Design Principles And Usability Heuristics

You can avoid common design pitfalls by following these rules-of-thumb for design

You can inspect an interface for usability problems by applying these same principles

Design Principles And Usability Heuristics

Heuristics are broad "rules of thumb" that describe features of "usable" systems:

1) Heuristics can be used as a design guide before the system is built.
2) Heuristics can be used as an evaluation mechanism after the system has been completed. The same principles can be used to "evaluate" a system for usability problems

Becoming quite popular:

1. User involvement is not required.
2. Can be applied without a great deal of prior training or time (another example of discount usability) yet it catches many design flaws.

Design Principles And Usability Heuristics (Advantages)

- **A few general guidelines can correct for the majority of usability problems**
- **Easily remembered, easily applied with modest effort**
- **A cheap and fast way to inspect a system**
- **It can be conducted by people who are not usability experts**

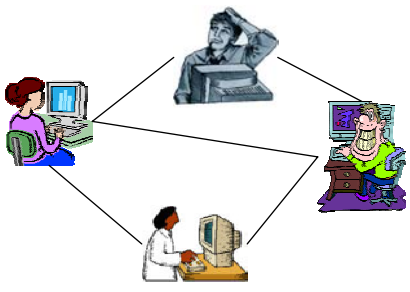
Design Principles And Usability Heuristics (Weaknesses)

Principles are more or less at the motherhood level

- Can't be treated as a simple checklist
- Subtleties involved in their use
- A very broad approach that may not cover specific situations
  - Example:

Heuristics for a single user

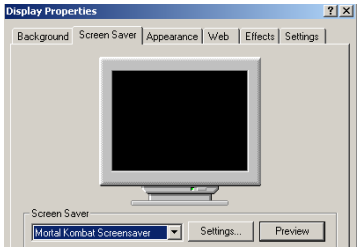
Heuristics for many collaborating users

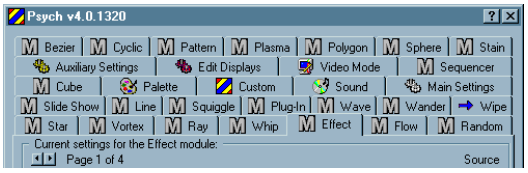


???

1. Simple And Natural Dialogue

Avoid making the interaction unnecessarily complex



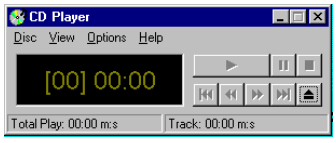


1. Simple And Natural Dialogue

Match the user's conceptual model

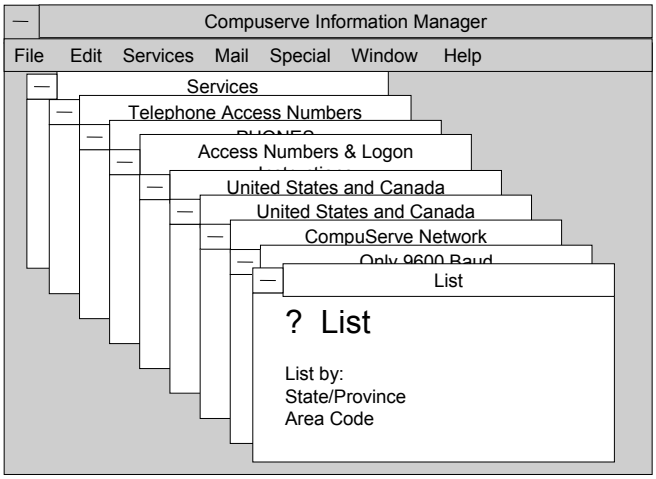
Match the users' task in as natural a way as possible

- Minimize mapping between interface and task semantics



1. Simple And Natural Dialogue

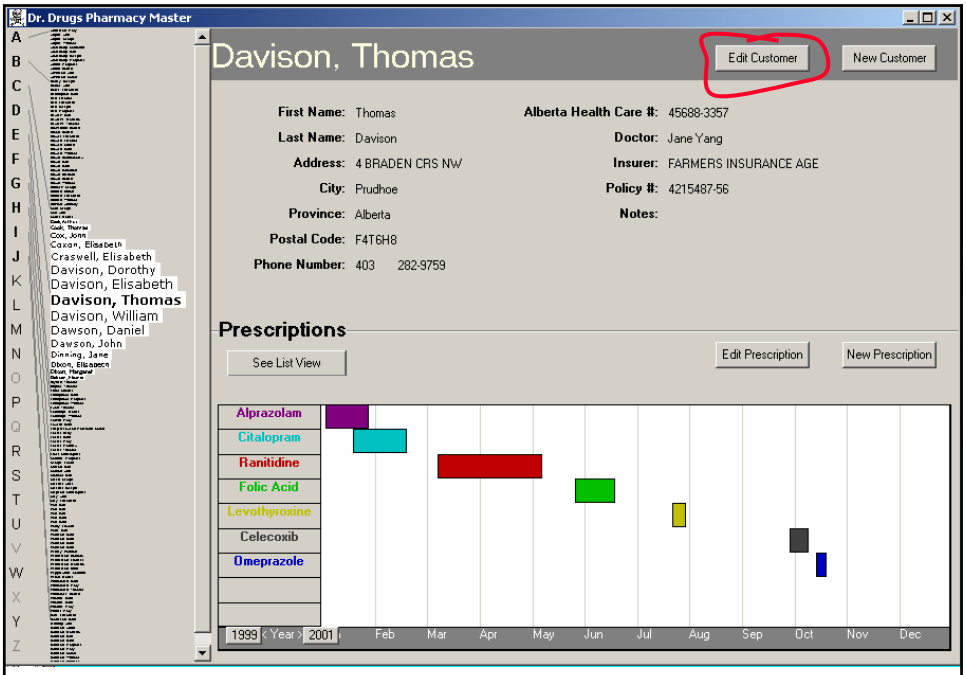
Menu or window? Which window?



1. Simple And Natural Dialogue

Present exactly the information the user needs when it is needed

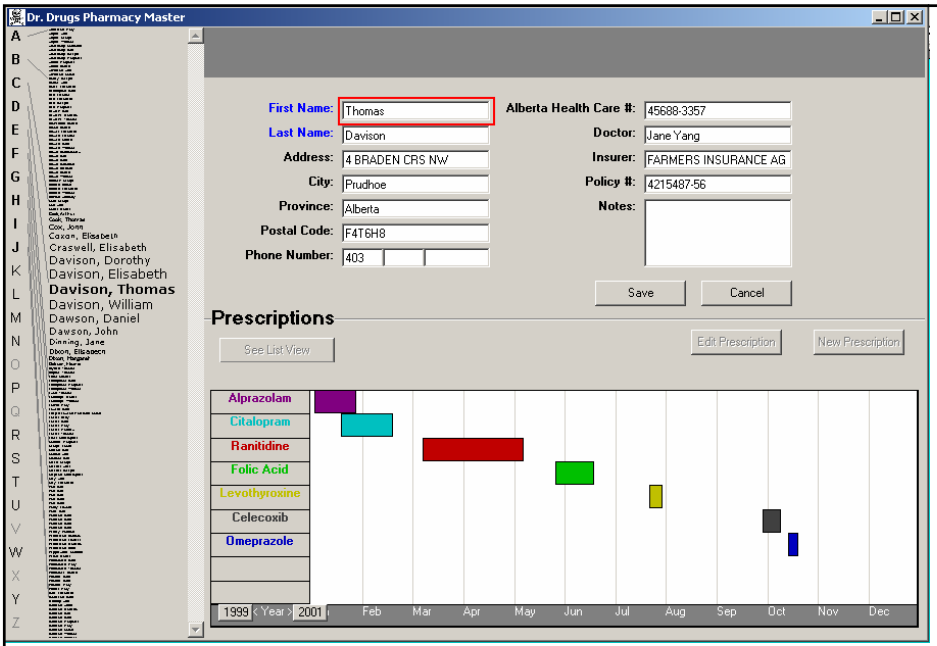
- Less is more
  - Less to learn, to get wrong, to distract...
- Remove or hide irrelevant or rarely needed information
  - Competes with important information on screen
- Information should appear in natural order
  - Order of accessing the information matches the user's expectations
  - Related information is graphically clustered
- Minimize or mitigate modes
- Use windows frugally
  - Don't make navigation and window management excessively complex



