Talk to People!

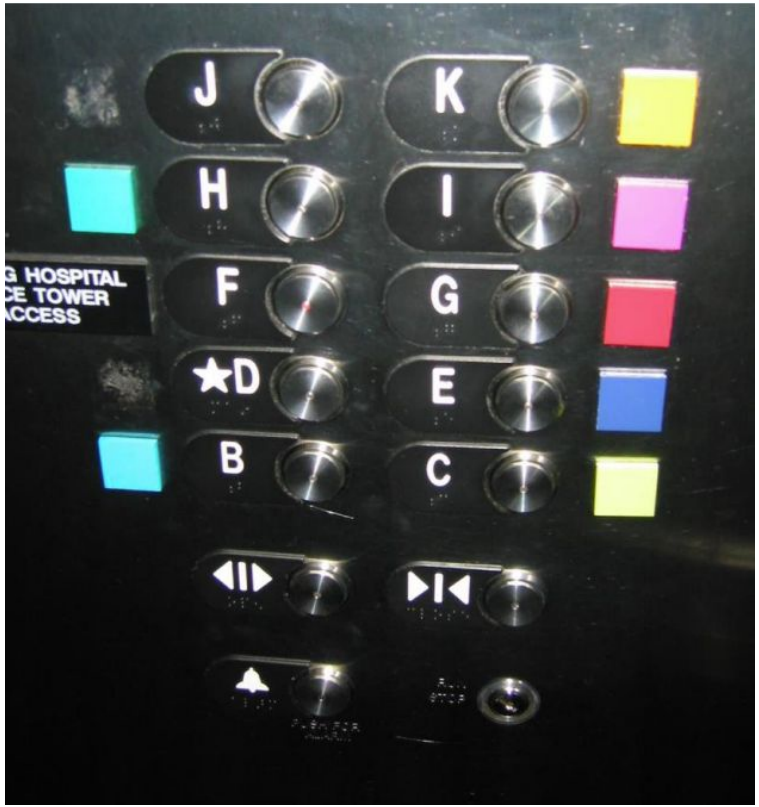
17-313: Foundations of Software Engineering

What is Usability

“The extent to which a product can be used by specified users to achieve specified goals with **effectiveness, efficiency and satisfaction** in a specified context of use”

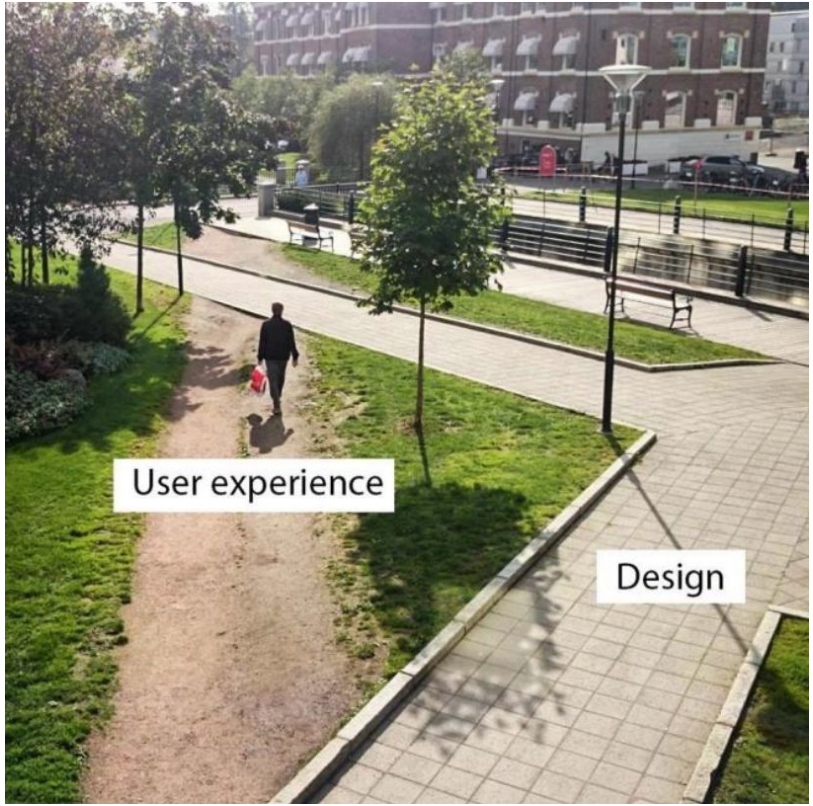
How Usable are These?





