DESIGN USE BUILD

PAUL G. ALLEN SCHOOL OF COMPUTER SCIENCE & ENGINEERING

CSE 440: Introduction to HCI

User Interface Design, Prototyping, and Evaluation!

Lecture 14: Heuristic Evaluation

Instructor: Amy Zhang, 11/18/2021

Today’s Topics

- Heuristic Evaluation
  - Nielsen’s 10 Heuristics
  - Heuristic evaluation process
- Paired team usability test of paper prototypes (15–20 min per team, then swap)
- Team work time on combining and cleaning up your heuristic evaluations for 3b (due 8PM)

Heuristic Evaluation

Reminder from last lecture:

- Heuristic Evaluation helps find **usability problems** in a design
- It’s a systematic UI inspection led by **experts**, such as designers on your team
- Method:
  - A small set of 3–5 evaluators examine the interface
  - They independently check compliance with a set of **design principles** (e.g., how easy is it to prevent errors?).
  - Different evaluators find different problems
  - Evaluators communicate at the end
- Can do this with working interfaces or sketches
- Developed by Jakob Nielsen, though several lists of design principles exist

Nielsen's 10 Heuristics

1. Visibility of system status
2. Match between system and the real world
3. User control and freedom
4. Consistency and standards
5. Error prevention
6. Recognition rather than recall
7. Flexibility and efficiency of use
8. Aesthetic and minimalist design
9. Help recognize, diagnose, and recover from errors
10. Help and documentation

Nielsen's 10 Heuristics

1. **Visibility of system status** Visibility and Exposing State
2. **Match between system and the real world** Metaphors, mapping
3. User control and freedom
4. **Consistency and standards** Internal and external consistency
5. Error prevention
6. **Recognition rather than recall** Knowledge in the world vs in the head
7. Flexibility and efficiency of use
8. Aesthetic and minimalist design
9. Help recognize, diagnose, and recover from errors
10. **Help and documentation**

Learnability!

Nielson's 10 Heuristics

1. Visibility of system status
2. Match between system and the real world
3. **User control and freedom** Undo
4. Consistency and standards
5. **Error prevention** Confirmation dialogs
6. Recognition rather than recall
7. Flexibility and efficiency of use
8. Aesthetic and minimalist design
9. **Help recognize, diagnose, and recover from errors** Error messages
10. Help and documentation

Safety!

Nielsen's 10 Heuristics

1. Visibility of system status
2. Match between system and the real world
3. User control and freedom
4. Consistency and standards
5. Error prevention
6. Recognition rather than recall
7. **Flexibility and efficiency of use** Multiple interaction styles
8. Aesthetic and minimalist design
9. Help recognize, diagnose, and recover from errors
10. Help and documentation

Efficiency!

Other sets of design principles/heuristics

- Bruce Tognazzini's 16 First Principles
- Ben Shneiderman's 8 Golden Rules
- Susan Weinschenk and Dean Barker's meta-classification of 20 heuristics
- Don Norman's 6 Design Principles (remember his gulfs of evaluation/execution)
    1. **Visibility**: make UI options visible
    2. **Feedback**: user actions need UI reaction
    3. **Affordance**: the way something looks should indicate how it's meant to be used
    4. **Mapping**: UI controls to something will resemble what they affect
    5. **Constraints**: limits to an interaction or UI are clear
    6. **Consistency**: same action causes same reaction

Nielsen’s 10 Heuristics

Heuristic #1: Visibility of system status

- **Visibility of system status**
  - The system should always keep users informed about what is going on, through appropriate feedback within reasonable time
  - Visibility helps to align a user’s mental model with interface+system model
- **Use of feedback**
  - The UI should give feedback for a user action so that they know if the system state has been updated

Heuristic #1: Visibility of system status

UI feedback should happen in an appropriate amount of time

- 0.1 sec: no special indicators needed
- 1.0 sec: user tends to lose track of data
- 10 sec: maximum duration if user to stay focused on action
  longer delays absolutely require percent-done progress bars

Heuristic #2: Real World Match

- **Match between system and the real world**
- The system should speak the users' language, with words, phrases and concepts familiar to the user, rather than system-oriented terms.
- Follow real-world conventions, making information appear in a natural and logical order. Refers to word and language choice, mental model, metaphor, mapping, and sequencing.

