Designing with the Mind in Mind

Chapters 13, 14, 15 and Appendix

Reading Summary

Gurpreet Singh
February 10, 2018
CS4474

Chapter 13

Our Hand-Eye Coordination Follows Laws

Fitts Law: Pointing at Displayed Targets

The larger your target on the screen and the closer it is to your starting point, the faster you can reach it. The formula for this is T = a + b log(1+ D/W) where T is the time required to point to something, D is the distance, W is the width of the target. Applies to all pointing devices and all people. a and b are variables accounting for differences between people and pointing devices.

Design implications

- Use decently sized click targets
- Make the click-boxes big enough to not frustrate the user
- Accept clicks on labels and graphics to improve hit rate
- Use margins around click targets to avoid misclicks
- Placing targets near the edge of the screen makes them easier to click

Steering Law: Moving Pointers Along Constrained Paths

Making users move a pointer along a narrow path is slower than allowing them a larger path to move the pointer. Design implications include making drop down menu entries wider and higher, avoiding pull-right menus and page rulers.

This law mainly describes the absolute pain it is to use pull-right menus. That feeling when you are 4 clicks deep into a pull-right and you go off UI and have to restart.

Scroll bars and page rulers are also another area where this problem is present.

Chapter 14

We have Time Requirements

Responsivness defined

Responsiveness is how quickly a system responds to the user's interactions. A system can be responsive even with poor performance. You should use callbacks to let the UI continue operating and let the user know that something will finish. When a system isnt responsive it cant meet the time deadlines of the human.

Time constraints for the human brain

1 milisecond is the shortest amount of silence you can detect. 5 miliseconds is the shortest amount of time you can see a visual change to be effected by it 80 miliseconds is how long it takes you to flinch to something. Then it takes 100 miliseconds to fully process something you have seen. Your system has to react to an action within 140 miliseconds (1.4 seconds) for the user to understand that their action had an effect on your system. Your attention takes 500 miliseconds to reset from one item to the next. It takes 700 miliseconds to do a motor action after observing something visual. Any gap in conversation longer than 1 second is awkward. A sub-mental task can take up to a maximum of 10 seconds.

Interactive systems need to keep the above constraints in mind when engineering.

HCI implications

The guidelines for this chapter are:

- React to user interactions instantly
- Indicate background processing
- Use callbacks and dont hold UI in focus when processing
- Animate smoothly
- Allow users to cancel processing
- provide a ETA for processing

Additional implications include - use busy indicators (spinning circles and loading bars) - Know when it is allowed to use a delay and when it will bother the user (sometimes its acceptable if they are doing something hard) - Display important data first and avoid delays by prompting for extra information instead of preprocessing everything for viewing - When the user is not doing anything directly, you can process information in the background and be ready for the next actions the user may take - Monitor how long your application has taken and if it falls into an acceptable time duration

Appendix

Well-Known User Interface Design Rules

The following are the major tips compiled from all appendix items without duplicates

- Always provide the user feedback in a way that they cannot be confused about the systems state
- Different types of interactions should use different styles of menus so they can be uniquely identified by the user in memory
- Always provide the ability to undo actions
- System should do stuff in a consistent way so the user has to remember less of the system
- Make the users feel like they are in control at all times.
- Help users recover from errors instead of quitting and making them restart
- Provide lots of documentation for exploration
- Function first, Presentation later
- Design for common case not the complex case
- Try stuff out on users, then fix it