaces

Consistency

- Enables users to predict what will happen
- Reduces learning curve
- Considers items within an application and across applications
- Pertains to many different levels
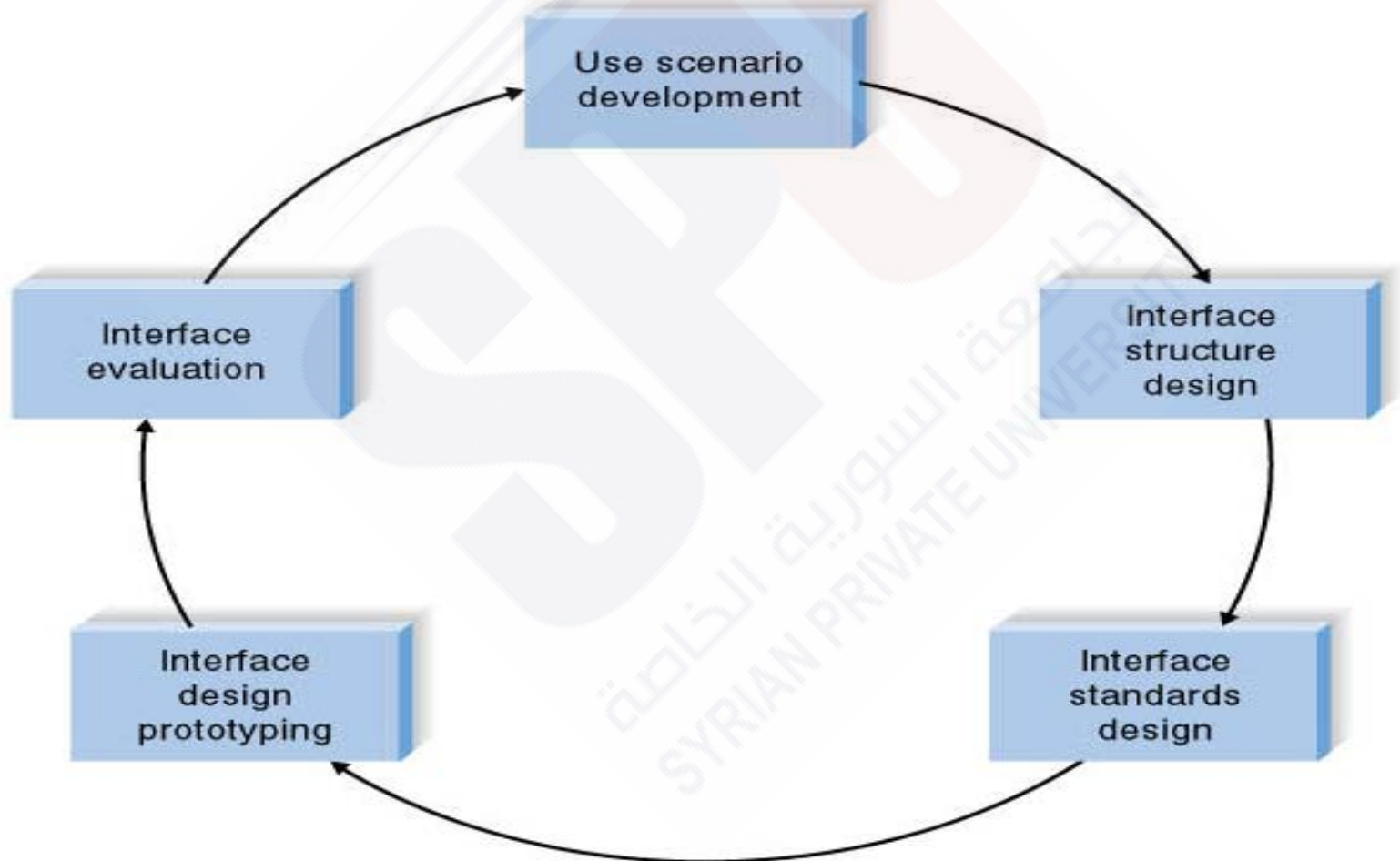 - Navigation controls
  - Terminology
  - Report and form design

Minimize Effort

- Three clicks rule
  - **Users should be able to go from the start or main menu of a system to the information or action they want in no more than three mouse clicks or three keystrokes**

USER INTERFACE DESIGN PROCESS

User Interface Design Process

Use Scenario Development

- An outline of steps to perform work

- Presented in a simple narrative tied through the related use case and DFD

- Document the most common paths through the use case so interface designs will be easy to use for those situations

Interface Structure Design

- A diagram showing how all screens, forms, and reports are related

- Shows how user moves from one to another

- Similar to DFD in using boxes and lines

- Boxes denote screens

- Lines show movement from one to another

- Different from DFD in having no standard rules or format

Interface Structure Diagram Example
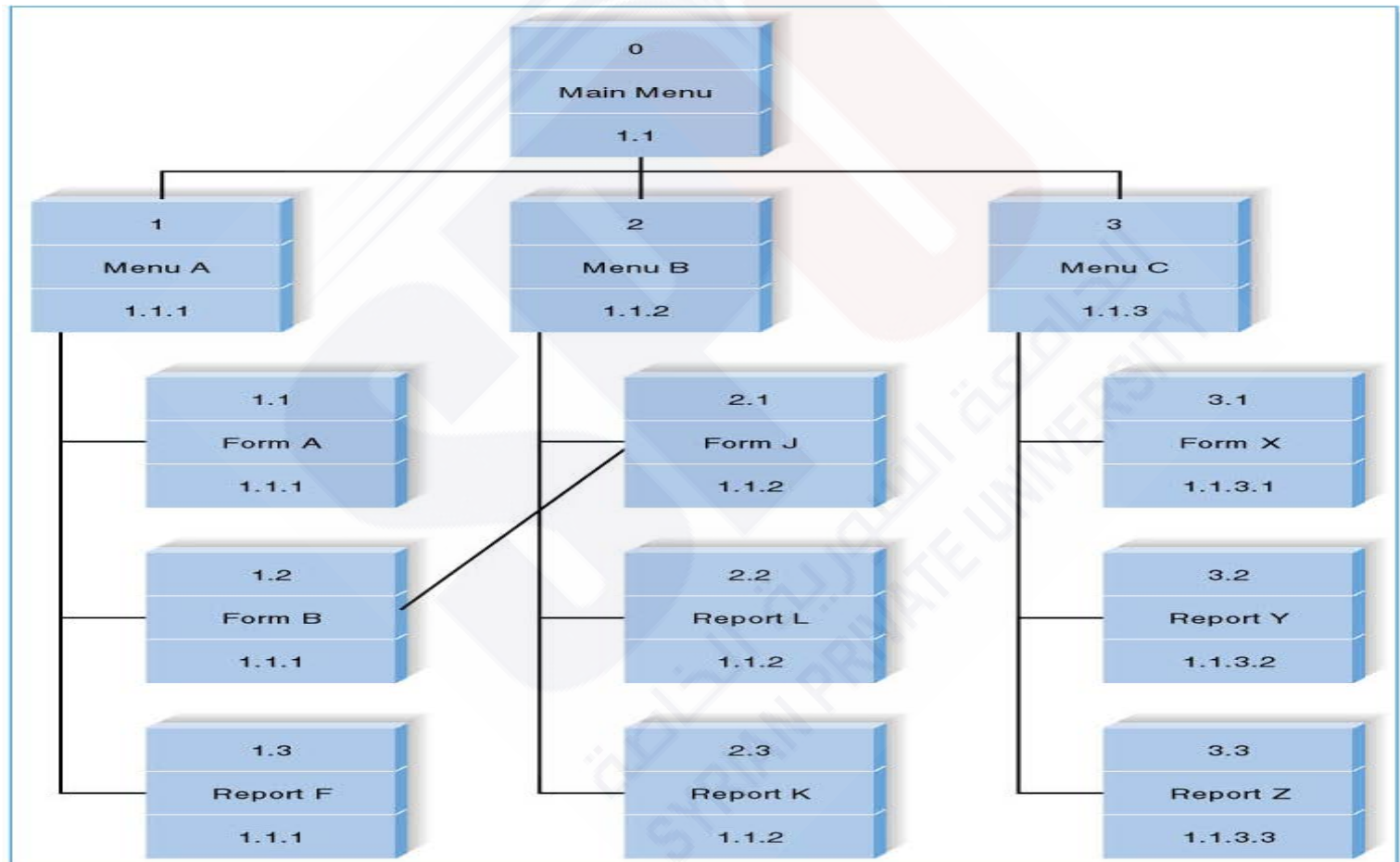
Interface Standards Design