Heuristic #2: Real World Match

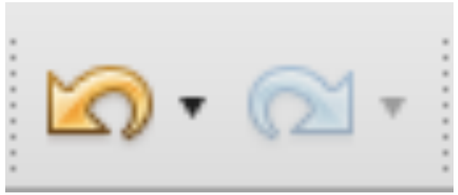
“mailto”, “protocol”?

Speak the user’s language

“mailto is not a registered protocol”
—>
“Your browser doesn’t have an email app connected”

Heuristic #3: User in Control

- **User control and freedom**
- Users often choose system functions by mistake and will need a clearly marked “emergency exit” to leave the unwanted state without having to go through an extended dialogue.
- Support undo and redo. Not just for navigation exits, but for getting out of any situation or state.

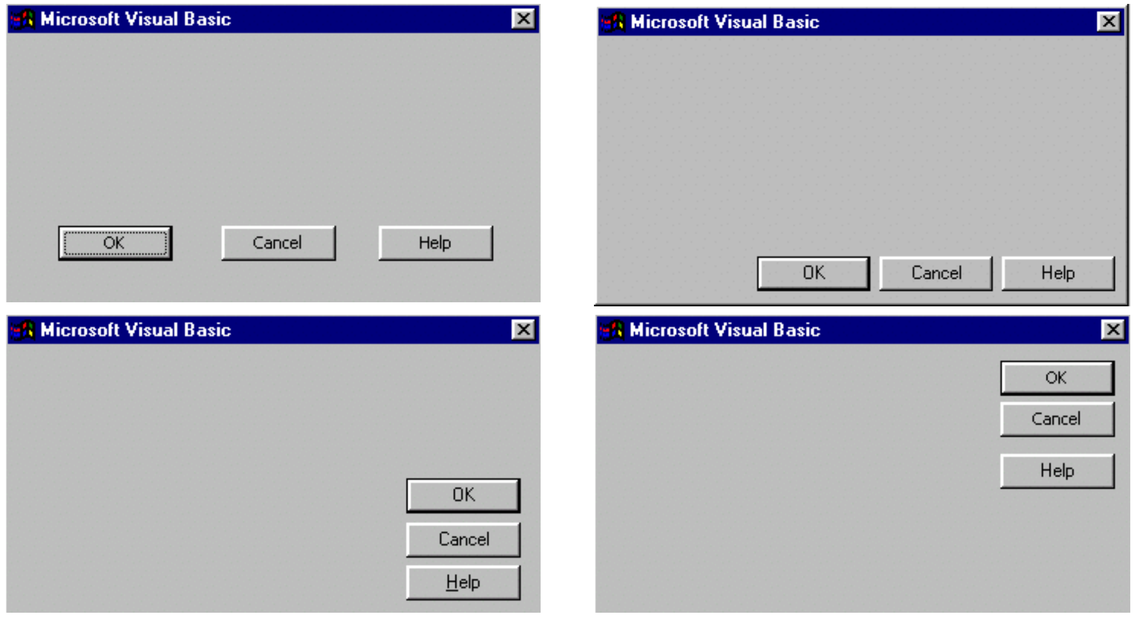


Heuristic #4: Consistency

- **Consistency and standards**
- Users should not have to wonder whether different words, situations, or actions mean the same thing. Follow platform conventions.
- Internal consistency is consistency throughout the same product.
- External consistency is consistency with other products in its class.

