UCI University of California, Irvine

Informatics 134

Software User Interfaces
Spring 2021

Mark S. Baldwin
baldwinm@ics.uci.edu

4/06/2021

Agenda

1. Evaluating Widgets Part 2

2. T1: Web Frameworks and Graphical Toolkits

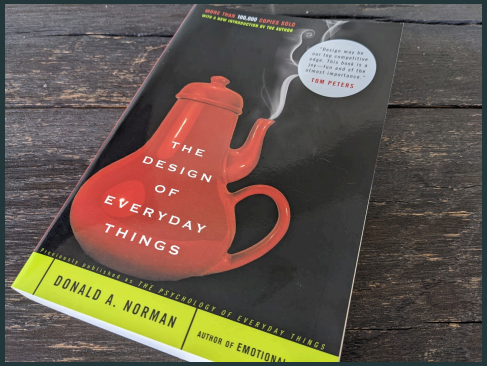
3. Next Class

Evaluating Widgets Part 2

Design Principles

On Execution and Evaluation

- Written by Don Norman (UCSD, nngroup.com)
- The hidden frustrations with everyday things
- Principles for design

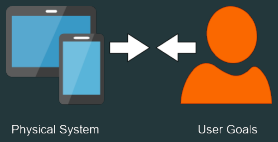


The Design of Everyday Things [Norman, 1988]

On Execution and Evaluation

> "The basic idea is simple. To get something done, you have to start with some notion of what is wanted—the goal that is to be achieved. Then, you have to do something to the world, that is, take action to move yourself or manipulate someone or something. Finally, you check to see that your goal was made. So there are four different things to consider: the goal, what is done to the world, the world itself, and the check of the world. The action itself has to major aspects: doing something and checking. Call these *execution* and *evaluation*."
>
> ——[Norman, 1988], p. 46

Design Principles



Stages of Execution

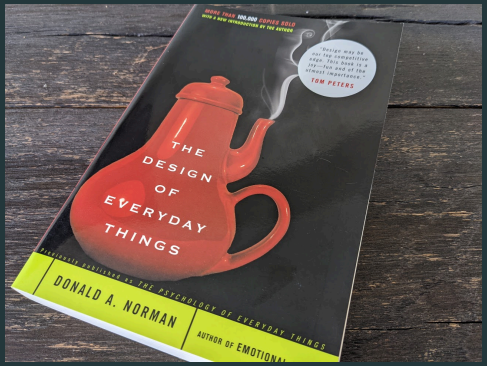
- Identify a goal
- Translate that goal to an intention to act
- Identify steps necessary to fulfill the intention

Stages of Evaluation

- Identify user perception of the world
- Interpret perception to meet expectations
- Evaluate by comparing expectations to execution of intention

Design Principles

- Natural Mapping
- Visibility
- Feedback
- Affordance
- Constraints
- Mental/Conceptual Models



The Design of Everyday Things [Norman, 1988]

Natural Mapping

The relationship between two things.

The relationship between controls, their manipulations, and the results in the world.

Design Principles

Visibility

Make capabilities perceivable and interpretable.

Counteracting factors: features, aesthetics, and abstractions.

Visibility Examples

When the number of functions is greater than the number of controls, functionality is hidden.

When capabilities are visible, memory is not required to use (“recognition over recall”).

Feedback

Sending back to the user information about what action has actually been done and what result has been accomplished (*e.g.*, sounds, change in physical state)

Design Principles

Someone is Typing...

Affordance

Perceived and actual properties or clues about something that determine just how that thing could possibly be used.

Note:

Affordance != Features

Design Principles

Norman's pet peeve:

[Huesler, 2020]

The point is...

Complex things *may* need explanation, but simple things *should* not.

If a simple thing requires instructions, it is likely a failed design.

On Affordances

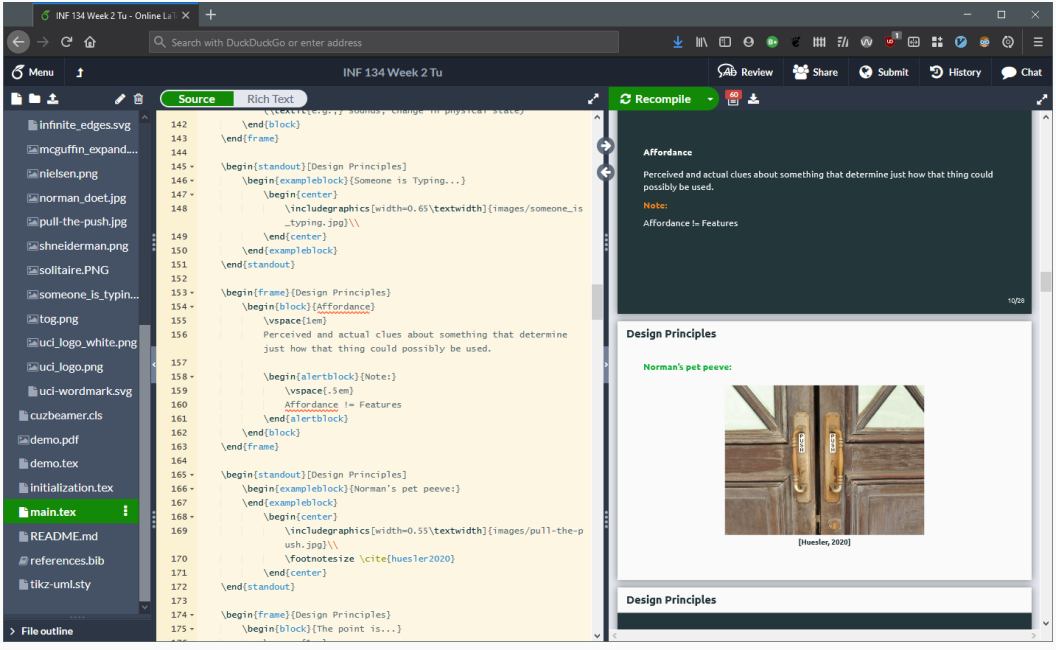
> “Affordances provide strong clues to the operations of things. Plates are for pushing. Knobs are for turning. Slots are for inserting things into. Balls are for throwing or bouncing. When affordances are taken advantage of, the user knows what to do just by looking: no picture, label, or instruction needed.”
>
> ——[Norman, 1988]

Affording Widgets...