Good: Info in the same place

By previous 481 students Brant LeClercq, Lloyd Yoon, Amy Yang

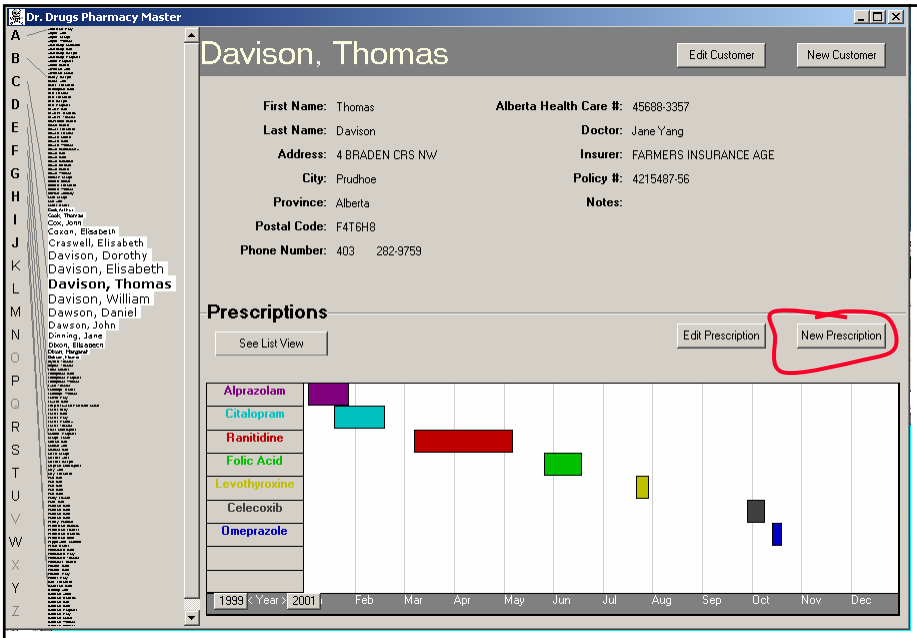
James Tam



Good: Info in the same place
Bad: Special edit mode

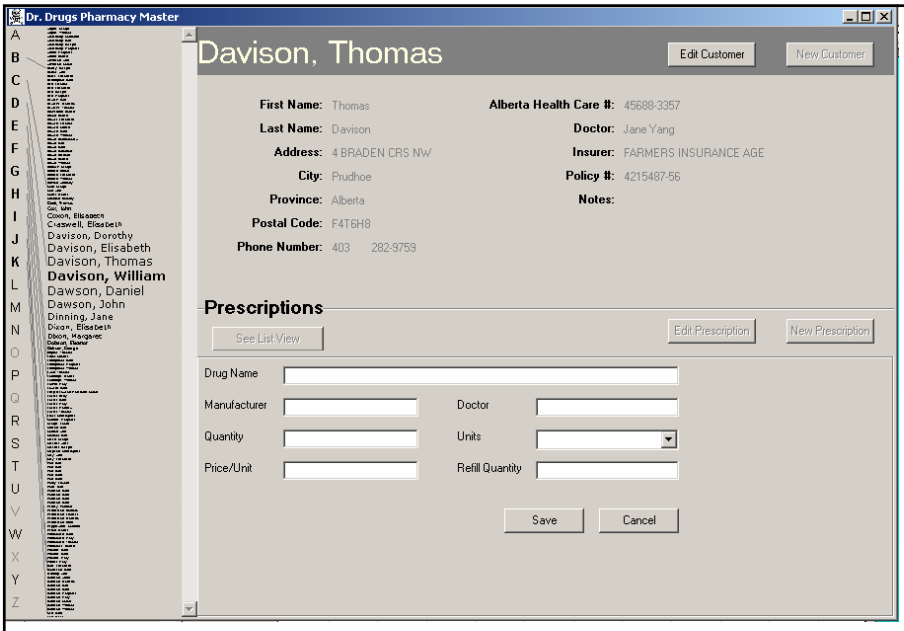
By previous 481 students Brant LeClercq, Lloyd Yoon, Amy Yang (with permission)

James Tam



By previous 481 students Brant LeClercq, Lloyd Yoon, Amy Yang (with permission)

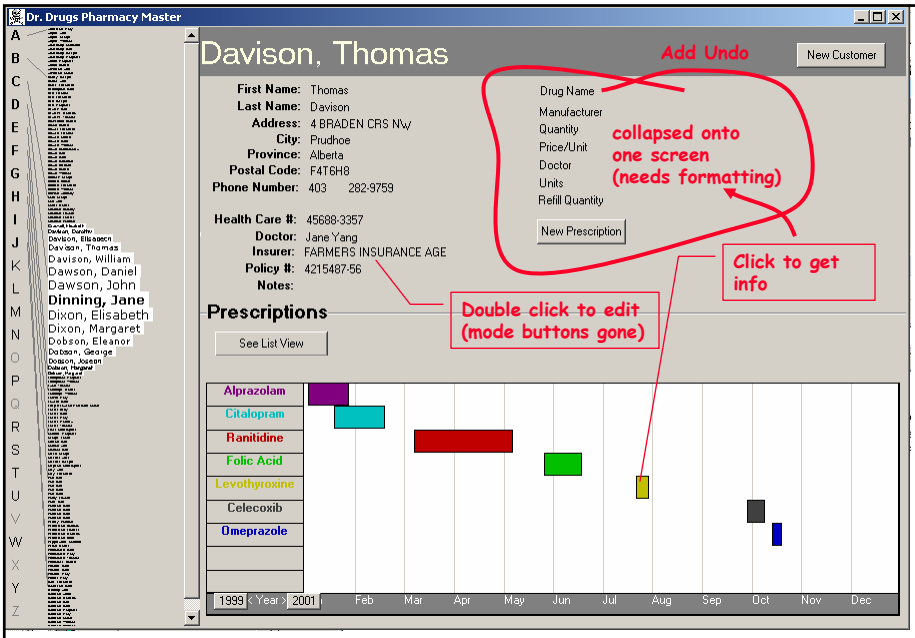
James Tam



Good: Stable parts of the window
Bad: Prescriptions separate from graphics

By previous 481 students Brant LeClercq, Lloyd Yoon, Amy Yang (with permission)

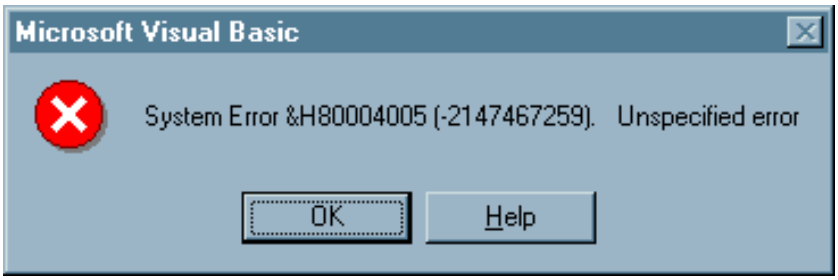
James Tam



By previous 481 students Brant LeClercq, Lloyd Yoon, Amy Yang (with permission)

James Tam

2. Speak The User's Language

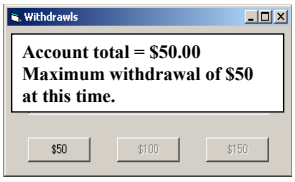


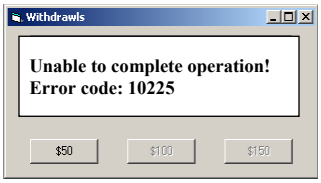
Visual Studio © Microsoft

James Tam

2. Speak The User's Language

Terminology based on users' language for task





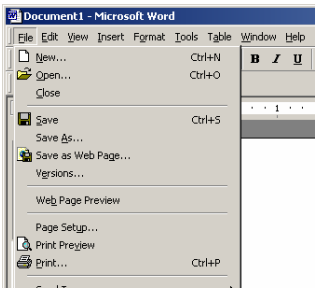
Use meaningful mnemonics, icons, and abbreviations

- eg File / Save
  - Ctrl + S (abbreviation)
  - Alt FS (mnemonic for menu action)
  - Open folder (tooltip icon)