- The basic elements that are common across individual screens, forms, and reports within the application

- Interface metaphor

  – Desktop, checkbook, shopping cart

- Interface objects

- Interface actions

- Interface icons

- Interface templates

Interface Design Prototyping

- A mock-up or simulation of screen, form, or report

- Common methods include
  - Paper
  - Storyboarding
  - HTML prototype

Storyboard Example
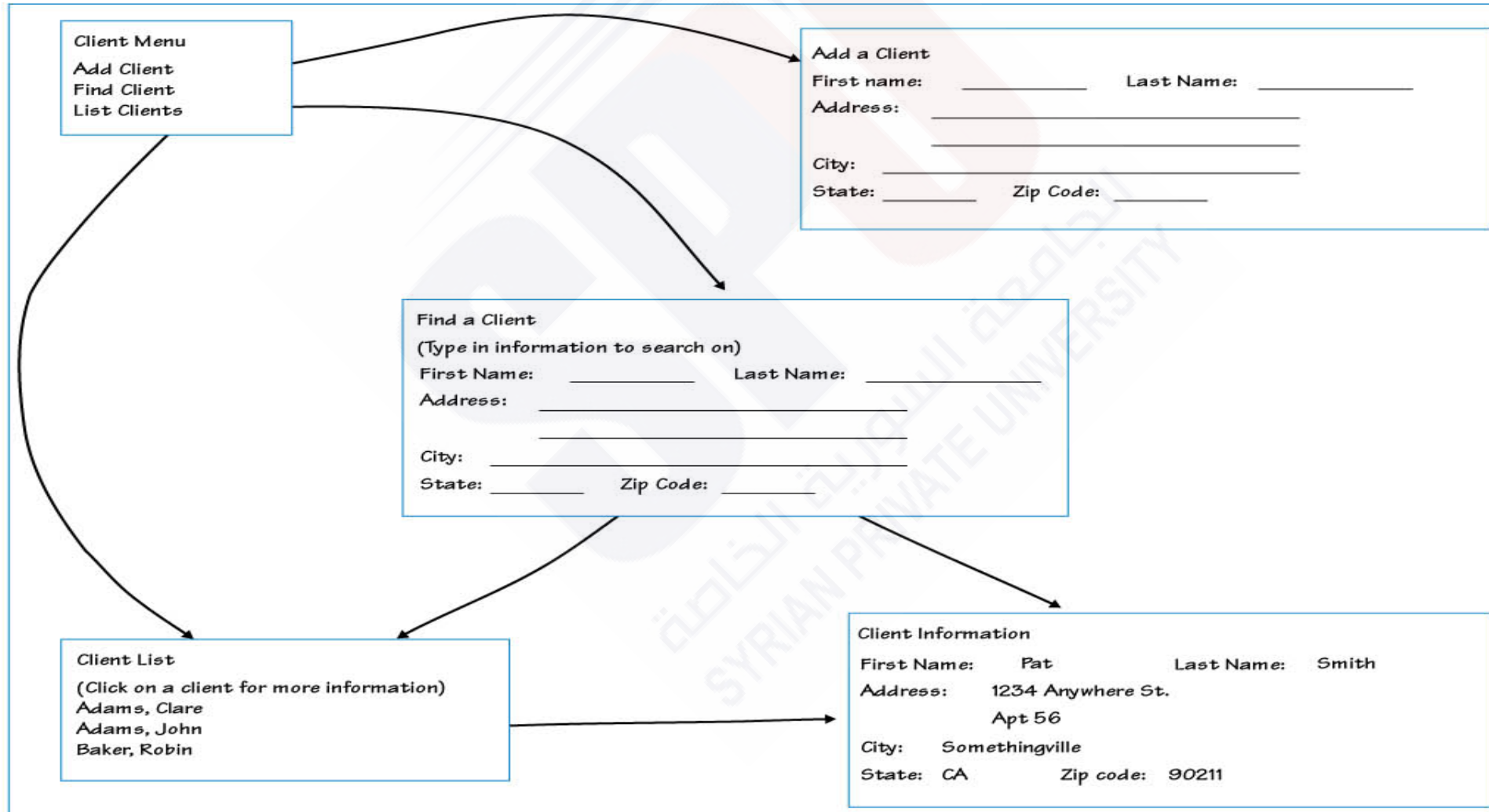
Interface Evaluation

- Heuristic evaluation
  - Compare design to checklist
- Walkthrough evaluation
  - Team simulates movement through components
- Interactive evaluation
  - Users try out the system
- Formal usability testing
  - Expensive
  - Detailed use of special lab testing

NAVIGATION DESIGN

Basic Principles of Navigation Design

- Assume users
  - Have not read the manual
  - Have not attended training
  - Do not have external help readily at hand
- All controls should be clear and understandable and placed in an intuitive location on the screen.

Basic Principles of Navigation Design

- Prevent mistakes
  - Limit choices
  - Never display commands that can't be used (or "gray them out")
  - Confirm actions that are difficult or impossible to undo
- Simplify recovery from mistakes
- Use consistent grammar order

Types of Navigation Control

- Languages
  - Command language
  - Natural language

- Menus
  - Generally aim at broad shallow menu
  - Consider using "hot keys"

- Direct Manipulation
  - Used with icons to start programs
  - Used to shape and size objects
  - May not be intuitive for all commands

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).
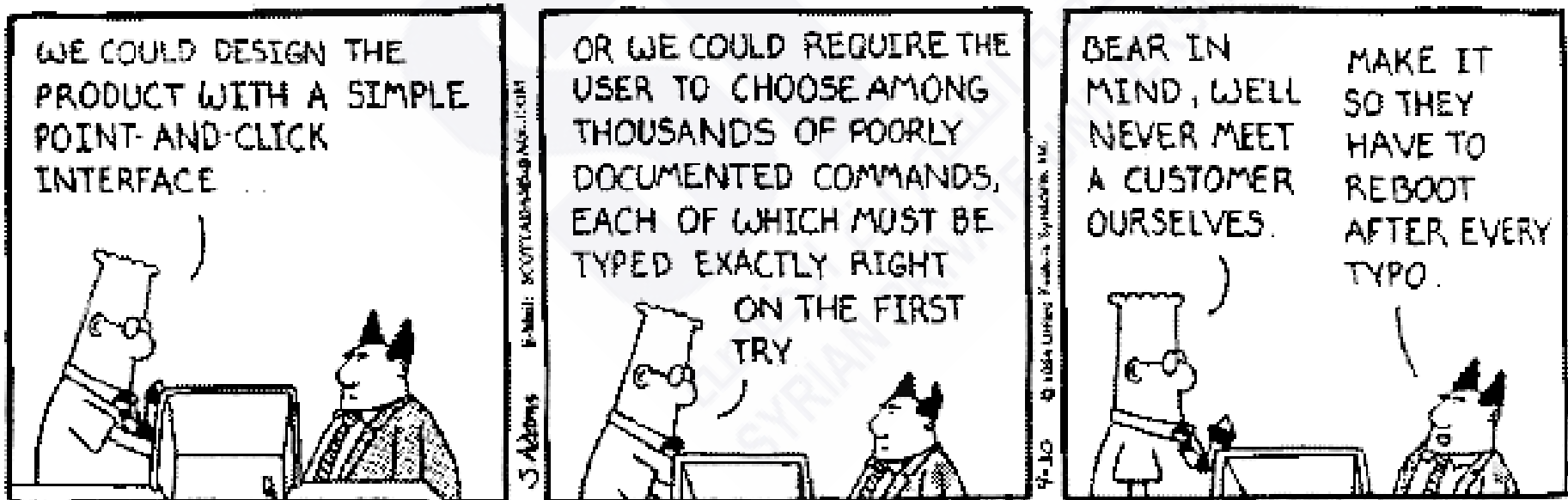
Good UI design and usability

- 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.