Does it afford:

- “clicking”?
- “dragging”?
- “pulling”?
- “sliding”?
- “swiping”?
- “spinning”?

Design Principles

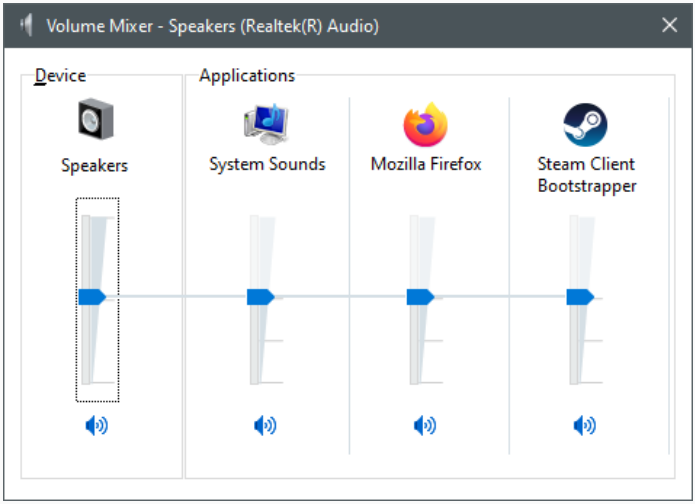


Constraints

“Physical” or psychological limitations that constrain possible actions.

Examples?

Design Principles



Conceptual Models

People build their own understanding of how things work by building a conceptual model around...

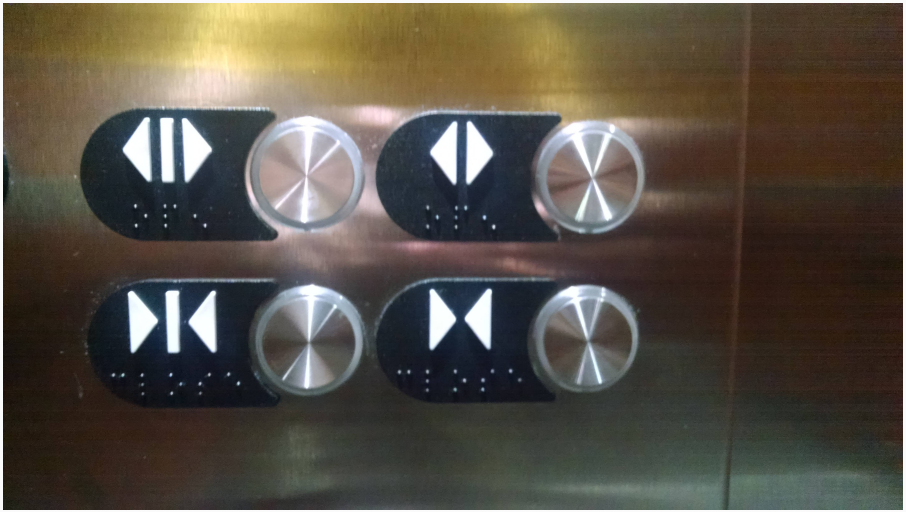
What?

Conceptual Models

People build their own understanding of how things work by building a conceptual model around...

- Mappings
- Visibility and Feedback
- Affordances
- Constraints

Design Principles



[Reddit, 2021]

Conceptual Models

People are explanatory (usually)

- Sometimes they get things right...
- Sometimes they blame the wrong cause...
- Sometimes they blame themselves (learned helplessness)...

Designing Interfaces

Designers (and programmers!) should work to foster the appropriate conceptual model

1. How does something actually work?
2. How does the user think the thing works?
3. How should the user conceptualize about 1?

T1: Web Frameworks and Graphical Toolkits

Web Framework vs. Graphical Toolkit

Web Framework

- Web based!
- Designed to reduce overhead in building web-based products.

Graphical Toolkit

- Native (runs on desktop/mobile not browser)
- Traditional tool used to create GUIs for desktop/mobile apps

T1: Requirements

Getting Started

- Independent exploration, find 2-3 examples of web frameworks and native toolkits that you find interesting.
- Meet with your team, discuss findings, and settle on the tools that you collectively want to report on.
- Sign up on the course spreadsheet! No duplicates!
- Identify 1 web framework and 1 native toolkit.
- Build a slide deck to share your findings.
- Each team will present during class.

T1: Requirements

Features

- History
  - When was it launched?
  - Why was it launched (*e.g.*, what perceived problem was it created to solve)?
  - What tool(s) are used to develop with it?
  - What OSes does it run on (native)
  - What is the license?
  - Who are its backers?
- Features
  - What programming language(s) does it use?
  - What are its advantages over similar frameworks?
  - What are the trade-offs?
- Sample Code (show, don't implement)
  - A button and corresponding click event.

Next Class

Thursday

- Check-in and team work time
- Keep working on A2
- Get started on T1 (DUE 4/12)

References

References i

- Huesler, S. (2020).
  Design undusted: Norman doors.
- Norman, D. A. (1988).
  The psychology of everyday things.
  Basic books.
- Reddit (2021).
  Reddit.