Heuristic #4: Consistency

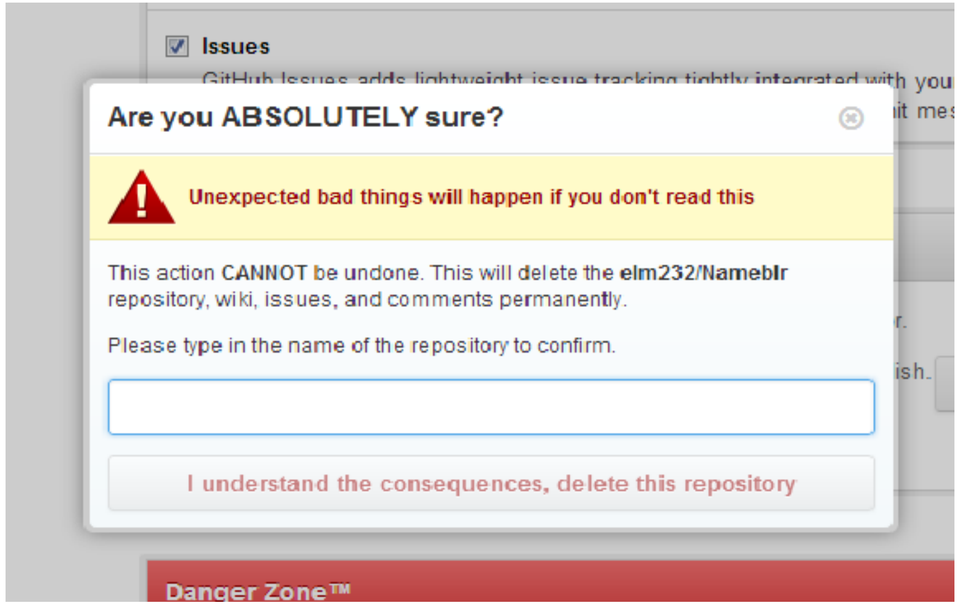
- Consistency and **standards**



Heuristic #5: Error Prevention

- Even better than good error messages is a careful design which prevents a problem from occurring in the first place.
- Either **eliminate error-prone conditions** or check for them and present users with a **confirmation option** before they commit to the action.

Heuristic #5: Error Prevention



Heuristic #6: Recognition over Recall

- **Minimize the user's memory load** by making objects, actions, and options visible.
- The user should not have to remember information from one part of the dialogue to another.
- Instructions for use of the system should be visible or easily retrievable whenever appropriate.

Heuristic #6: Recognition over Recall

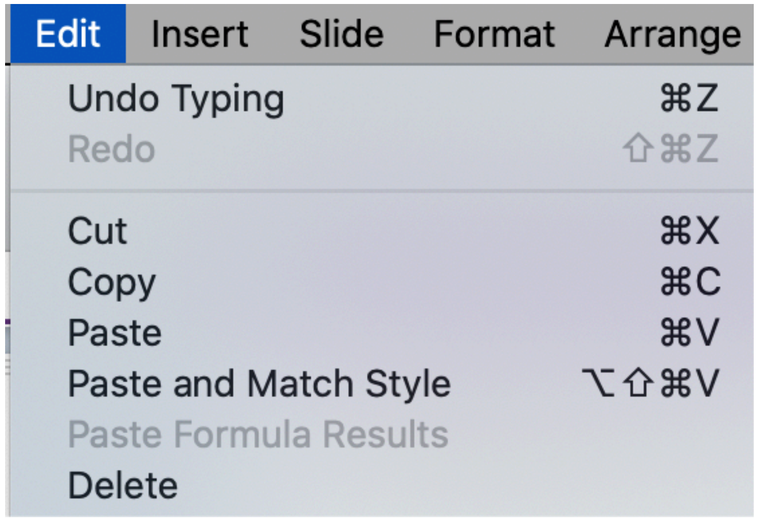
- Hidden and false affordances (ex: Norman doors) violate this rule (minimize the user's memory load).
  - hidden: need to remember where to do something
  - false: need to remember the right way to do something

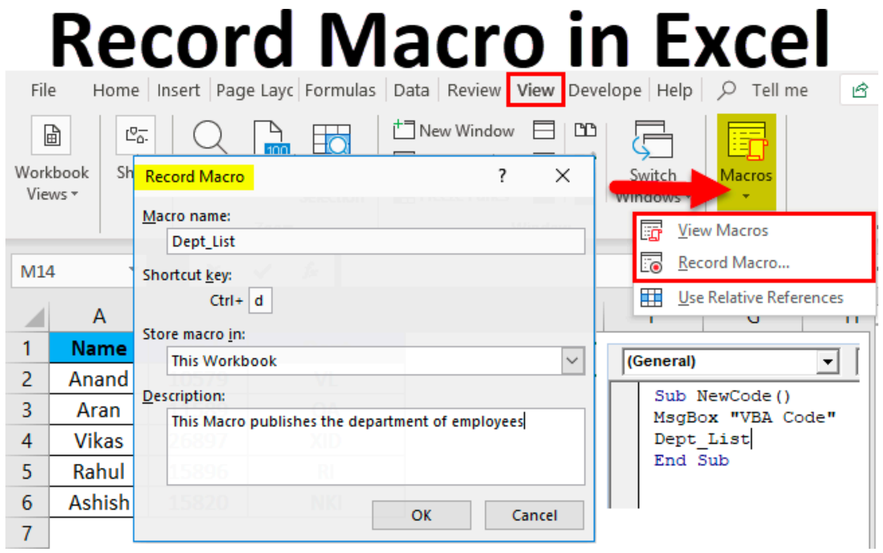
Heuristic #7: Flexibility and Efficiency

- **Avoid repetitive actions** that must be done manually.
- **Allow multiple ways** to do things.