- 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
  - in the past, was done in simple languages like Visual Basic
  - modern IDEs have UI builders for more robust languages such as Java, C#

- ## implementation by hand
  - writing a "quick" version of your code
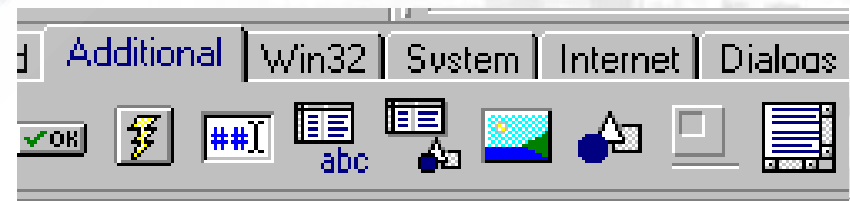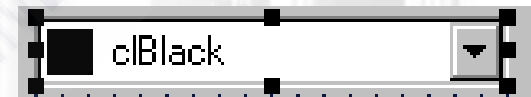
Paper prototyping

- **paper prototyping**: a means of usability testing where representative users perform tasks by interacting with a paper version of a user interface

- Why paper prototype?
  Why not just code up a working 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

Comparison of techniques

	paper prototype	UI builder	actual implementation
ease of use	+ +	0	--
fast turn-around	+ +	+	--
flexibility, control	+ +	+	-
executable	--	+	+ +
team design	+	-	0

Where does P.P. fit in?

- At what point in the software lifecycle should we do (paper) prototyping? When would it be most useful to do it? Why?

- We talk about requirements being about "what" and design being about "how." 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

A P.P. usability session

- user is given tasks to perform using paper prototype
- session can be observed by people or camera
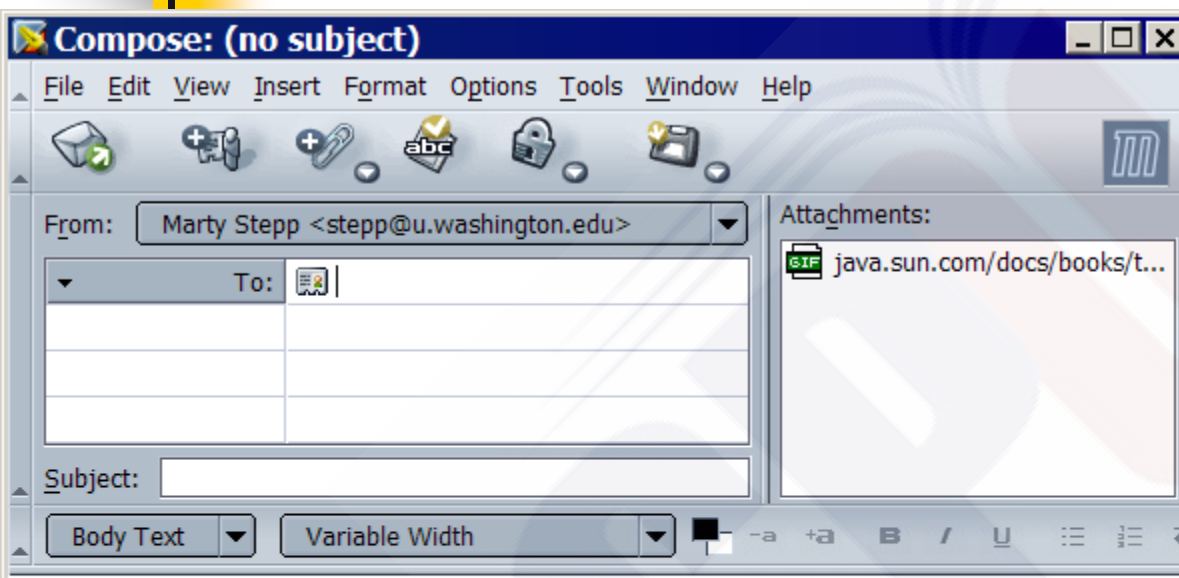- one developer can "play computer"
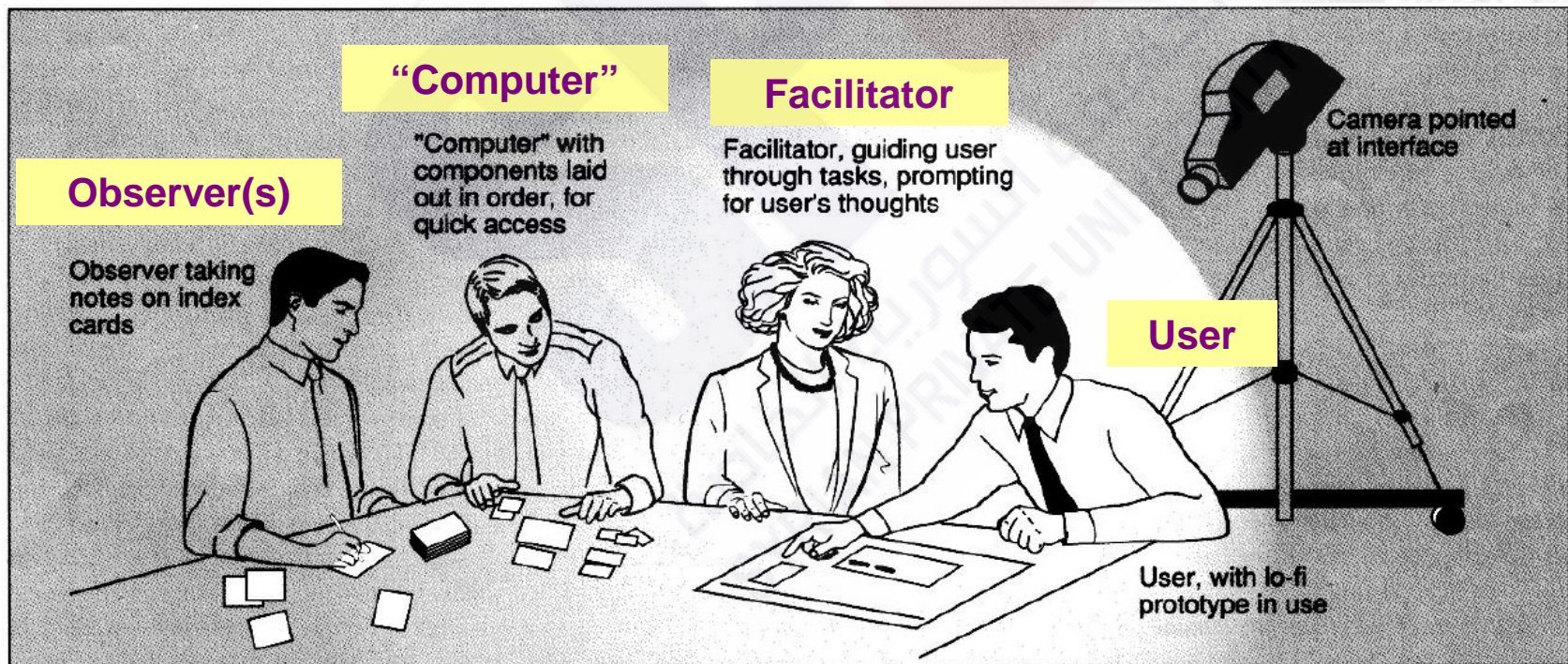
Schneiderman's 8 Golden Rules

