Heuristic Evaluation

Prof. James A. Landay
Computer Science Department
Stanford University
Autumn 2014
October 30, 2014

Hall of Fame or Shame?

Pocket

By Read It Later

Hall of Fame or Shame?

Items can be saved from iPhone, iPad, Android devices, web app, Chrome/Firefox extension, or email.

Hall of Fame or Shame?

Truth, Lies, and ‘Doxing’: The Real Moral of the Gawker/Reddit Story

Hall of Fame or Shame?

forecast.io
courtesy of William D.
Visual Design Review

- Start with Context, what is the nature of the information? What is the most important?
- Avoid clutter, focus on the essence of your tasks – CUT!
- Design first in?  
  - grayscale to focus on hierarchy
- Small changes help us see  
  - key differences (e.g., small multiples)
- Use color properly – not for  
  - ordering!
- Only use one or two colors at a time, unless absolutely necessary

Evaluation

- About figuring out how to improve design
- Issues with lo-fi tests?

Not realistic  
- visuals & performance

Not on actual interface  
- can’t test alone

Need participants
Heuristic Evaluation

- Developed by Jakob Nielsen
- Helps find usability problems in a UI design
- Small set (3-5) of evaluators examine UI
  - independently check for compliance with usability principles (“heuristics”)
  - 3-5 evaluators because?
    • different evaluators will find different problems
    • evaluators only communicate afterwards
    • findings are then aggregated
  - Can perform on working UI or on sketches

Why Multiple Evaluators?

- Every evaluator doesn’t find every problem
- Good evaluators find both easy & hard ones

Heuristic Evaluation Process

- Evaluators go through UI several times
  - inspect various dialogue elements
  - compare with list of usability principles
  - consider other principles/results that come to mind
- Usability principles
  - Nielsen’s “heuristics”
  - supplementary list of category-specific heuristics
    • competitive analysis & user testing of existing products
- Use violations to redesign/fix problems

Heuristics (original)

- H1-1: Simple & natural dialog
- H1-2: Speak the users’ language
- H1-3: Minimize users’ memory load
- H1-4: Consistency
- H1-5: Feedback
- H1-6: Clearly marked exits
- H1-7: Shortcuts
- H1-8: Precise & constructive error messages
- H1-9: Prevent errors
- H1-10: Help and documentation

Heuristics (revised set)

- H2-1: Visibility of system status
  - keep users informed about what is going on
  - example: pay attention to response time
    • 0.1 sec: no special indicators needed, why?
    • 1.0 sec: user tends to lose track of data
    • 10 sec: max. duration if user still focused on action
      for longer delays, use percent-done progress bars
- H2-2: Match between system & real world
  - follow real world conventions
  - speak the users’ language

Heuristics (cont.)

- Bad example: Mac desktop
  - Dragging disk to trash
    • should delete it, not eject it
- H2-2: Match between system & real world
  - follow real world conventions
  - speak the users’ language
Heuristics (cont.)

- H2-3: User control & freedom
  - “exits” for mistaken choices, undo, redo
  - don’t force down fixed paths
    - like that BART ticket machine...

- Wizards
  - must respond to Q before going to next
  - for infrequent tasks
    - e.g., router config.
  - not for common tasks
    - good for beginners
  - have 2 versions (WinZip)

Heuristics (cont.)

- H2-4: Consistency & standards
  - use the same language, placement, etc. everywhere
  - follow platform conventions

Heuristics (cont.)

- H2-5: Error prevention
- H2-6: Recognition rather than recall
  - make objects, actions, options, & directions visible/easily retrievable

Heuristics (cont.)

- H2-7: Flexibility and efficiency of use
  - accelerators for experts (e.g., gestures, kb shortcuts)
  - allow users to tailor frequent actions (e.g., macros)

Heuristics (cont.)

- H2-8: Aesthetic & minimalist design
  - no irrelevant information in dialogues
Heuristics (cont.)

- H2-9: Help users recognize, diagnose, & recover from errors
  - error messages in plain language
  - precisely indicate the problem
  - constructively suggest a solution

Good Error Messages

- Clearly indicate what has gone wrong
- Human readable
- Polite
- Describe the problem
- Explain how to fix it
- Highly noticeable

Heuristics (cont.)