Heuristic #7: Flexibility and Efficiency

- UI should cater to both inexperienced and experienced users (accelerators for experts).
- Allow users to automate frequent actions.



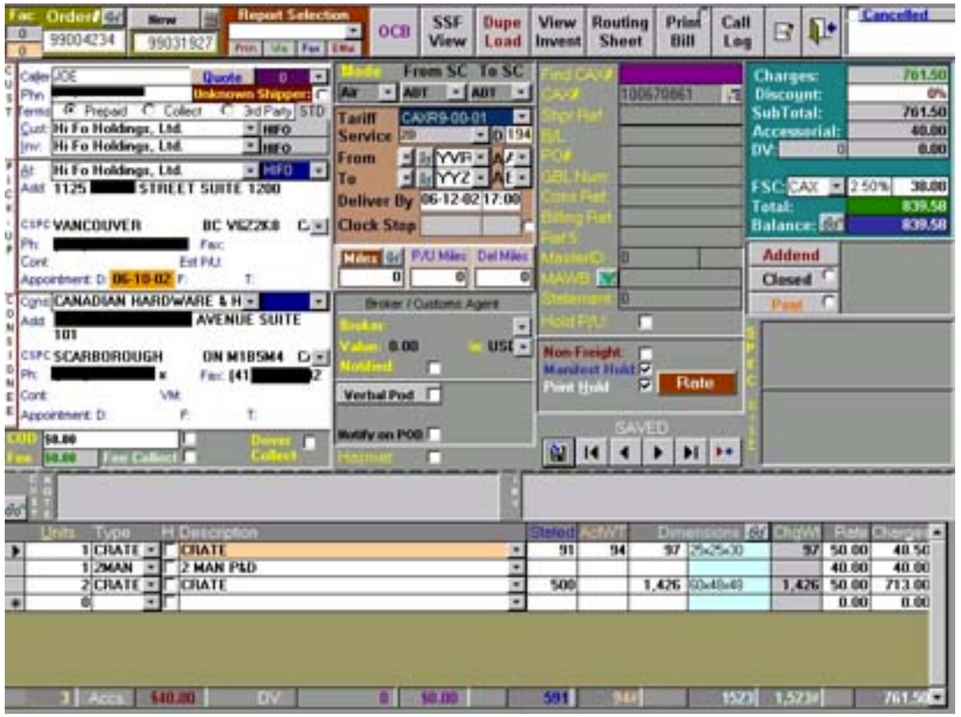


Heuristic #8: Aesthetic and Minimalist Design

- Dialogues should not contain information which is **irrelevant or rarely needed**.
- Every extra unit of information competes with the relevant units of information and diminishes their relative visibility.

Heuristic #8: Aesthetic and Minimalist Design

- Not just about “ugliness”.
- This is about clutter, overload of visual field, visual noise, distracting animations, and so on.