- 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.
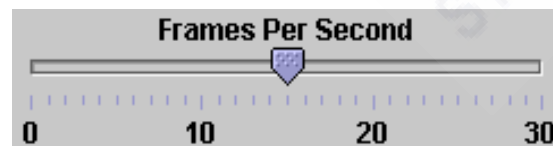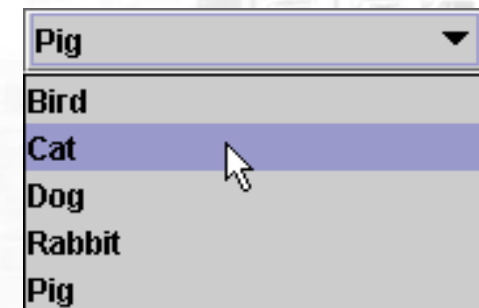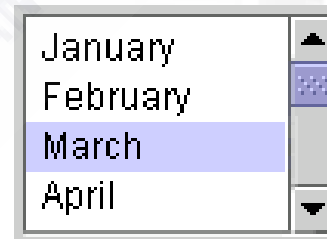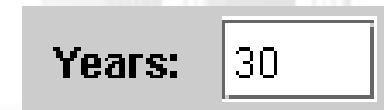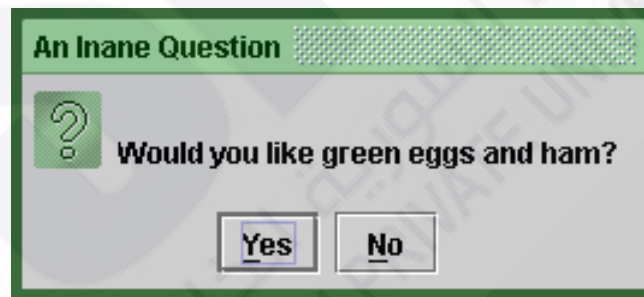- Reduce short-term memory load on the user.
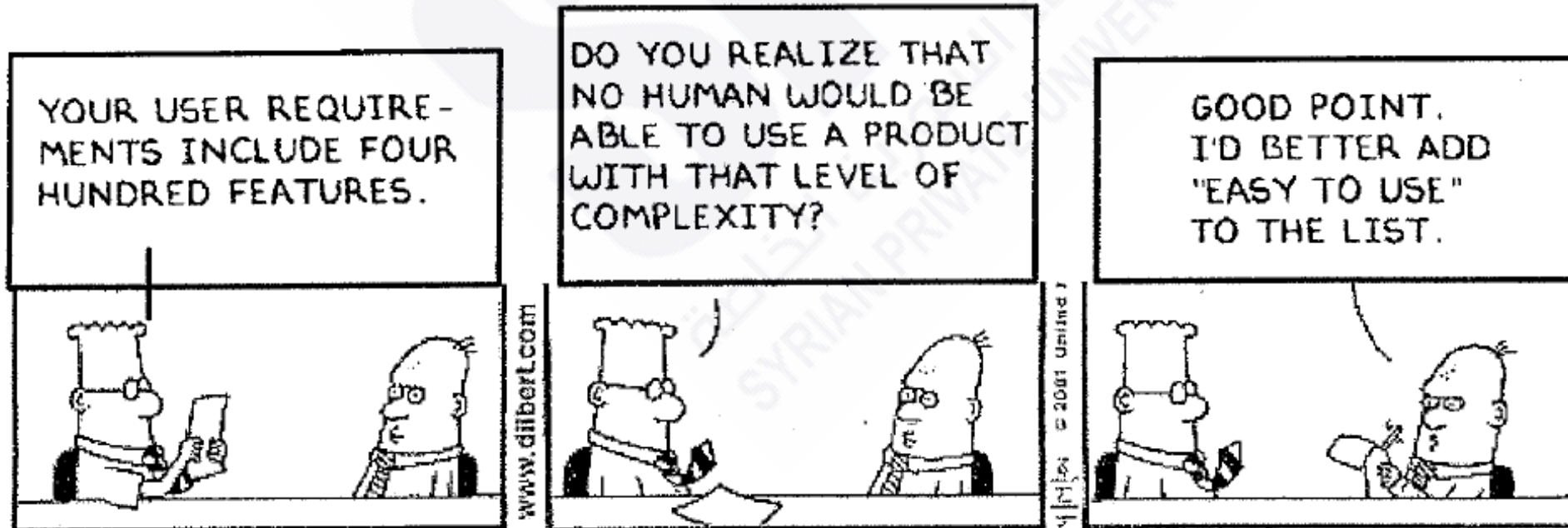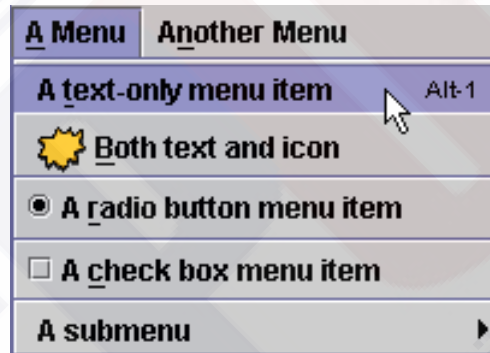
UI design examples
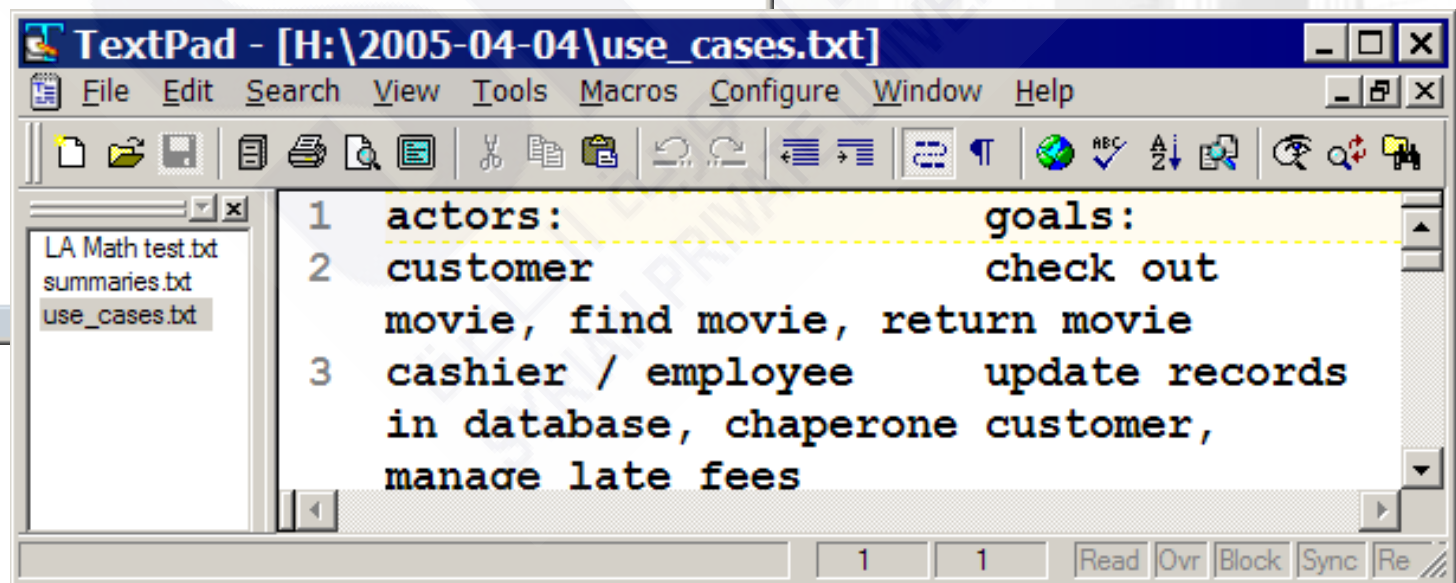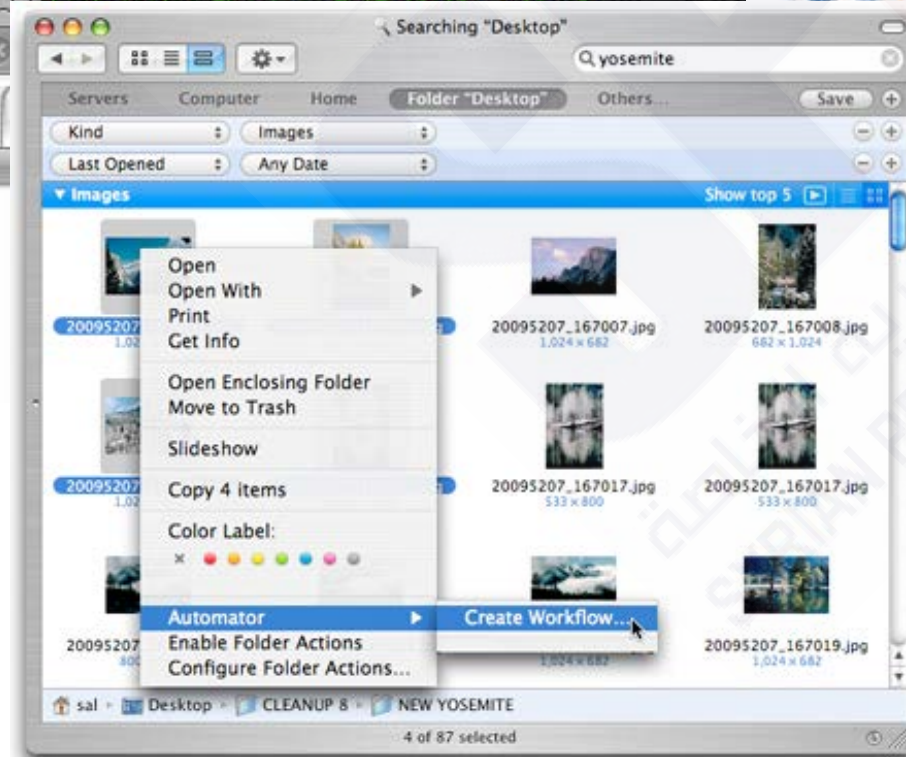
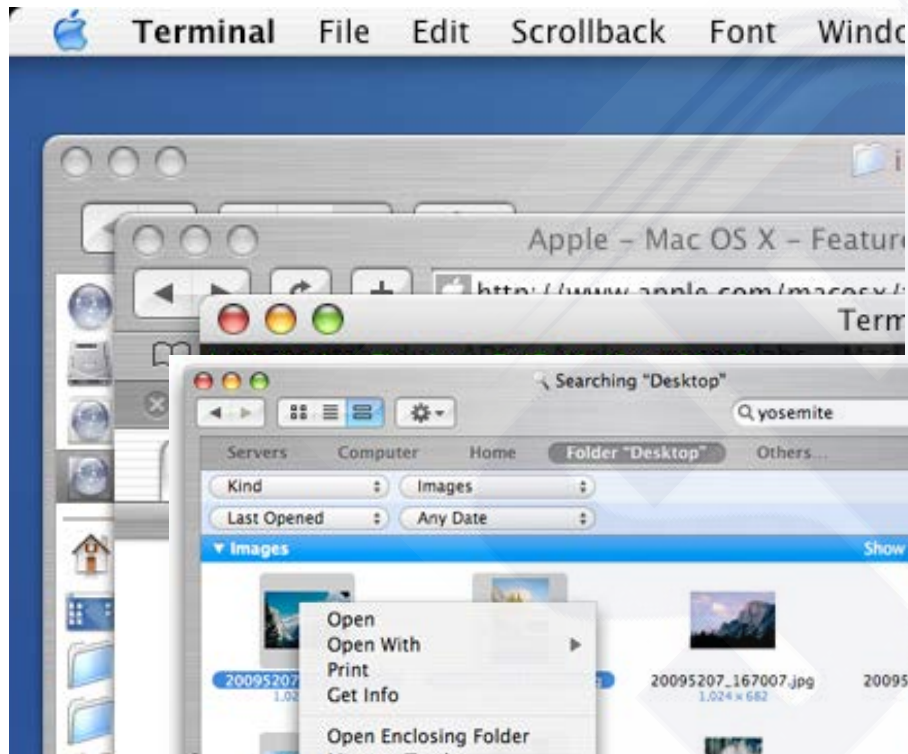# UI design, components