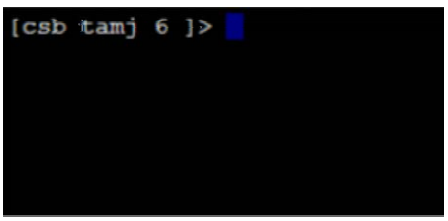
3. Minimize The User's Memory Load

•Describe required the input format, use examples, provide default inputs

•Promote recognition over recall



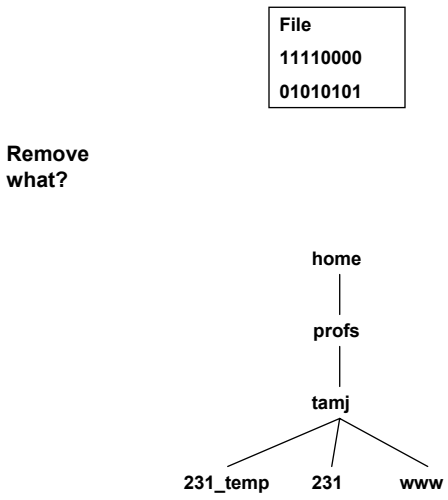
Recognition



Recall

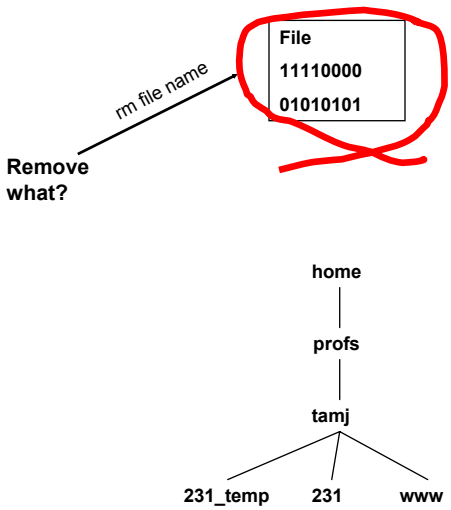
3. Minimize The User's Memory Load

Small number of rules applied universally



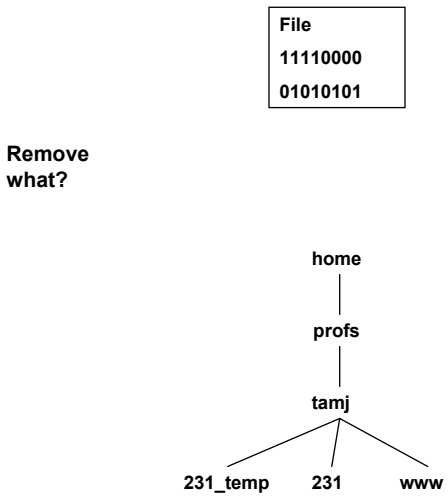
3. Minimize The User's Memory Load

Small number of rules applied universally



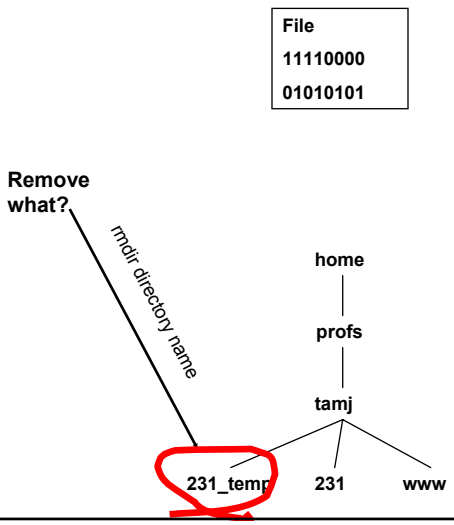
3. Minimize The User's Memory Load

Small number of rules applied universally



3. Minimize The User's Memory Load

Small number of rules applied universally



4. Be Consistent

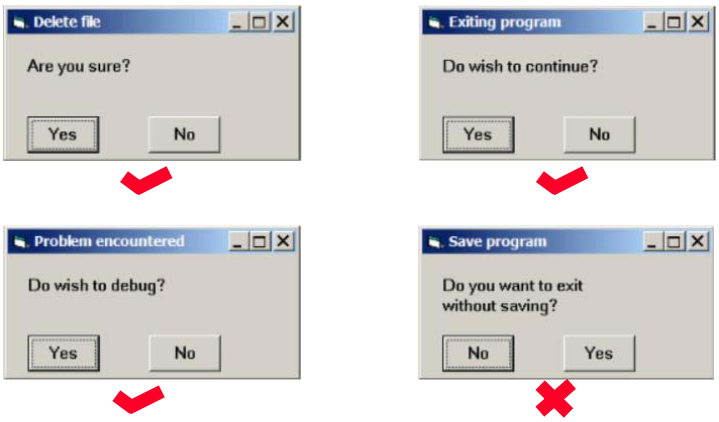
Consistency of effects

- Same words, commands, actions will always have the same effect in equivalent situations
- Makes the system more predictable
- Reduces memory load

4. Be Consistent

Consistency of language and graphics

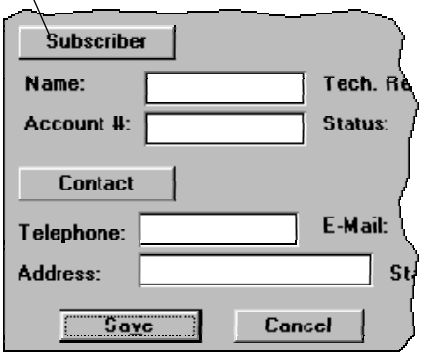
- Same information/controls in same location on all screens / dialog boxes forms follow boiler plate.
- Same visual appearance across the system (e.g. widgets).



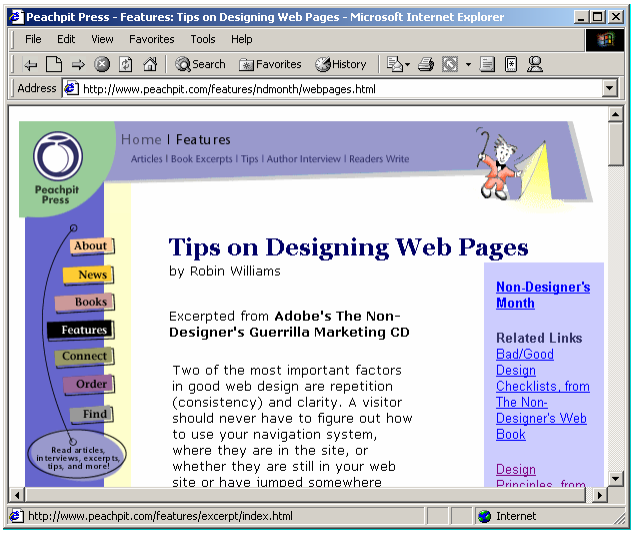
4. Be Consistent

These are labels with a raised appearance.

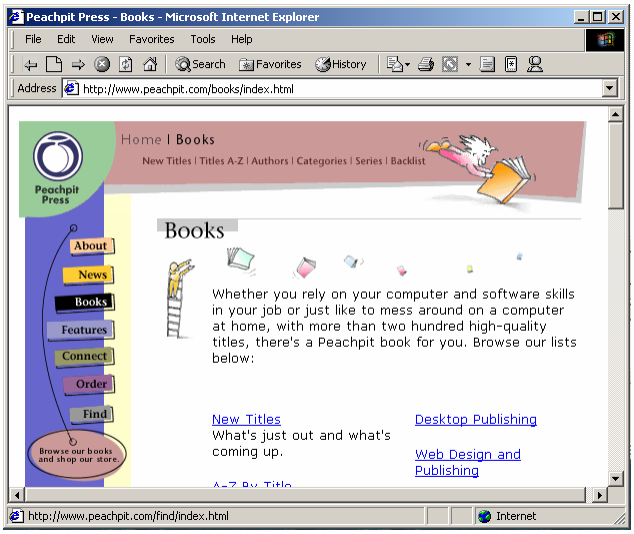
Is it any surprise that people try and click on them?



4. Be Consistent

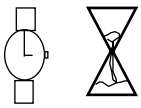


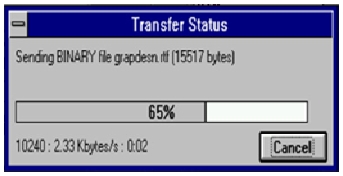
4. Be Consistent

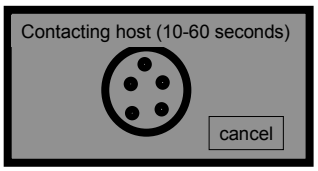


5. Provide Feedback

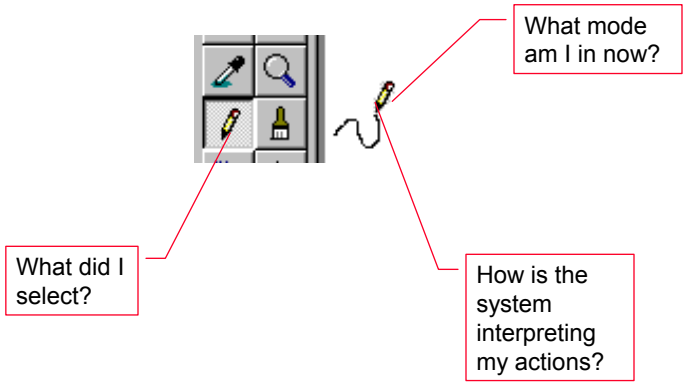
What is the program doing?







5. Provide Feedback

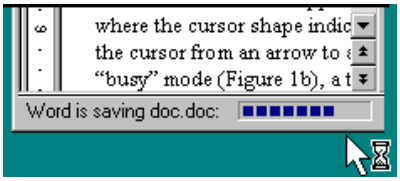


James Tam

5. Provide feedback

Should be as specific as possible, based on user's input

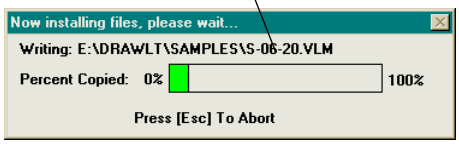
Best within the context of the action



James Tam

5. Provide feedback

Multiple files being copied,
but feedback is file by file.



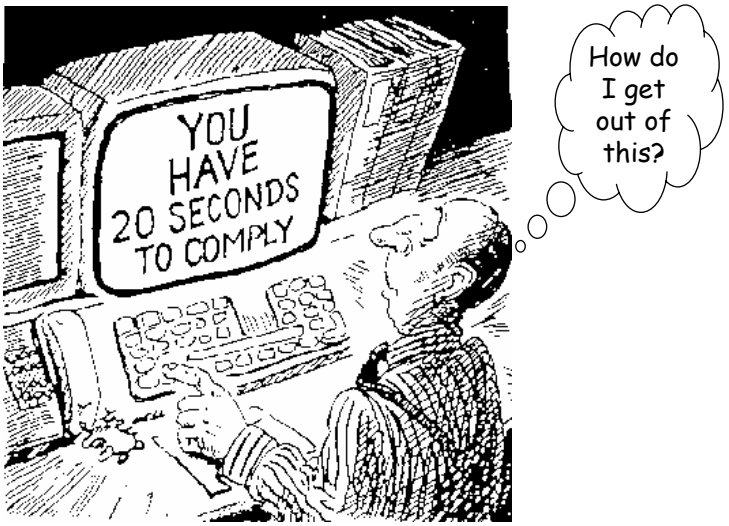
5. Provide feedback

Feedback response time

- How users perceive delays

Delay	Effect
0<= delay <= 0.1 seconds	Delay not noticed (system appears to operate instantaneously)
0.1 < delay <= 1 second	Delay noticed but train of thought remains uninterrupted
1 < delay <= 10 seconds	Train of thought interrupted but person can still remain focused on the system
Delay > 10 seconds	Person wants to do something else while waiting for the system

6. Provide Clearly Marked Exits



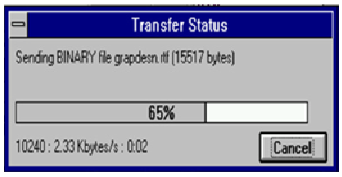
6. Provide Clearly Marked Exits

Universal Undo/Redo

- e.g., <Ctrl>-<Z> and <Ctrl> <Y>

Progress indicator & Interrupt

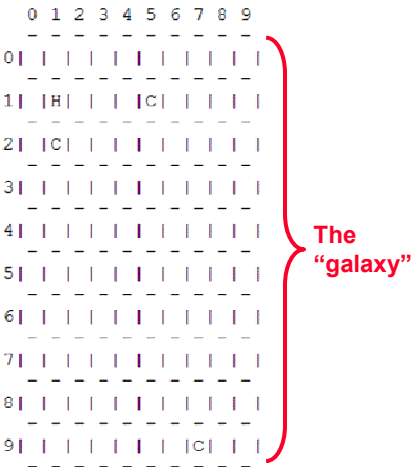
- Length operations

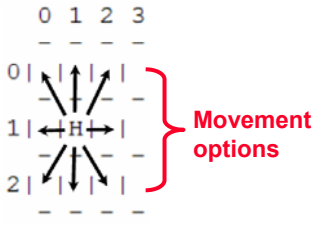


6. Provide Clearly Marked Exits

This can apply even with systems that are written for 'experts'