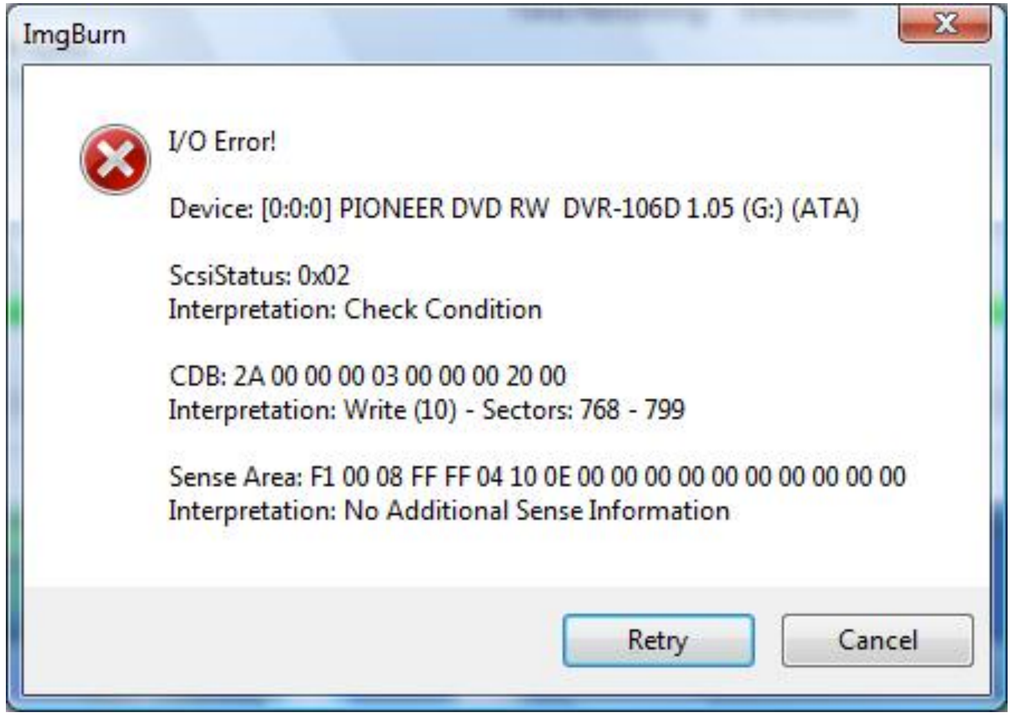


Heuristic #9: Error Recovery

- **Help users recognize, diagnose, and recover from errors**
- Error messages should be expressed in plain language (no codes), precisely indicate the problem, and constructively suggest a solution.

Heuristic #9: Error Recovery

- plain language
- precisely indicate problem
- constructively suggest solution



Heuristic #10: Help and Documentation

- Even though it is better if the system can be used without documentation, **it may be necessary to provide help and documentation**.
- Any such information should be easy to search, focused on the user's task, list concrete steps to be carried out, and not be too large.

Heuristic Evaluation Process

Heuristic Evaluation Process

1) **Usability test setup**

   The facilitator informs evaluators of the context surrounding the scenario and the task.

2) **Evaluators evaluate interface & make lists of problems**

   The evaluators should take at least two passes through the interface—first to get a feeling for the task flow of the task, then a second time to focus on evaluation and design inspection (in the 2nd time, you can explore more/ask more questions). As the evaluators, you should be individually taking notes and focus on generating as many problems as you can, don't rank severity yet.

3) **Severity rating**

   After the study, evaluators get together and combine all the noticed problems together. *Individually* determine how severe each problem is (from 0–4).

4) **Aggregation**

   As a group again, discuss the problems and come to consensus on severity ratings.

5) **Debriefing**

   Discuss the outcome with design team.

Severity Rating

Used to allocate resources to fix problems

Combination of:

frequency - how common?

impact - how hard to overcome?

persistence - how often to overcome?

Should only be calculated after all problems by all evaluators have been identified

Should be done independently by all evaluators, then discussed as a group and aggregated

Severity Rating

0 - Do not agree this is a problem.

1 - **Usability blemish**. Mild annoyance or cosmetic problem. Easily avoidable.

2 - **Minor usability problem**. Annoying, misleading, unclear, confusing. Can be avoided or easily learned. May occur only once.

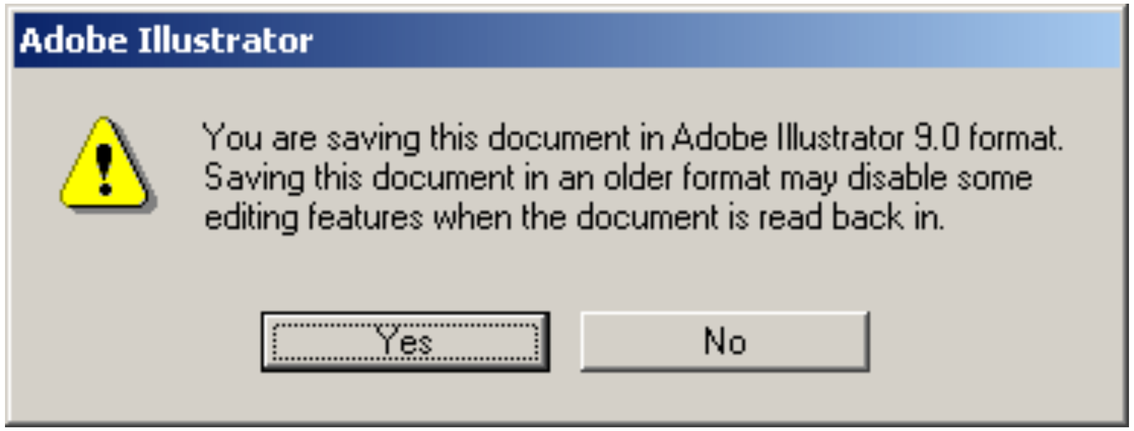
3 - **Major usability problem**. Prevents users from completing tasks. Highly confusing or unclear. Difficult to avoid. Likely to occur more than once.