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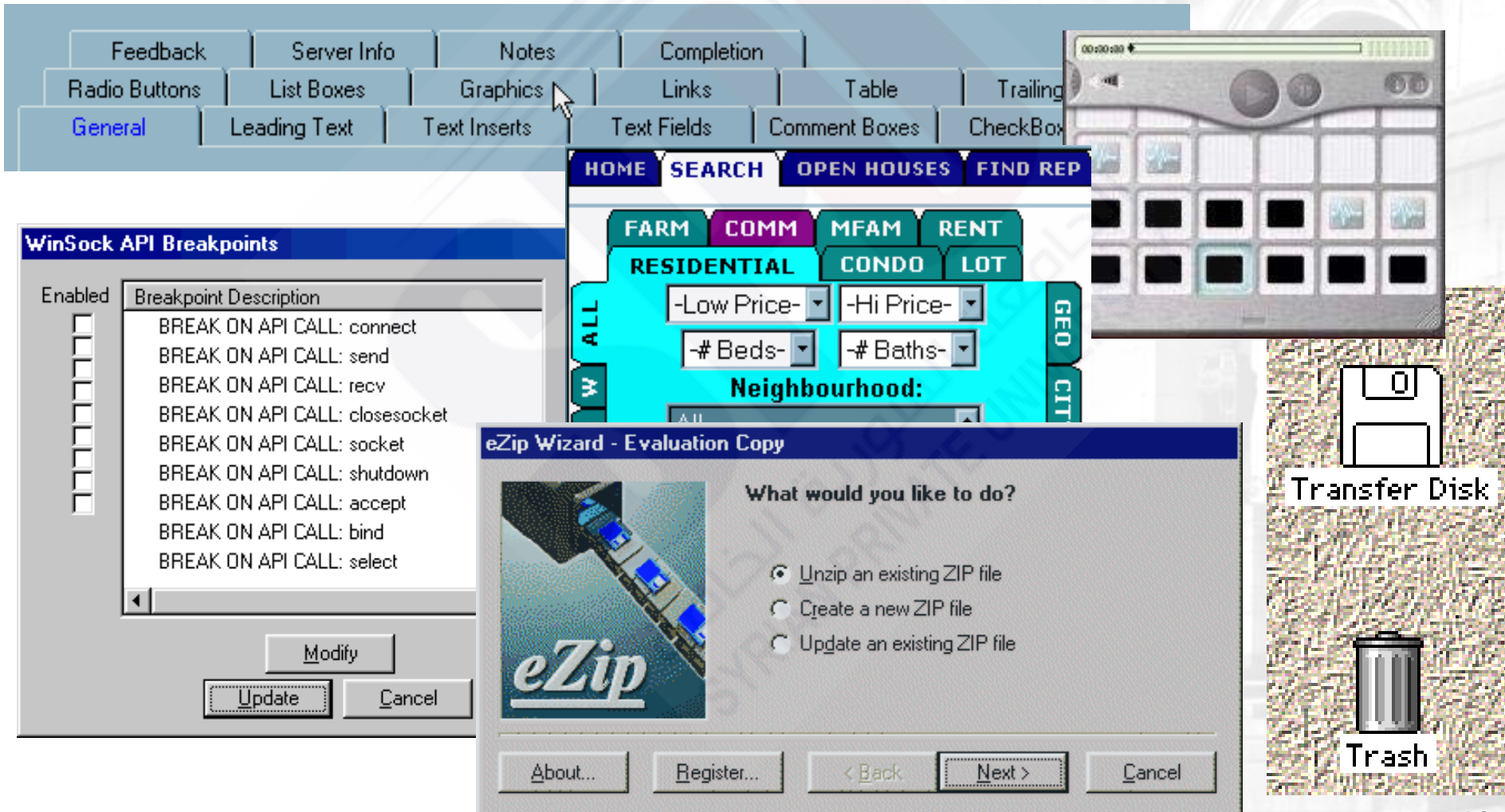 - A text field?
  - A list?
  - A combo box?
  - A menu?
  - A dialog box?
  - Other..?