- Example: A text-based space simulation game





6. Provide Clearly Marked Exits

```
do
{
        insideGalaxy = isEmpty = withinRange = true;
        System.out.print("Enter the destination row (0 – 9): ");
        row = Console.in.readInt();
        System.out.print("Enter the destination column(0 – 9): ");
        column = Console.in.readInt();
       if ((row < 0) || (row > (Galaxy.SIZE-1)) || (column < 0) || (column > (Galaxy.SIZE-1)))
        {
           insideGalaxy = false;
           System.out.println("You cannot move outside the galaxy");
        }
            .                    .
} while ((insideGalaxy == false) || (isEmpty == false) || (withinRange == false));
```

Loop while input is invalid

6. Provide Clearly Marked Exits

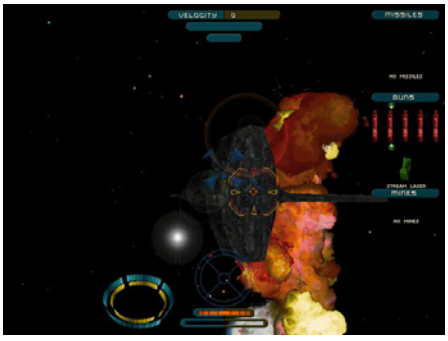
```
do
{
        insideGalaxy = isEmpty = withinRange = true;
        System.out.print("Enter the destination row (0 – 9, -1 to pass): ");
        row = Console.in.readInt();
        if (row == -1)
        {
           System.out.println("You have chosen not to move your ship.");
           break;
        }
        System.out.print("Enter the destination column(0 – 9, -1 to pass): ");
        column = Console.in.readInt();
            :              :
} while ((insideGalaxy == false) || (isEmpty == false) || (withinRange == false));
```

Allow the user an option to quit

6. Provide Clearly Marked Exits

Restoring defaults

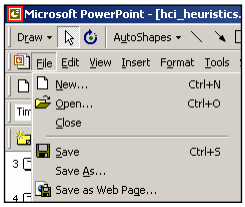
- Getting back original settings



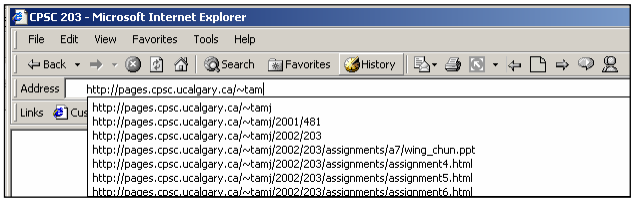
Wing Commander: Privateer 2 © Origin-EA

7. Provide Shortcuts

Keyboard accelerators



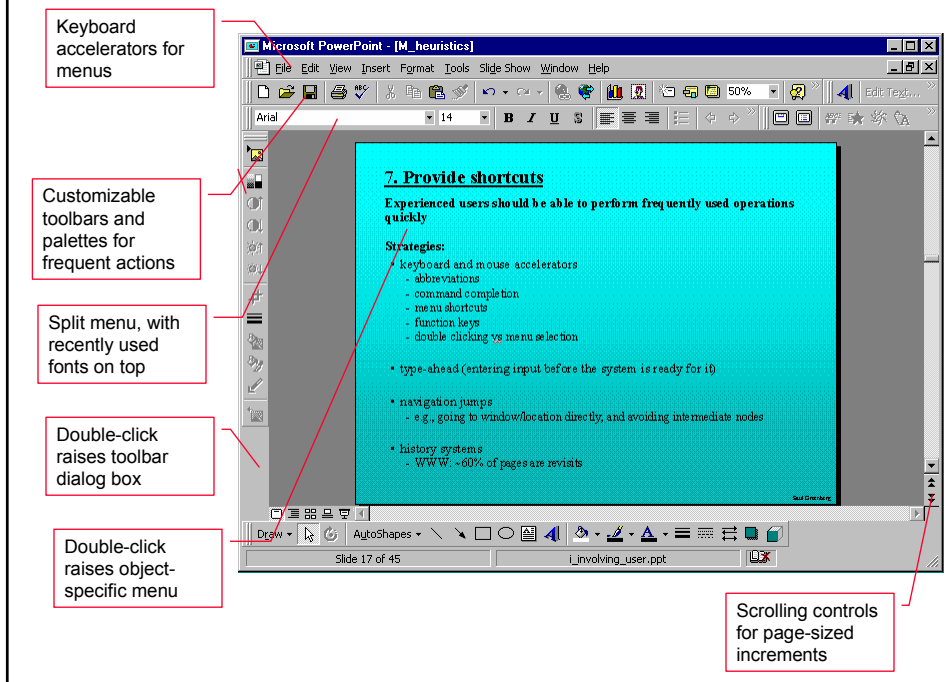
Name completion



7. Provide Shortcuts

Experienced (power) users should be able to perform frequently used operations quickly

- Type-ahead (entering input before the system is ready for it)
- Navigation jumps
  - e.g., going to window/location directly, and avoiding intermediate nodes
- History systems
  - WWW: ~60% of pages are revisits

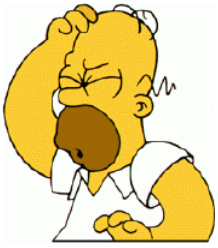


PowerPoint © Microsoft

James Tam

8. Deal With Errors In A Helpful And Positive Manner

People will make errors!



Errors we make

- Mistakes
- Slips

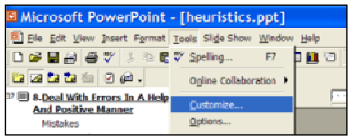
Image and sound file from "The Simpson's"

James Tam

8. Deal With Errors In A Helpful And Positive Manner

Mistakes

- Arise from ***conscious deliberations*** that lead to an error instead of the correct solution



Slips

- ***Unconscious behavior*** that gets misdirected en route to satisfying goal
- Shows up frequently in skilled behavior
  - Usually due to inattention
- Often arises from similarities of actions

```
[csb 481 16 ]> rm *
rm: remove file1 (yes/no)? y
rm: remove file2 (yes/no)? y
rm: remove file3 (yes/no)? y
rm: remove file4 (yes/no)? y
rm: remove file5 (yes/no)?
```

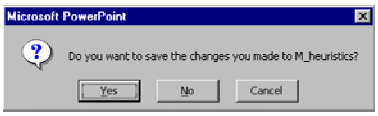

Types Of Slips

1) **Capture error**
2) **Description error**
3) **Loss of activation**
4) **Mode error**

Types Of Slips: Capture Error

The frequently done activity "captures" the intended activity

- Occurs when common and rarer actions have same initial sequence
- "Force of habit results in the slip"



James Tam

Types Of Slips: Description Error

Intended action has much in common with others that are possible

- Usually occurs when right and wrong objects physically near each other
- "Perform the right action but on the wrong object"
  - e.g., Pour juice into coffee cup instead of glass

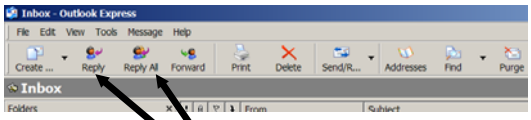
www.baddesigns.com

James Tam

Types Of Slips: Description Error (2)

PowerPoint

Print preview??



Outlook Express

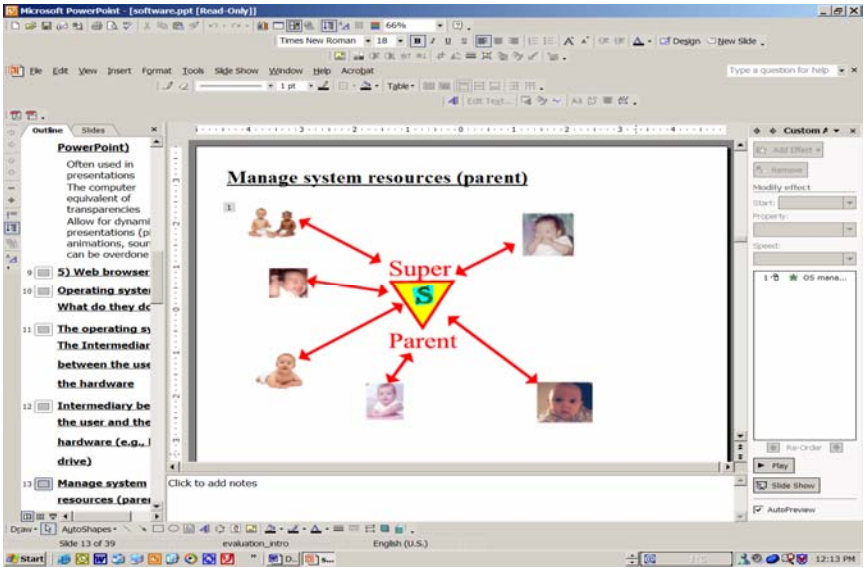
Oops! I only meant for that letter to only go to one person.

