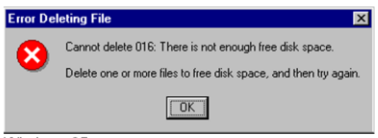
CPSC 217,
Loops In Python: Part 3

In this section of notes you will learn some usability heuristics which can be used to design more user friendly systems.

James Tam

Not So Friendly Examples (If There Is Time)

Do I have any choice in this? [AXE a hex editor]



Windows 95

Uhhh… I give up on this one [Mac shareware version of RISK]

James Tam

Some Rules (Of Thumb) For Designing Software (If There Is Time)

- (The following list comes from Jakob Nielsen's 10 usability heuristics from the book "*Usability Engineering*"
  1. Minimize the user's memory load
  2. Be consistent
  3. Provide feedback
  4. Provide clearly marked exits
  5. Deal with errors in a helpful and positive manner

James Tam

1. Minimize The User's Memory Load (If There Is Time)

- Computers are good at 'remembering' large amounts of information.
- People are not so good remembering things.

James Tam

1. Minimize The User's Memory Load (If There Is Time)

- To reduce the memory load of the user:
  - Describe required the input format, show examples of valid input, provide default inputs
- Examples:

Example 1:

Example 2:

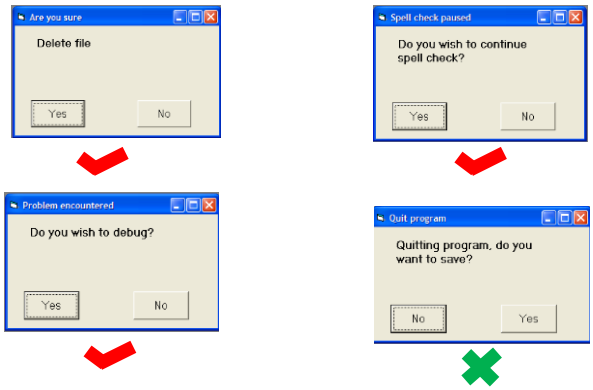
```
[csc loops 25 ]> python hci.py
Enter your birthday <month> <day> <year> e.g., 11 17 1977
Birthday:
```


2. Be Consistent (If There Is Time)

- Consistency of effects
  - Same words, commands, actions will always have the same effect in equivalent situations
  - Makes the system more predictable
  - Reduces memory load
- Consistency of layout
  - Allows experienced users to predict where things should be (matches expectations)

2. Be Consistent (If There Is Time)

- 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).



Images courteously of James Tam

James Tam

2. Be Consistent (If There Is Time)

```
FIRST CATEGORY: ELECTRICITY
---------------------------
You can either enter your monthly kilowatt hours or have an estimate
based on the size of the accomodation that you live in.

(e)stimate
(k)ilowatt hours used
(q)uit this question and proceed to the next question
Enter selection: q

Tons of carbon generated from powering accomodation: 0
Current tons of carbon currently generated: 0


SECOND CATEGORY: HEATING
------------------------

What size of place do you live:
(s)mall house or a flat
(m)edium house
(l)arge house
(q)uit this question and proceed to the next question
Enter selection:
```

This last option allows the user to proceed to the next question.

James Tam

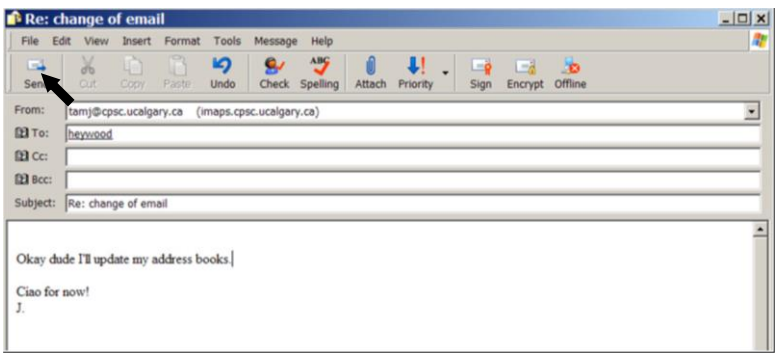
3. Provide Feedback (If There Is Time)

- Letting the user know:
  - What the program is currently doing: was the last command understood, has it finished with its current task, what task is it currently working on, how long will the current task take etc.