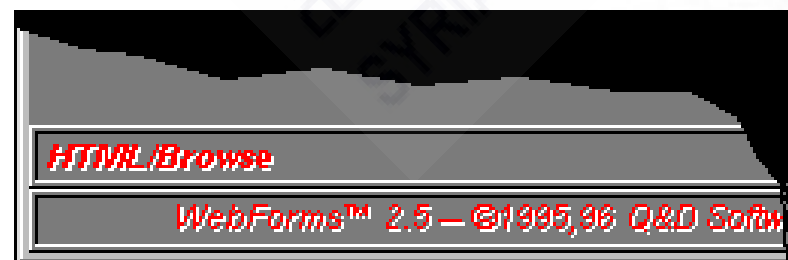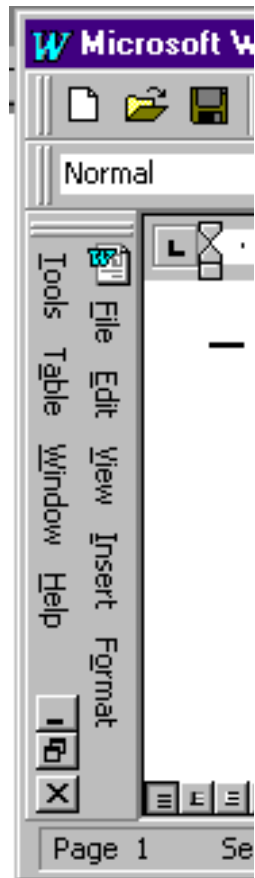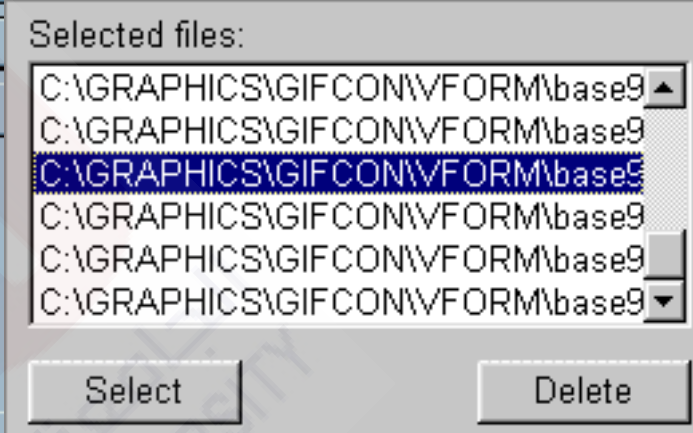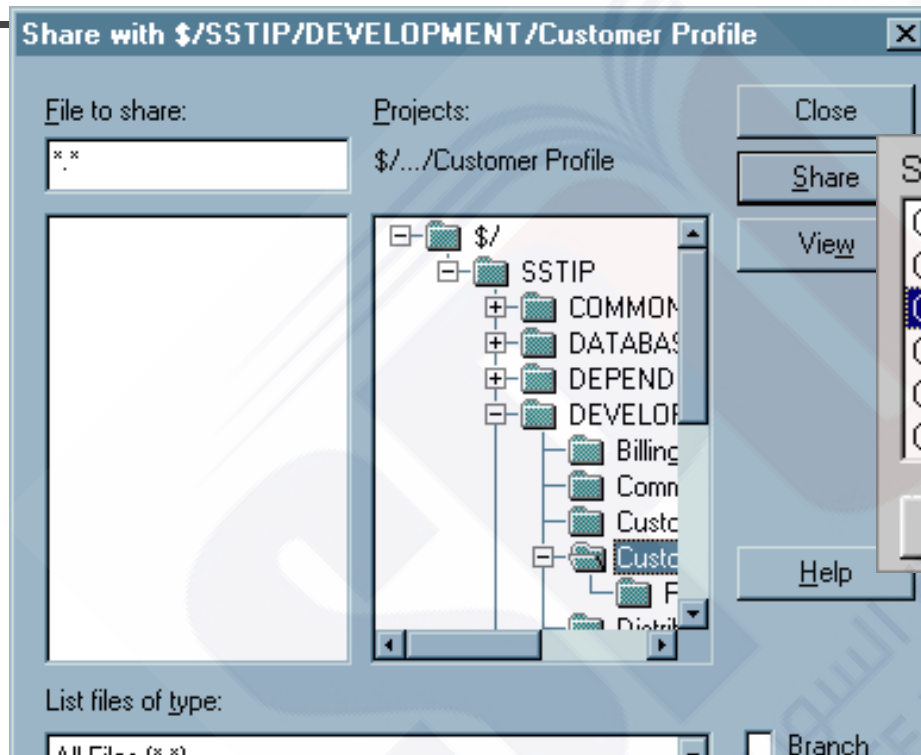# Apple Mac user interfaces

UI Hall of Shame

Layout and color

Message Tips

- Should be clear, concise, and complete
- Should be grammatically correct and free of jargon and abbreviations (unless they are the users)
- Avoid negatives and humor