Types Of Slips: Loss Of Activation

Loss of activation

- Forgetting what the goal is while undergoing the sequence of actions
  - e.g., Start going to room and forget why you are going there
- "What the heck was I doing again?"
- Misordering a sequence

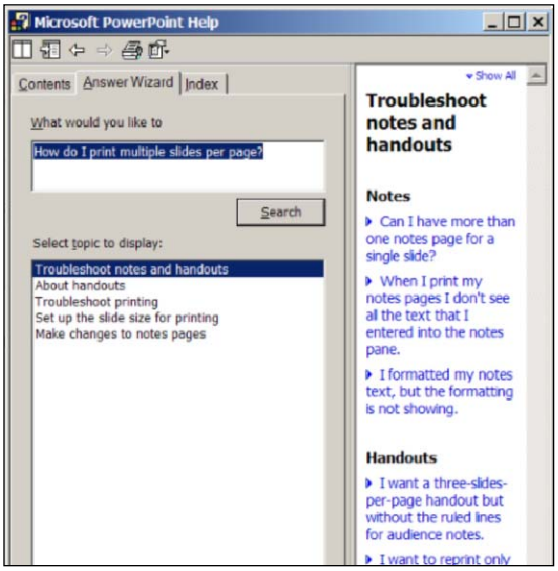
Loss Of Activation



PowerPoint © Microsoft

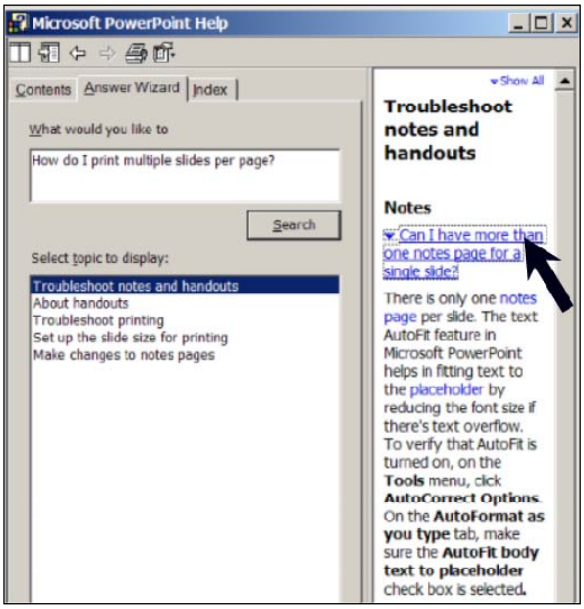
James Tam

Loss Of Activation (2)



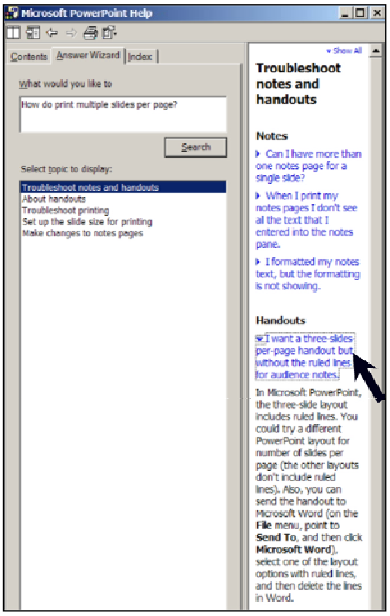
James Tam

Loss Of Activation (3)



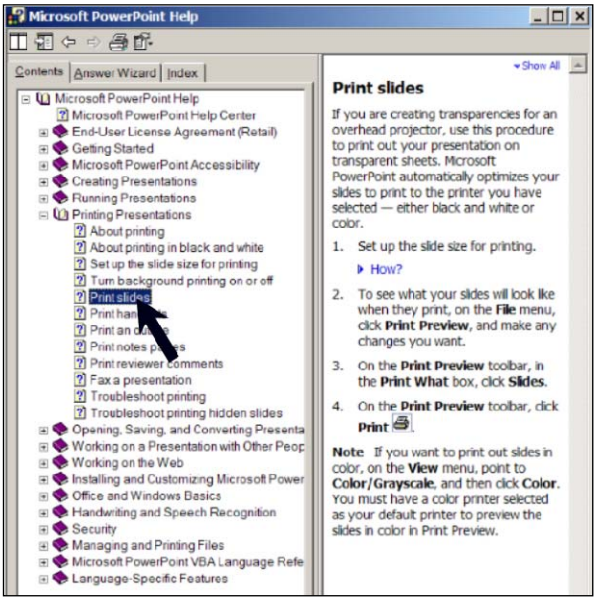
James Tam

Loss Of Activation (4)

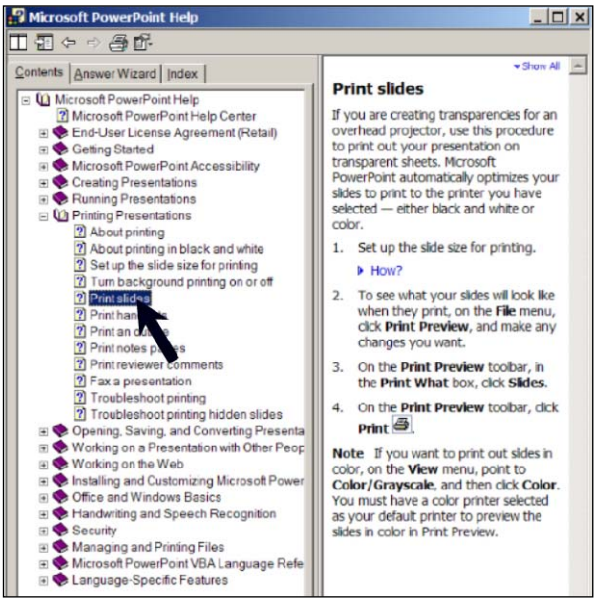


James Tam

…Loss Of Activation (???)



…Loss Of Activation (???)

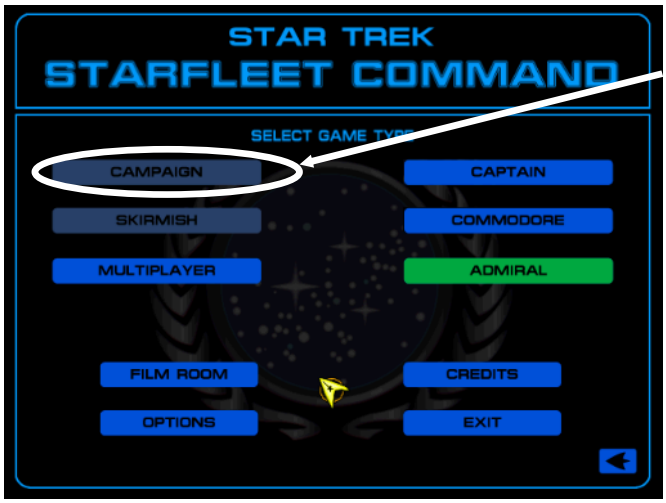


What was I trying to do again???

Types Of Slips: Mode Error

Mode errors

- People do actions in one mode thinking they are in another mode



Game is in multiplayer mode only (cannot load saved games)

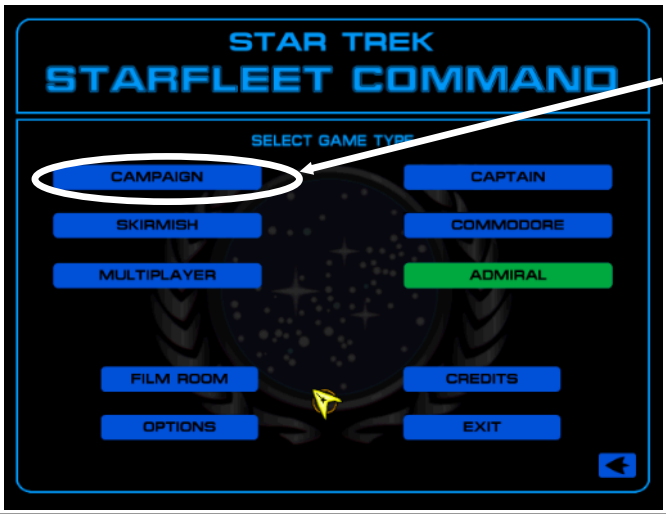
Star Fleet Command © Interplay Productions

James Tam

Types Of Slips: Mode Error (2)

Mode errors

- People do actions in one mode thinking they are in another mode



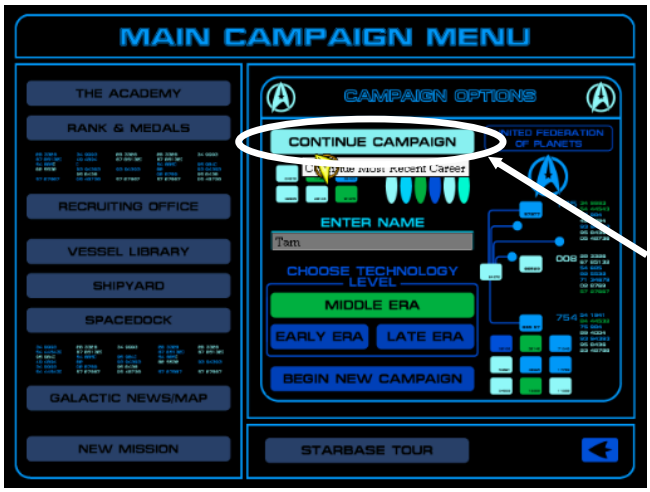
Game is in single and multiplayer mode (can start new campaigns or load existing games)

James Tam

Types Of Slips: Mode Error (3)

Mode errors

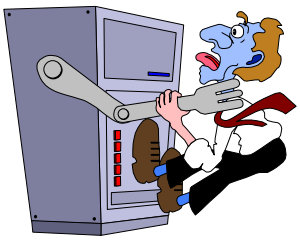
- People do actions in one mode thinking they are in another mode



In this mode it is possible to load a game

James Tam

Designing For Slips



General rules

- Prevent slips before they occur
- Detect and correct slips when they do occur
- User correction through feedback and undo

Examples

- Capture errors
  - One action captures another
  - Allow actions to be undone rather than asking for confirmations
- Description errors
  - Correct action on the wrong object
  - Make objects look physically distinctive and/or far apart
- Loss of activation
  - Forgot goal
  - If system knows goal, make it explicit
  - If not, allow person to see path taken or shorten steps
- Mode errors
  - Mistake modes
  - Have as few modes as possible (possibly none)
  - Make modes highly visible

James Tam

Generic System Responses For Errors

General idea: Forcing functions

- Prevent / mitigate continuation of wrongful action
  - e.g., range selection for dynamic queries