4 - **Critical usability problem**. Users will not be able to accomplish their goals. Users may quit using system all together.

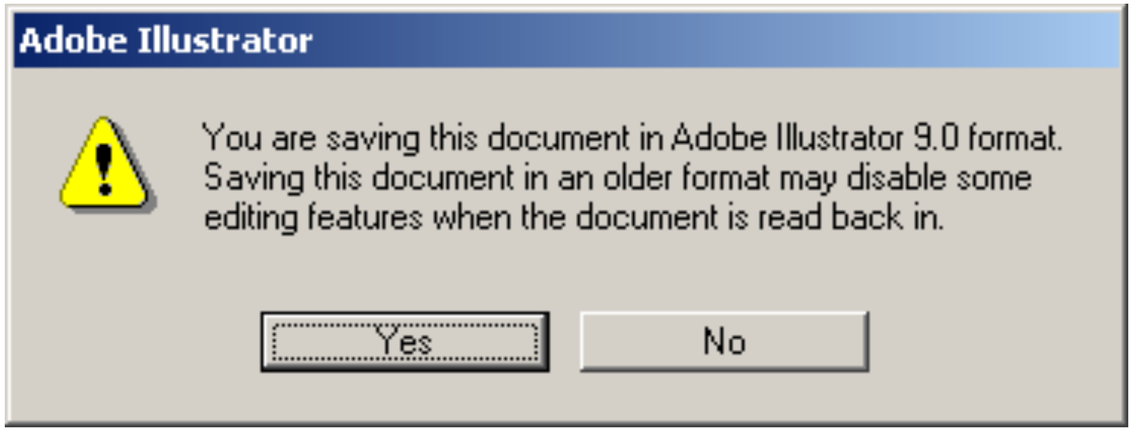
Writing Good Heuristic Evaluations

- Remember our tips regarding giving feedback!
- Be **tactful**
  - Not: “the menu organization is a complete mess”
  - Better: “menus are not organized by function”
- Be **specific**
  - Not: “text is unreadable”
  - Better: “text is too small, and has poor contrast (black text on dark green background)”

Example: Heuristic Evaluation



Example



- **Problem**: It is unclear what happens when the user presses “Yes” or “No” since the dialog is not asking a question but instead confirming an action.
- **Heuristics**: #4 Consistency and Standards, #5 Error Prevention
- **Severity**: 2 - minor
- **Recommendation** (optional): Replace “Yes” button with “OK” and the “No” button with “Cancel”.

- Find another team that you did not pair up with on Tuesday and that is not in your section.
- [20 min x2] First, start out with a usability test. The presenting team picks roles (computer, facilitator, observer [switch up from last time!]) and then the facilitator reads aloud their task script. Note critical incidents.
  - Evaluating team just tries to performs the task in their 1st pass. In the 2nd pass, they are exploring the interface more and asking more questions (presenting team can now answer Q's in the 2nd pass). The whole time, each evaluator is taking individual notes about problems they notice or encounter. Here's a worksheet for note-taking: https://tinyurl.com/f9hk7me4
  - Then swap!
- [15 min] Once both teams are done, get back into your team to aggregate all the problems. Then ***individually*** rate the severity. Then come together again and discuss and converge on severity ratings.
  - Final heuristic evaluation of another team's prototype due at 8PM.
- If you have remaining time left over, try another team's UI in a usability test!

Other announcements

- Friday's section (tomorrow) is now going to be an exam prep session. We'll also share the results of your heuristic evaluation.
- Yes, we do have class next Tuesday.
- 3c (usability tests) is out and is due Nov 29th (but I encourage you to finish before Thanksgiving holiday).