Types of Messages

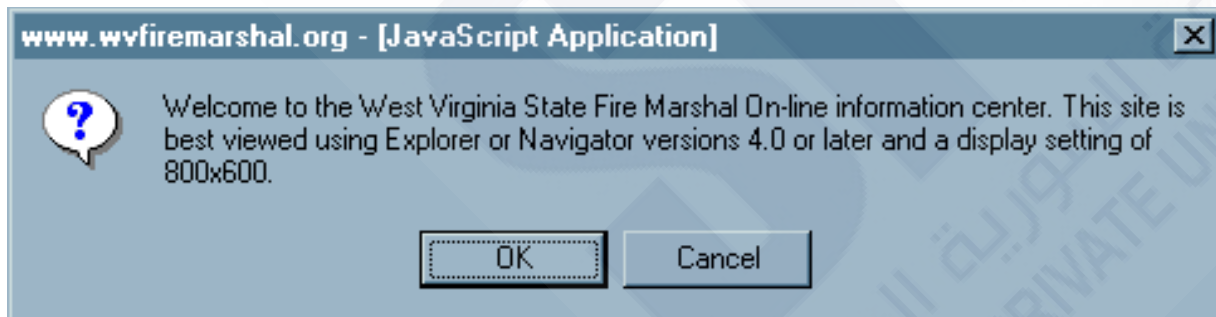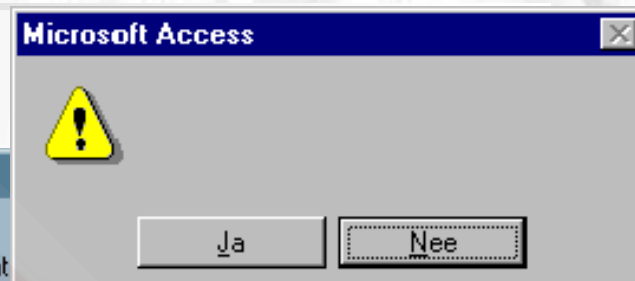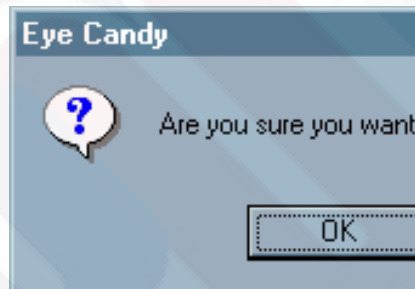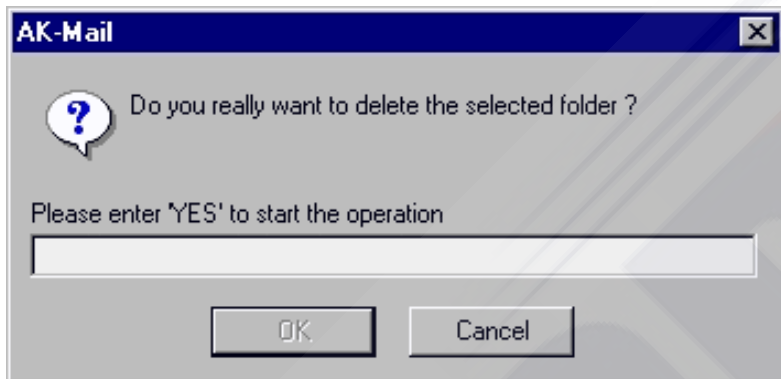| Type of Messages	When to Use	Notes
Error message Informs the user that he or she has attempted to do something to which the system cannot respond	When user does something that is not permitted or not possible	Always explain the reason and suggest corrective action. Traditionally, error messages have been accompanied by a beep, but many applications now omit it or permit users to remove it.
Confirmation message Asks the user to confirm that he or she really wants to perform the action selected	When user selects a potentially dangerous choice, such as deleting a file	Always explain the cause and suggest possible action. Often include several choices other than "OK" and "cancel."
Acknowledgment message Informs the user that the system has accomplished what it was asked to do	Seldom or never; users quickly become annoyed with all the unnecessary mouse clicks	Acknowledgment messages are typically included because novice users often like to be reassured that an action has taken place. The best approach is to provide acknowledgment information without a separate message on which the user must click. For example, if the user is viewing items in a list and adds one, then the updated list on the screen showing the added item is sufficient acknowledgment.
Delay message Informs the user that the computer system is working properly	When an activity takes more than seven seconds	This message should permit the user to cancel the operation in case he or she does not want to wait for its completion. The message should provide some indication of how long the delay may last.
Help message Provides additional information about the system and its components	In all systems	Help information is organized by table of contents and/or keyword search. *Context-sensitive help* provides information that is dependent on what the user was doing when help was requested. Help messages and on line documentation are discussed in Chapter 13.

Bad error messages

AK-Mail

Do you really want to delete the selected folder?

Please enter 'YES' to start the operation

[   ]

OK   Cancel

Eye Candy

Are you sure you want

OK

Microsoft Access

Ja   Nee

---

www.wvfiremarshal.org - [JavaScript Application]

Welcome to the West Virginia State Fire Marshal On-line information center. This site is best viewed using Explorer or Navigator versions 4.0 or later and a display setting of 800x600.

OK   Cancel

Microsoft Access

Wrong button!
This button doesn't work.

Solution
Try another.

OK

Dialog

CuteFTP is currently working. If you press Disconnect, the session will be interrupted. Do you want to disconnect?

☐ Don't show this dialog again

OK   Help

Document Wizard Result

HTML

Conversion complete!

Press View Result to view resulting documenation.

View Result

47

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 <u>M</u>nemonics 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)
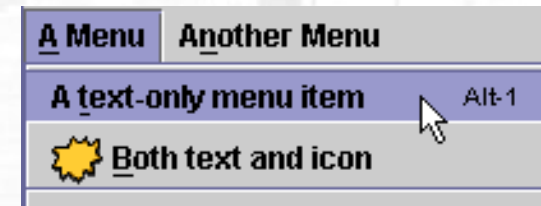
Checkboxes, radio buttons

- 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).
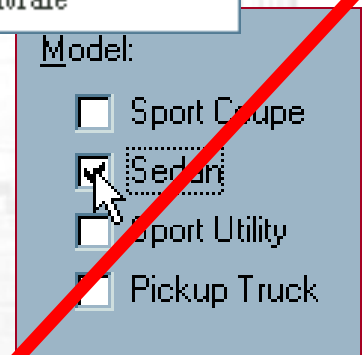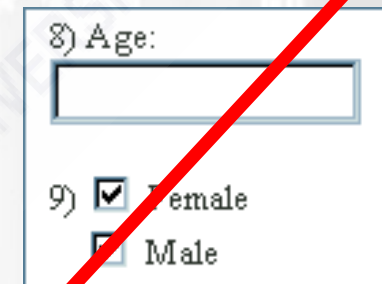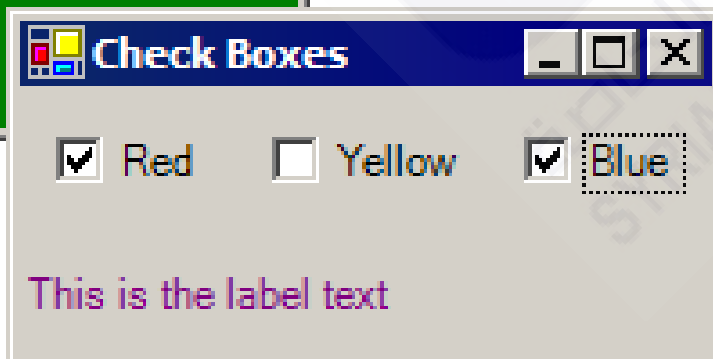
Lists, combo boxes

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

  Years: 30

- 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

  January
  February
  March
  April

- 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

  Pig
  Bird
  Cat
  Dog
  Rabbit
  Pig

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

  Frames Per Second
  0   10   20
  20

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)

LIBSYS: Search

Choose collection: [All ▲▼]

Word or phrase: [ ]

Search by: [Title ▲▼]
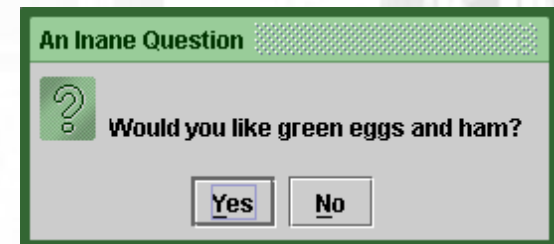
Adjacent words   ● Yes   ○ No

[OK]   [Default]   [Cancel]

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
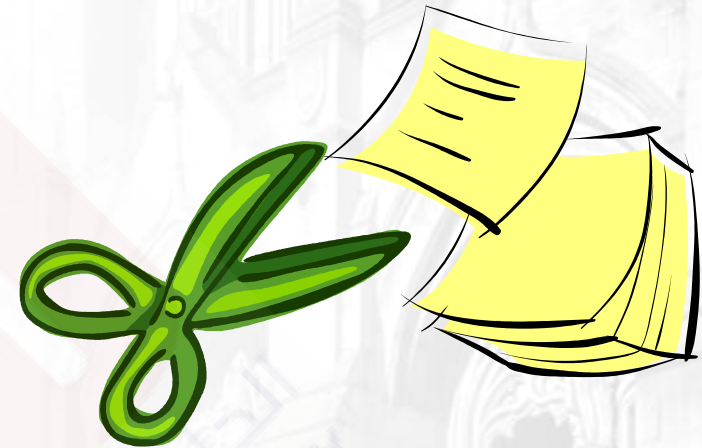
Creating a paper prototype