Gag

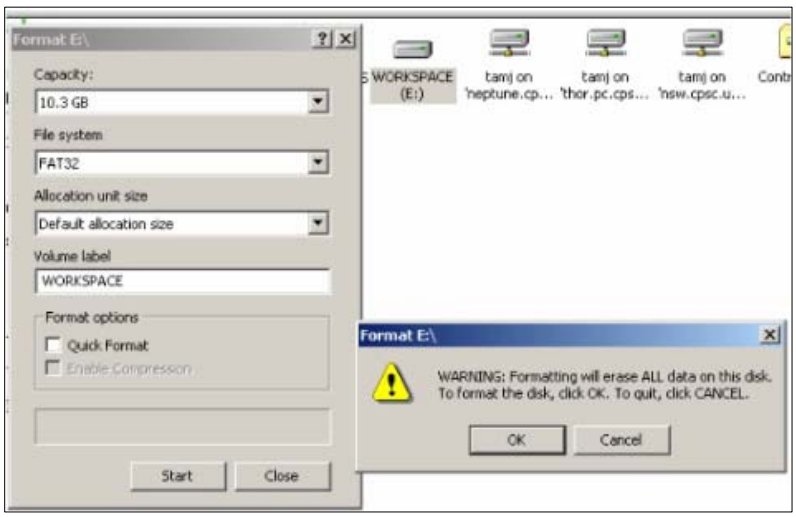
- Deals with errors by preventing the user from continuing
  - e.g., cannot get past login screen until correct password entered

Warn

- Warn people that an unusual situation is occurring
- Better than nothing but when overused, becomes an irritant
  - e.g., audible bell, alert box

Generic System Responses For Errors (2)

Warning (probably needed)



Generic System Responses For Errors (3)

Warning (can get annoying)

Generic System Responses For Errors (4)

Do nothing

- Illegal action just doesn't do anything
- User must infer what happened
  - e.g., enter letter into a numeric-only field (key clicks ignored)
  - e.g., put a file icon on top of another file icon (returns it to original position)

Self-correct

- System guesses legal action and does it instead - but leads to a problem of trust
  - e.g., spelling checkers

Lets talk about it

- System initiates dialog with user with solution to the problem
  - e.g., compile error brings up offending line in source code

Generic System Responses For Errors (5)

Teach me

- System asks user what the action was supposed to have meant
- Action then becomes a legal one

Error Handling In "The Good Ole Days"

What is "error 15762"?

Examples Of Error Handling From The "The Good Ole Days"

Below is the full list of error codes with[out] a suitable explanation to their meaning.

103: Insufficient Free store

105: Task Table Full

120: Argument line invalid or to long

121: File is not an object module

122: Invalid resident library during load

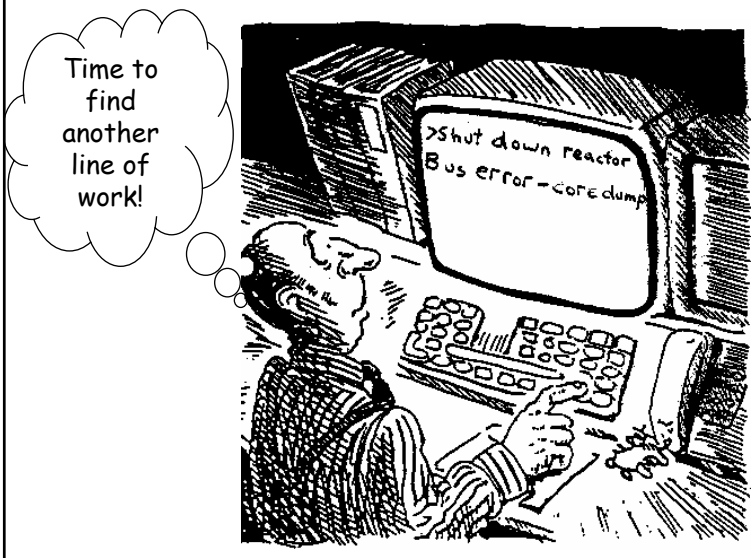
202: Object in use

203: Object already exists

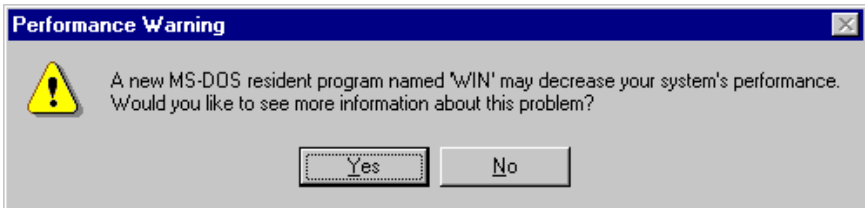
204: Directory not found

List of error codes from the AmigaDos OS

Errors: Don't Make Things Seem Worse Than They Really Are



Errors: Giving Away Too Much Information?



Windows 95 dialog box

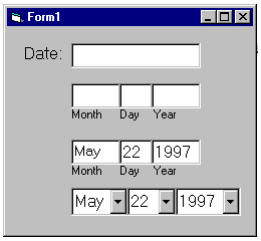
Rules Of Thumb For Error Messages

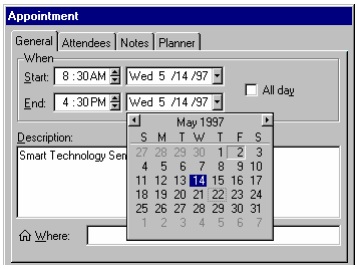
1. **Polite and non-intimidating**
   - Don't make people feel stupid
     - Try again, bonehead!
2. **Understandable**
   - Error 25
3. **Specific**
   - Cannot open this document
   - Cannot open "chapter 5" because the application "Microsoft Word" is not on your system
4. **Helpful**
   - Cannot open "chapter 5" because the application "Microsoft Word" is not on your system. Open it with "Teachtext" instead?

Examples Of Dealing With Errors In A Positive And Helpful Manner

Prevent errors

- Try to make errors impossible
- Modern widgets: only "legal commands" selected, or "legal data" entered

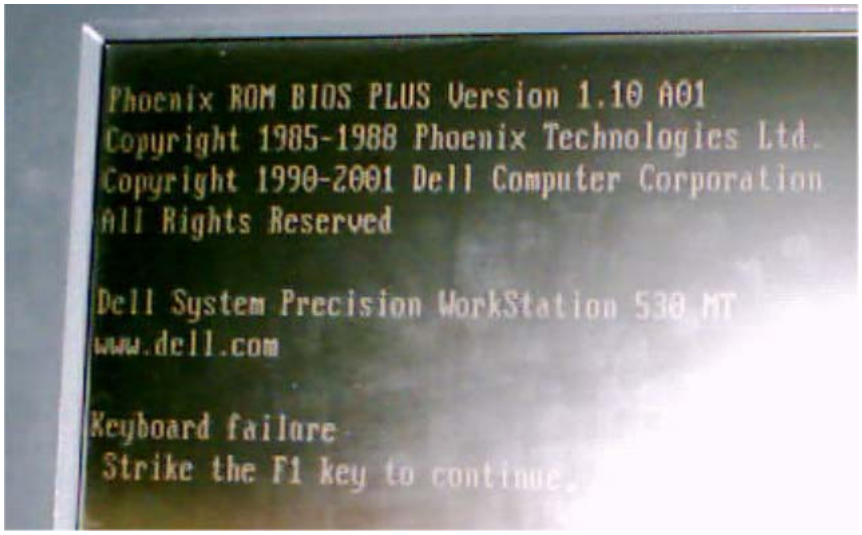




Provide reasonableness checks on input data

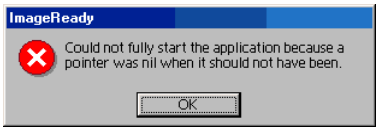
- On entering order for office supplies
  - 5000 pencils is an unusually large order. Do you really want to order that many?

I'd Rather Deal With The Any Key Instead!



Picture courtesy of James Tam

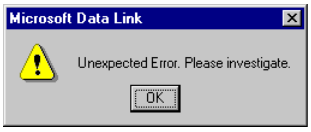
Examples Of Bad Error Messages (2)



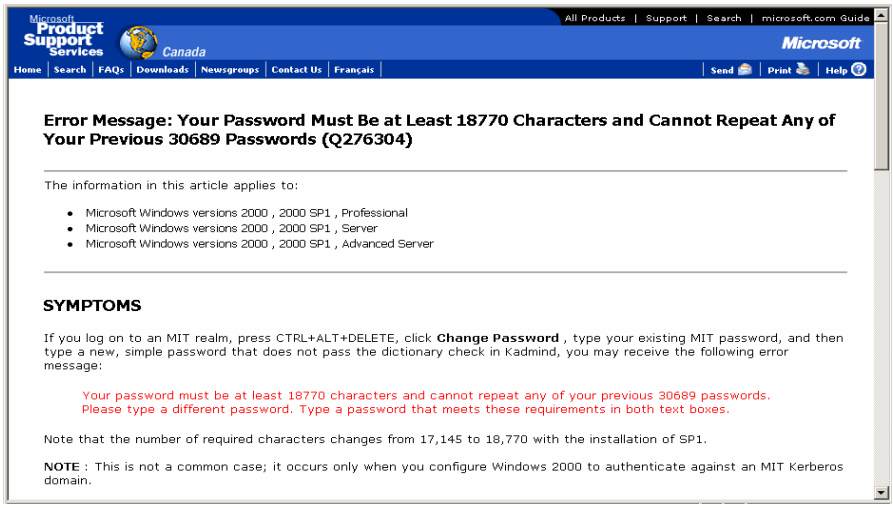
Adobe's *ImageReady*

Microsoft's *NT Operating System*

Examples Of Bad Error Messages (3)



Examples Of Bad Error Messages (4)



Microsoft Product Support Services — Canada

All Products | Support | Search | microsoft.com Guide

Microsoft

Home | Search | FAQs | Downloads | Newsgroups | Contact Us | Français

Send | Print | Help

Error Message: Your Password Must Be at Least 18770 Characters and Cannot Repeat Any of Your Previous 30689 Passwords (Q276304)

The information in this article applies to:

- Microsoft Windows versions 2000 , 2000 SP1 , Professional
- Microsoft Windows versions 2000 , 2000 SP1 , Server
- Microsoft Windows versions 2000 , 2000 SP1 , Advanced Server

SYMPTOMS

If you log on to an MIT realm, press CTRL+ALT+DELETE, click **Change Password** , type your existing MIT password, and then type a new, simple password that does not pass the dictionary check in Kadmind, you may receive the following error message:

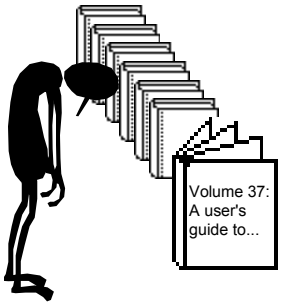
Your password must be at least 18770 characters and cannot repeat any of your previous 30689 passwords. Please type a different password. Type a password that meets these requirements in both text boxes.