James Tam

3. Provide Feedback (If There Is Time)

- What is the program doing?

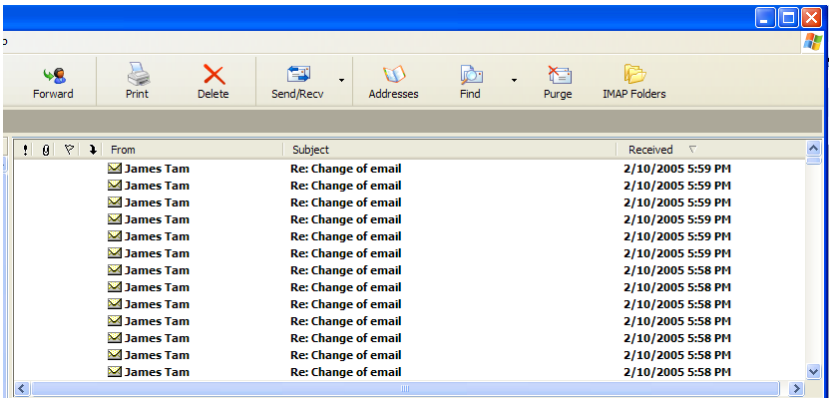


Outlook Express image courteously of James Tam

James Tam

3. Provide Feedback (If There Is Time)

- The rather unfortunate effect on the (poor) recipient.



Outlook Express image courteously of James Tam

James Tam

3. Provide Feedback (If There Is Time)

- In terms of this course, feedback is appropriate for instructions that may not successfully execute
  - what the program is doing (e.g., opening a file),
  - what errors may have occurred (e.g., could not open file),
  - and why (e.g., file "`input.txt`" could not be found)
- ...it's not hard to do and not only provides useful updates with the state of the program ("Is the program almost finished yet?") but also some clues as to how to avoid the error (e.g., make sure that the input file is in the specified directory).
- At this point your program should at least be able to provide some rudimentary feedback
  - E.g., if a negative value is entered for age then the program can remind the user what is a valid value (the valid value should likely be shown to the user as he or she enters the value):

```
age = int(input ("Enter age (0 – 114): "))
```

James Tam

4. Provide Clearly Marked Exits (If There Is Time)

- This should obviously mean that quitting the program should be self-evident (although this is not always the case with all programs!).
- In a more subtle fashion it refers to providing the user the ability to reverse or take back past actions (e.g., the person was just experimenting with the program so it shouldn't be 'locked' into mode that is difficult to exit).
- Users should also be able to terminate lengthy operations as needed.

James Tam

4. Provide Clearly Marked Exits (If There Is Time)

- This doesn't just mean providing an exit from the program but the ability to 'exit' (take back) the current action.
  - Universal Undo/Redo
    - e.g., <Ctrl>-<Z> and <Ctrl>-<Y>
  - Progress indicator & Interrupt
  - Length operations

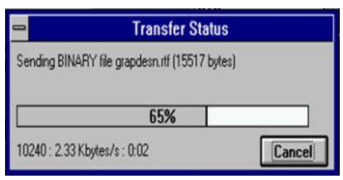
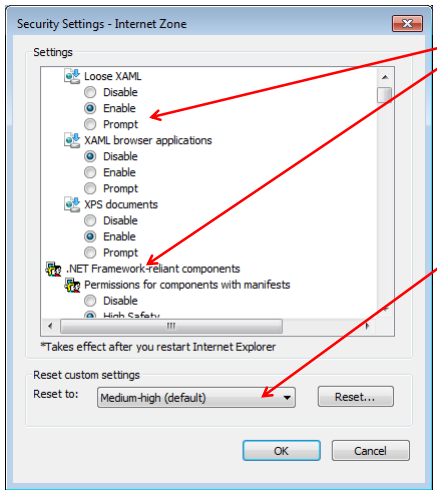


Image: From the "HCI Hall of Shame"

James Tam

4. Provide Clearly Marked Exits (If There Is Time)

- Restoring defaults
  - Getting back original settings



- **What option did I change?**
- **What was the original setting?**
- **Allows for defaults to be quickly restored**