- H2-10: Help and documentation
  - easy to search
  - focused on the user’s task
  - list concrete steps to carry out
  - not too large

Mobile Heuristics

<table>
<thead>
<tr>
<th>Mobile Heuristic</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Heuristic 1</td>
<td>Visibility of system status and capability/ availability of the mobile device</td>
</tr>
<tr>
<td>Heuristic 2</td>
<td>Match between system and the real world</td>
</tr>
<tr>
<td>Heuristic 3</td>
<td>Consistency and completeness</td>
</tr>
<tr>
<td>Heuristic 4</td>
<td>Good error messages and remedial design</td>
</tr>
<tr>
<td>Heuristic 5</td>
<td>Ease of learning, error, and testability</td>
</tr>
<tr>
<td>Heuristic 6</td>
<td>Flexibility, manageability, and extensibility</td>
</tr>
<tr>
<td>Heuristic 7</td>
<td>Aesthetic parameters and social conventions</td>
</tr>
<tr>
<td>Heuristic 8</td>
<td>Usability, error, and interactivity</td>
</tr>
</tbody>
</table>

Phases of Heuristic Evaluation

1) Pre-evaluation training
   - give evaluators needed domain knowledge & information on the scenario
2) Evaluation
   - individuals evaluates UI & makes list of problems
3) Severity rating
   - determine how severe each problem is
4) Aggregation
   - group meets & aggregates problems (w/ ratings)
5) Debriefing
   - discuss the outcome with design team

How to Perform Evaluation

- At least two passes for each evaluator
  - first to get feel for flow and scope of system
  - second to focus on specific elements
- If system is walk-up-and-use or evaluators are domain experts, no assistance needed
  - otherwise might supply evaluators with scenarios
- Each evaluator produces list of problems
  - explain why with reference to heuristic or other information
  - be specific & list each problem separately

Examples

- Can’t copy info from one window to another
  - Violates “Minimize the users’ memory load” (H1-3)
  - fix: allow copying
- Typography uses different fonts in 3 dialog boxes
  - Violates “Consistency and standards” (H2-4)
  - slows users down
  - probably wouldn’t be found by user testing
  - fix: pick a single format for entire interface
How to Perform Heuristic Evaluation

- Why separate listings for each violation?
  - risk of repeating problematic aspect
  - may not be possible to fix all problems

- Where problems may be found
  - single location in UI
  - two or more locations that need to be compared
  - problem with overall structure of UI
  - something that is missing
    - common problem with paper prototypes
      - note: sometimes features are implied by design docs and just haven’t been “implemented” – relax on those

Severity Rating

- Used to allocate resources to fix problems
- Estimates of need for more usability efforts
- Combination of
  - frequency
  - impact
  - persistence (one time or repeating)
- Should be calculated after all evals. are in
- Should be done independently by all judges

Severity Ratings (cont.)

0 – don’t agree that this is a usability problem
1 - cosmetic problem
2 - minor usability problem
3 - major usability problem; important to fix
4 - usability catastrophe; imperative to fix

Debriefing

- Conduct with evaluators, observers, and development team members
- Discuss general characteristics of UI
- Suggest potential improvements to address major usability problems
- Dev. team rates how hard things are to fix
- Make it a brainstorming session
  - little criticism until end of session

Severity Ratings Example

1. [H1-4 Consistency] [Severity 3][Fix 0]

The interface used the string “Save” on the first screen for saving the user’s file, but used the string “Write file” on the second screen. Users may be confused by this different terminology for the same function.

HE vs. User Testing

- HE is much faster
  - 1-2 hours each evaluator vs. days-weeks
- HE doesn’t require interpreting user’s actions
- User testing is far more accurate (by def.)
  - takes into account actual users and tasks
  - HE may miss problems & find “false positives”
- Good to alternate between HE & user testing
  - find different problems
  - don’t waste participants
Results of Using HE

- Discount: benefit-cost ratio of 48 [Nielsen94]
  - cost was $10,500 for benefit of $500,000
  - value of each problem ~15K (Nielsen & Landauer)
  - how might we calculate this value?
    - in-house → productivity; open market → sales
- Correlation between severity & finding w/ HE
- Single evaluator achieves poor results
  - only finds 35% of usability problems
  - 5 evaluators find ~75% of usability problems
  - why not more evaluators???? 10? 20?
    - adding evaluators costs more & won’t find more probs

Decreasing Returns

Summary

- Have evaluators go through the UI twice
- Ask them to see if it complies with heuristics
  - note where it doesn’t & say why
- Combine the findings from 3 to 5 evaluators
- Have evaluators independently rate severity
- Alternate with user testing

Further Reading

- Books
  - Usability Engineering, by Nielsen, 1994
- Web Sites & mailing lists
  - http://www.nngroup.com/articles/

Next Time

- User Testing
  - No Reading
- Individual assignment
  - Simple Heuristic Evaluation (individual)
  - Due Tue, Nov. 4 at 11:59 PM (via CourseWork)
- Looking ahead
  - week off from major team project work
  - you will be doing Heuristic Evaluation of other team’s
  - next week
  - next week get basic web sites together (for HE)