Note that the number of required characters changes from 17,145 to 18,770 with the installation of SP1.

NOTE : This is not a common case; it occurs only when you configure Windows 2000 to authenticate against an MIT Kerberos domain.

9. Provide Help

Help is not a replacement for bad design!



Simple systems:

- Walk up and use; minimal instructions

Most other systems:

- Feature rich
- Some users will want to become “experts” rather than “casual” users
- Intermediate users need reminding, plus a learning path (novice to advanced)

Documentation And How It's Used

Many users do not read manuals

- Prefer to spend their time pursuing their task
- Paper manuals unavailable in many businesses!
  - e.g. single copy locked away in system administrator's office

Typical usage scenarios:

- Usually used when users are in some kind of panic, need immediate help
  - Indicates need for online documentation, good search/lookup tools
  - Online help can be specific to current context
- Sometimes used for quick reference
  - Syntax of actions, possibilities...
  - List of shortcuts ...

Types Of Help

Reference Manuals

Reminders

- Reference cards
- Keyboard templates

Context-sensitive help

- Tool tips
- What's this (Balloon help)

Wizards

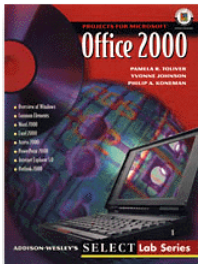
Tips

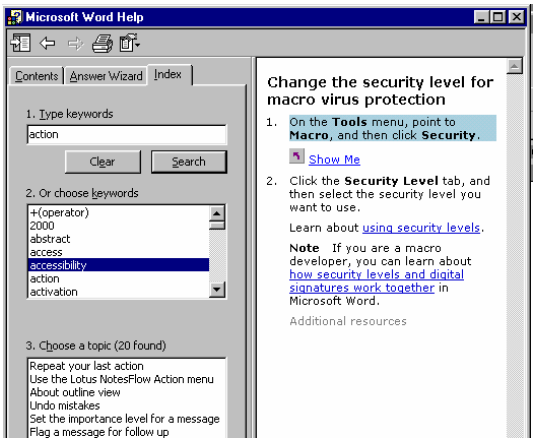
Reference Manuals

Traditional form of help

Detailed

Paper or online



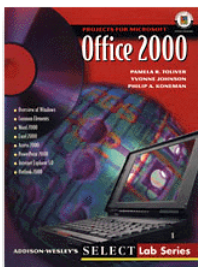


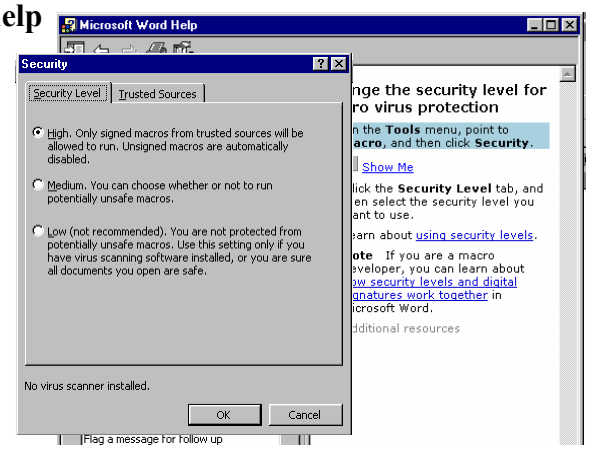
Reference Manuals

Traditional form of help

Detailed

Paper or online

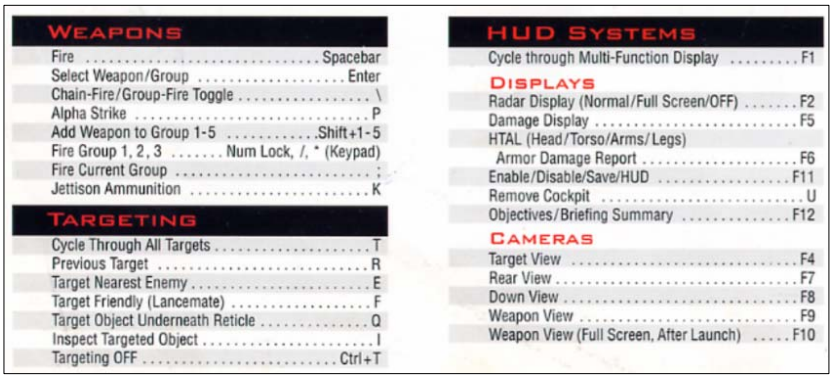




Reminders

Short reference cards

- Expert user who just wants to check facts
- Novice who wants to get overview of system's capabilities



WEAPONS

Fire	Spacebar
Select Weapon/Group	Enter
Chain-Fire/Group-Fire Toggle	\
Alpha Strike	P
Add Weapon to Group 1-5	Shift+1-5
Fire Group 1, 2, 3	Num Lock, /, * (Keypad)
Fire Current Group	;
Jettison Ammunition	K

TARGETING

Cycle Through All Targets	T
Previous Target	R
Target Nearest Enemy	E
Target Friendly (Lancemate)	F
Target Object Underneath Reticle	Q
Inspect Targeted Object	I
Targeting OFF	Ctrl+T

HUD SYSTEMS

Cycle through Multi-Function Display	F1

DISPLAYS

Radar Display (Normal/Full Screen/OFF)	F2
Damage Display	F5
HTAL (Head/Torso/Arms/Legs) Armor Damage Report	F6
Enable/Disable/Save/HUD	F11
Remove Cockpit	U
Objectives/Briefing Summary	F12

CAMERAS

Target View	F4
Rear View	F7
Down View	F8
Weapon View	F9
Weapon View (Full Screen, After Launch)	F10

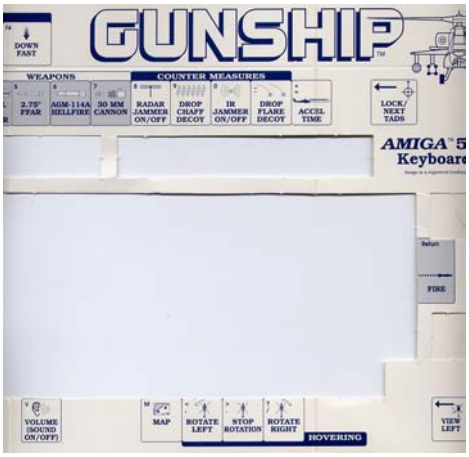
From "Mechwarrior 2: Mercenaries" © Microprose

James Tam

Reminders (2)

Keyboard templates

- Shortcuts/syntactic meanings of keys: recognition vs. recall.

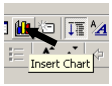


From "Gunship" © Microprose

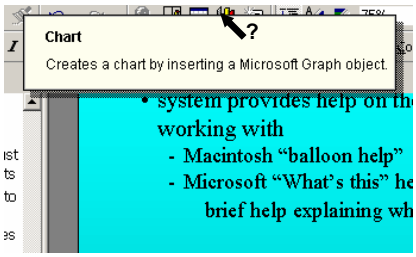
James Tam

Context-Sensitive Help

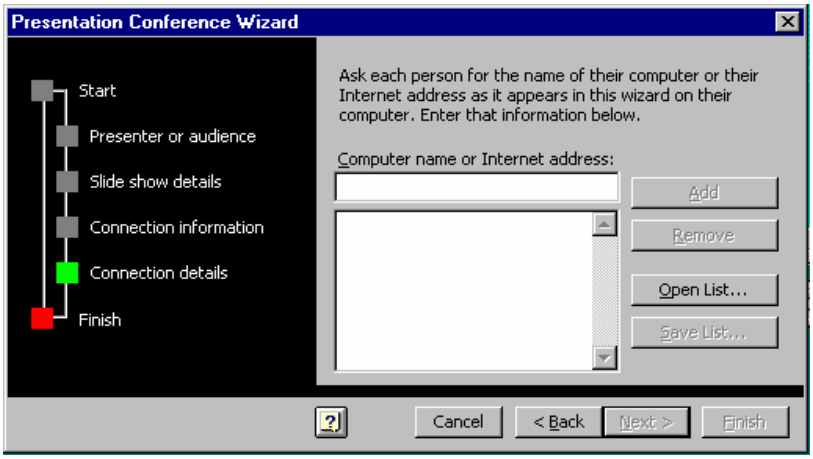
Tool tips



What's this (Windows) or Balloon help (Mac)