Image: Internet Explorer security settings curtesy of James Tam

James Tam

4. Provide Clearly Marked Exits (If There Is Time)

The user can skip or 'exit' any question

```
FIRST CATEGORY: ELECTRICITY
---------------------------
You can either enter your monthly kilowatt hours or have an estimate
based on the size of the accomodation that you live in.

(e)stimate
(k)ilowatt hours used
(q)uit this question and proceed to the next question
Enter selection: q

Tons of carbon generated from powering accomodation: 0
Current tons of carbon currently generated: 0


SECOND CATEGORY: HEATING
------------------------

What size of place do you live:
(s)mall house or a flat
(m)edium house
(l)arge house
(q)uit this question and proceed to the next question
Enter selection:
```

Image: An old CPSC 231 assignment curtesy of James Tam

James Tam

5. Deal With Errors In A Helpful And Positive Manner (If There Is Time)

- (JT: with this the heuristic it states exactly what should be done).

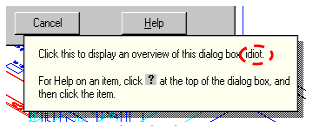
James Tam

Rules Of Thumb For Error Messages (If There Is Time)

1. Polite and non-intimidating
   - Don't make people feel stupid
     - `Try again, bonehead!` ← No

2. Understandable
   - Error 25 ← Not

So obvious it could never happen?



AutoCAD Mechanical

3. Specific
   - Cannot open this document ← Why?
   - Cannot open "chapter 5" because the application "Microsoft Word" is not on your system ← Better

4. Helpful
   - Cannot open "chapter 5" because the application "Microsoft Word" is not on your system. Open it with "WordPad" instead?

Even better: A potentially helpful suggestion

James Tam

(If There Is Time)

"HIT ANY KEY TO CONTINUE"

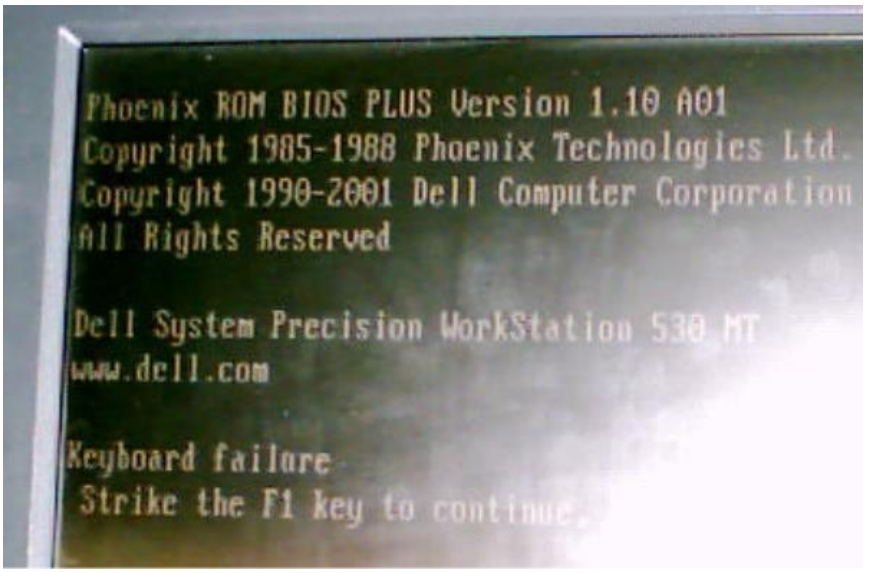
James Tam

(If There Is Time)

Image: Curtesy of James Tam

James Tam

I'd Rather Deal With The 'Any' Key (If There Is Time)



Picture courtesy of James Tam: An error message from a Dell desktop computer

James Tam

After This Section You Should Now Know

- Rules of thumb for interaction design (if there is time)
  1. Minimize the user's memory load
  2. Be consistent
  3. Provide feedback
  4. Provide clearly marked exits
  5. Deal with errors in a helpful and positive manner

James Tam

Copyright Notification

- "Unless otherwise indicated, all images in this presentation are used with permission from Microsoft."