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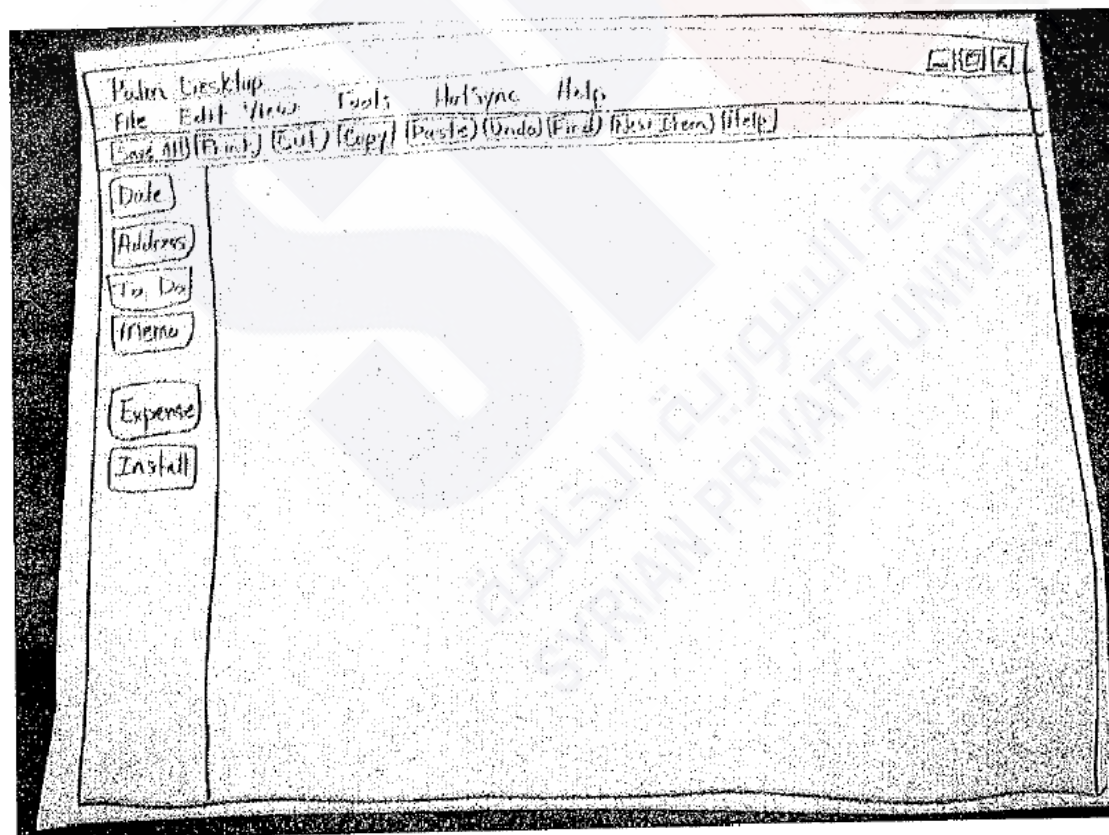 - tape, scissors
  - highlighters, transparencies

- identify the screens in your UI
  - consider use cases, inputs and outputs to user

- think about how to get from one screen to next
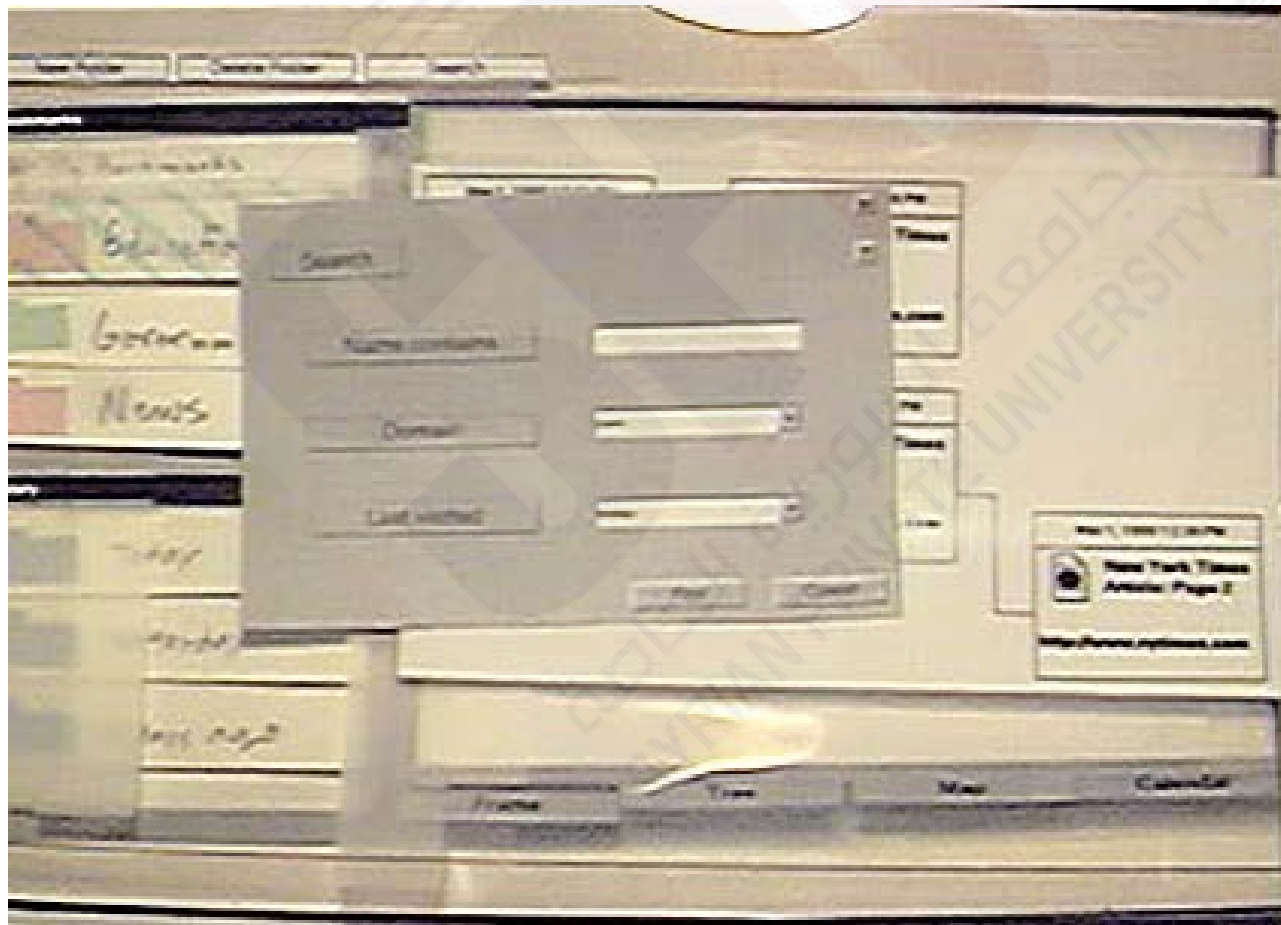 - this will help choose between tabs, dialogs, etc.

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
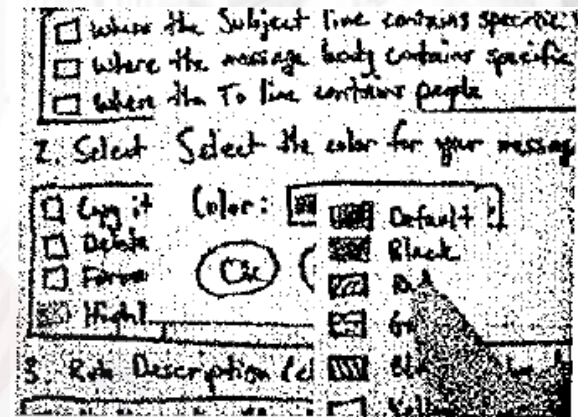
Representing a changing UI

- 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: say "beep" (hah!)

Example paper prot. screen

Example full paper prototype