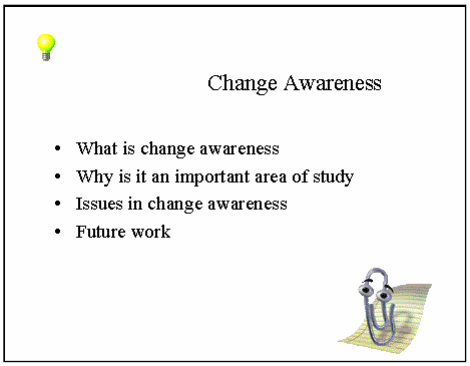


Wizards



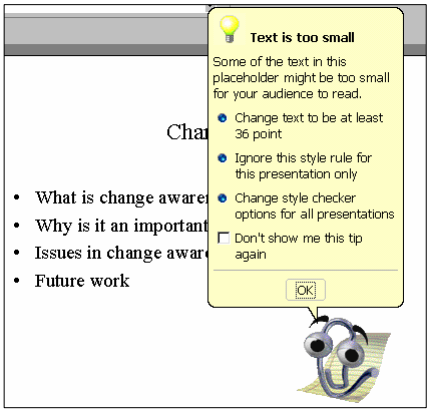
Tips

- **Introduce advanced features**
- **Point out incorrect use**



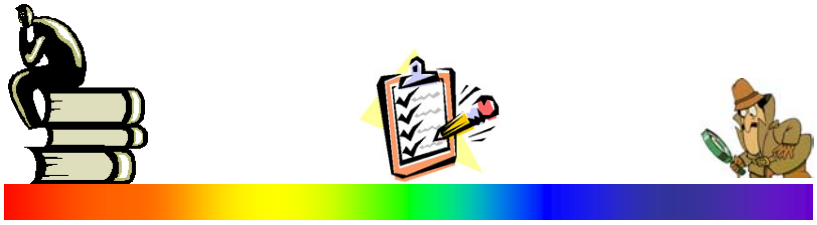
Tips

- **Advanced features**
- **Point out incorrect use**



Conducting A Heuristic Evaluation

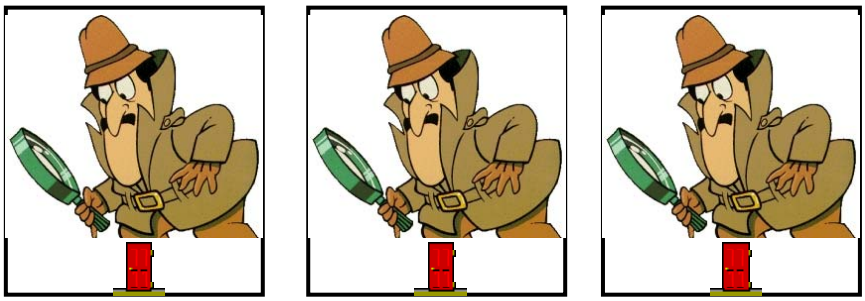
It's a compromise between extensive style guides and intuition-based inspections



Style guides | Heuristic evaluation | Inspections

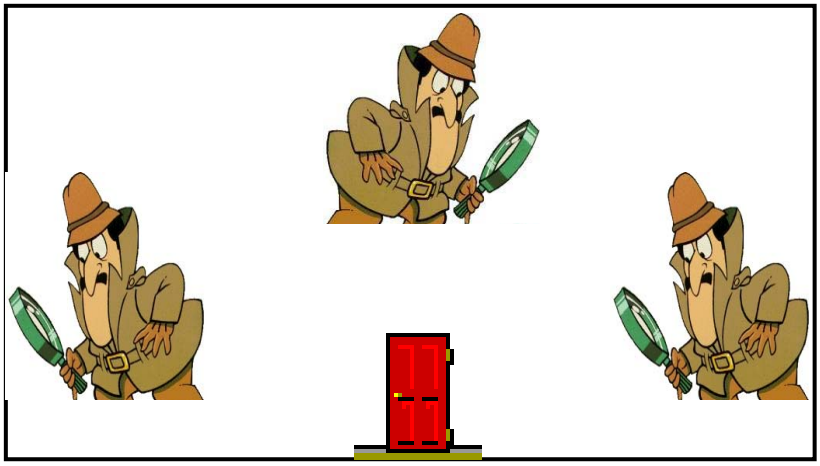
Conducting A Heuristic Evaluation (2)

- **Employ a small set of evaluators (3–5) examine interface using heuristics as a structure**
- **Each person evaluates the system according to the heuristics individually (~1 – 2 hours)**



Conducting A Heuristic Evaluation (3)

- **After this bring the evaluators together to pool/compare their results**

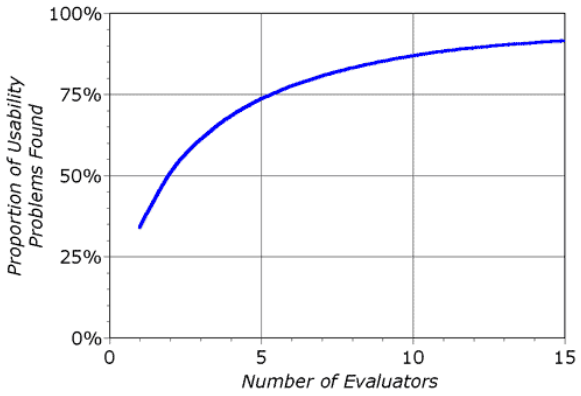


James Tam

Conducting A Heuristic Evaluation (4)

Benefits of group evaluations

- Single evaluator only catches ~35% of usability problems
- 5 evaluators catch ~75%



Statistics from "Usability Engineering" by Jakob Nielsen

James Tam

Heuristic Evaluations: Who Should/Could Conduct Them

Interface experts

End users

Software development team

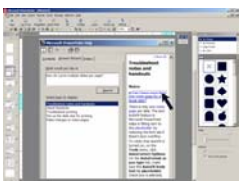
Double experts

Heuristic Evaluations: Types Of Systems To Evaluate

Low fidelity paper prototypes

Medium fidelity prototypes

Completed systems



Other Guidelines: Style Guides

Guidelines published by producers of graphical user interfaces (GUIs)

- Examples:
  - Open Software Foundation MOTIF
  - Open Look
  - MS Windows
  - Apple

Describes the "look and feel" of the GUI

- e.g. Open Look
  - Grouping items in the same menu:
    Use white space between long groups of controls on menus or in short groups when screen real estate is not an issue

Good, but hard too follow

- GUI and widget specific
- Vast number of guidelines
- May miss fundamental design principles

Example Pages From Motif Style Guide, Release 1.1

Message Dialogs

Description

MessageDialogs should be used to convey a message to the user. They must not interrupt the user's interaction with the application. They should include a message, and one of the following button arrangements.

OK
OK Help
OK Cancel
OK Cancel Help
Yes No
Yes No Help
Yes No Cancel
Yes No Cancel Help
Cancel
Cancel Help
Retry Cancel
Retry Cancel Help

Related Information

For more information, see the reference pages for DialogBox, ErrorDialog, InformationDialog, QuestionDialog, WorkingDialog, and WarningDialog

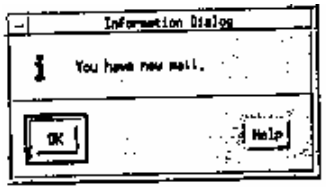
Information Dialog

Description

An InformationDialog should be used to convey information the the user. It must not interrupt the user's interaction with the application. It should include an information symbol, a message, and one of the following button arrangements.

OK
OK Help

Illustration



Related Information

For more information, see the reference page for DialogBox

Other Guidelines: Widget-level "Guides"

Toolkit "hard-wires" guidelines

- Repertoire of widgets
- Look & feel of particular widgets
- Easier to use defaults then to re-invent the wheel!

Some toolkits

- Look & feel is programmer-settable or platform-dependent

Advantages:

- Easy to be consistent
- Widgets developed by experts (e.g., graphical designers)

Disadvantages

- Can be hacked around
- Interfaces "assembled" by non-interface designers can still be terrible

You Now Know

Nine principles of design

- Simple and natural dialog
- Speak the user's language
- Minimize user's memory load
- Be consistent
- Provide feedback
- Provide clearly marked exits
- Provide shortcuts
- Deal with errors in a positive manner
- Provide help

You Now Know (2)

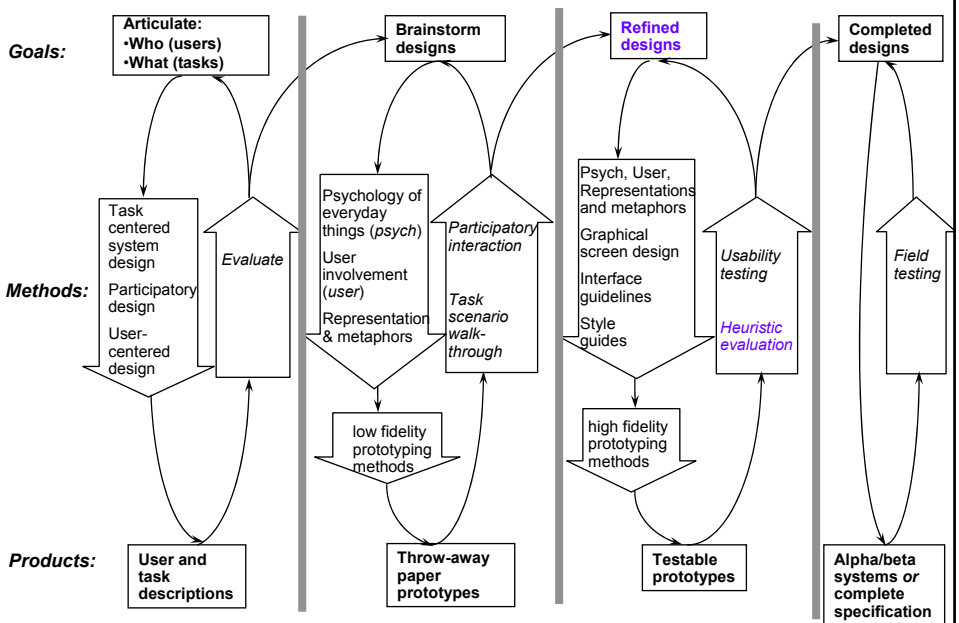
Heuristic evaluation

- Principles can be used to systematically inspect the interface for usability problems

Style guides are mostly platform-dependant design principles

Widget-level guidelines are built into the widgets themselves

Interface Design And Usability Engineering



This diagram is a variation of the one presented by Saul Greenberg