Interaction Techniques

SWE 432, Fall 2016
Design and Implementation of Software for the Web
Today

• What principles guide the design of usable interaction techniques?
  • How can interaction designs help support making plans, taking action, and interpreting feedback?
  • How does a direct manipulation interface make complex tasks easier?
Interaction technique

• A method by which a user can perform an action or sequence of actions with a computer.
• Might encompass software (e.g., accelerators on a menu) and/or specialized hardware (momentum scrolling on iOS)
• What makes a good interaction technique?
  • Usability: task performance, discoverability, learnability, …
Example: Filtering

- http://www.kayak.com
Gulfs of execution and evaluation
Norman’s 7 stages of action

1. Goal (form the goal)
2. Plan (the action)
3. Specify (action sequence)
4. Perform (action sequence)
5. Perceive (the state of the world)
6. Interpret (the perception)
7. Compare (outcome w/ goal)
Translation

goals $\rightarrow$ action sequence
Signifiers

- a.k.a “cognitive affordances” [Hartson & Pyla]

Goals

- Show which UI elements can be manipulated
- Show how they can be manipulated
- Help users get started
- Guide data entry
- Suggest default choices
- Support error recovery
Hinting

• Indicate which UI elements can be interacted with
• Possible visual indicators
  • Static hinting - distinctive look & feel
  • Dynamic hinting - rollover highlights
  • Response hinting - change visual design with click
  • Cursor hinting - change cursor display
Help users predict outcome of actions

- What does this do?
- Should I click it?
Clarity of wording (Example)

- Design for clarity & precision
Clarity of wording

- Choose words carefully
- Speak the user’s language
- Avoid vague, ambiguous terms
- Be as specific as possible
- Clearly represent domain concepts
Consistency

• In use of **terms**
  • e.g., do not use "revise" and "edit" interchangeably

• In how commands **map** to UI interactions
Likely & useful defaults

- Default text, if relevant (e.g., date)
- Default cursor position
- Avoid requirements to retype & re-enter data
Avoid using modes

- Modes create inconsistent mapping
  - E.g., control S sometimes saves, sometimes sends email
  - Especially dangerous for frequent interactions that become highly automatic System 1 actions
- Avoid when possible
- Clearly distinguish if necessary
Physical actions
Provide intermediate feedback during interactions

• As user is interacting with objects, provide feedback on interactions

• Examples
  • While dragging object, show new position
  • As selecting text, show selection
  • While clicking on button, show button changing
Avoid physical awkwardness

• Switching between input devices takes time
• Avoid forcing user to constantly switch between input devices (e.g., keyboard & mouse)
  • e.g., Effective tab order between fields
• Avoid awkward keyboard combinations
Fitt’s law

- Time required to move to a target **decreases** with target **size** & **increases** with **distance** to the target.
- Movements typical consist of:
  - one large quick movement to target (**ballistic movement**)
  - fine-adjustment movement (**homing** movements)
- Homing movements generally responsible for most of movement time & errors.
- Applies to rapid pointing movements, not slow continuous movements.
Design implications of Fitt’s law

• **Constraining** movement to one dimension dramatically increases speed of actions
  • e.g., scroll bars are 1D
Design implications of Fitt’s law

- Making controls **larger** reduces time to invoke actions
- Locating controls closer to user **cursor** reduces time
  - e.g., context menus
Design implications of Fitt’s law

- Positioning button or control along **edge** of screen acts as barrier to movement, substantially reducing homing time & errors
System feedback
System response times

- 0.1 second - reacting **instantaneously**
  - requiring no special feedback except displaying result
  - limit for direct manipulation of objects in UI
- 1.0 second - **freely** navigating commands
  - noticeable delay, limit for keeping user’s flow of thought uninterrupted
- 10 seconds - keeping users **attention**
  - limit for keeping user’s attention focus in UI
  - longer delays create task breaks
- [Nielsen, Usability Engineering, 1993]
Automation

• Keep user in control at highest task levels
• Take control from user when need is obvious & user is busy
• Provide visibility of automation & opportunities to correct when necessary
Provide feedback for all user actions

- Feedback helps keep users on track in accomplishing goals
- Request confirmation to prevent costly errors (but use sparingly)
- Make feedback visible, noticeable, legible, located w/ in users focus of attention
- Provide feedback early
- Provide feedback consistently
Tone of feedback

- Establishes relationship with user
- Important not to take user feel “stupid”
- Make the system take blame for errors
- Be positive, to encourage
- Provide helpful messages, not cute messages
- Avoid violent, negative, demeaning, threatening terms (e.g., illegal, invalid)
Crafting feedback text

- Clarity - support clear understanding of outcome
- Precise wording
- Completeness - include enough information to fully understand outcomes
Show users how to fix errors

• Good: detecting user errors
• Better: directly showing how errors can be fixed
• (Best: using constraints to prevent errors from ever occurring)
Avoid anthropomorphism (in most contexts)

• Anthropomorphism - the attribution of human characteristics to non-human objects
  • e.g., “Sorry, I but I cannot find the file you need”
• Provides a false mental model
  • leads to user thinking they can interact with system as person
  • can be over promising & condescending
• May work in spoken interaction settings, where system does match user’s mental model
In Class Activity

• In groups of 2 or 3:
  • Identify at least 3 separate usability issues of a web application that violates one of the interaction design principles in this lecture
  • For each issue, brainstorm ways that this usability issue might be addressed.
Direct manipulation
Motivation

• User is trying to do a task, manipulating some [model] of world
• Hard to plan out long sequence of actions in advance
• Gulf of execution: hard to know if took correct action
• Gulf of evaluation: hard to understand if successfully manipulated world
• Hard to compare hidden world to desired world
Direct manipulation

- “Rapid incremental reversible operations whose impact on the objects of interest is immediately visible” (Shneiderman, 1982)
Benefits

- Supports exploration
  - Don’t plan long sequence of actions: pick an action, try it, can change mind if want to do something else instead
- Provides immediate feedback
  - Can quickly see what outcome of actions are in manipulating the world
  - Easy to compare desired state of the world to actual state of the world
Example - GUI builder
Example - Spreadsheets
Example - Microsoft TerraServer
Example - Google Maps
Example - Kayak

DCA ↔ CHI
10G of 1,115 flights
Dec 16 → Dec 19
Friday → Monday
Economy 1 traveler

Sort by: Price Recommended Duration More
Round-trip | Flight-by-flight

$207
JustFly, Experience world-class service
Click “View Deal” to find our cheapest flights
www.justfly.com

$227
American Airlines
8:12p DCA → 9:26p ORD 2h 14m nonstop
3:25p ORD → 6:12p DCA 1h 47m nonstop

$227
American Airlines
8:12p DCA → 9:26p ORD 2h 14m nonstop
11:55a ORD → 2:42p DCA 1h 47m nonstop

LaToza/Bell
GMU SWE 432 Fall 2016
In Class Activity: Direct Manipulation Programming Interactions

• In groups of 2
  • Design a system for writing React code through direct manipulation
    • Create sketches showing key screens
  • Should support
    • Standard programming language features (variables, conditionals, loops, functions)
    • Should make it faster and easier to make code changes
    • Should make it easier to get feedback on if program works