Design for People





Usability in Programming Tools

```
 + 1311c40...d5b9653 gh-pages   -> origin/gh-pages  (forced update)
hint: You have divergent branches and need to specify how to reconcile them.
hint: You can do so by running one of the following commands sometime before
hint: your next pull:
hint:
hint:   git config pull.rebase false  # merge
hint:   git config pull.rebase true   # rebase
hint:   git config pull.ff only       # fast-forward only
hint:
hint: You can replace "git config" with "git config --global" to set a default
hint: preference for all repositories. You can also pass --rebase, --no-rebase,
hint: or --ff-only on the command line to override the configured default per
hint: invocation.
fatal: Need to specify how to reconcile divergent branches.
michaelhilton@MICHAELHMACM1 cmu-17-356.github.io % git config pull.rebase false
michaelhilton@MICHAELHMACM1 cmu-17-356.github.io % git pull
fatal: refusing to merge unrelated histories
```

How do decide what to build next??

Test with actual USERS

- Put your product in from of actual people
- Can they figure it out?
- How long does it take them?
- Do they need help?
- Any complaints?

Don’t blame the users…. Blame the design and *fix it!*

How do you measure usability

- How do you measure if it’s usable?
- How do you test it with people?

Interviews & Survey

- Open-ended questions
- Users will surprise you
- Don’t ask leading questions
  - “How much do you love my app”
- Responses don’t always match user behavior
- **Interviews**
  - Deeper and more detailed
  - time-consuming and difficult to scale.
- **Surveys**
  - Scalable
  - Responses may lack depth

Prototyping

- Implement an ***incomplete*** version of the product
- Let users interact with it
- Ask questions such as:
  - Is this what you expected?
  - Is anything missing?
  - Is the design easy to understand?
  - How would you expect this feature to work?
- Come up with / update **requirements**

Aspects of Usability

Learnability



Efficiency



Satisfaction



Memorability



Errors



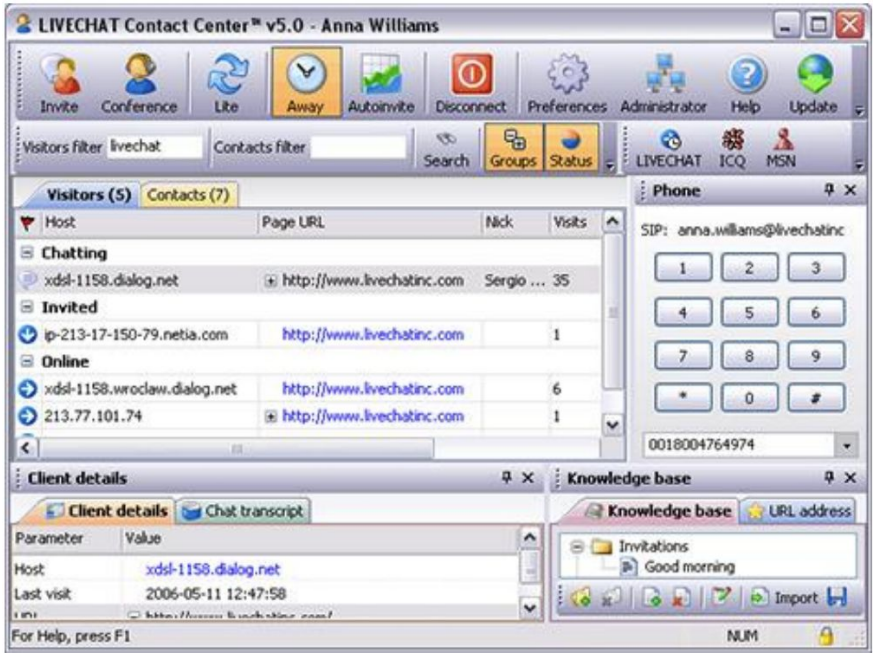
Learnability

- Can users learn how to use your product / feature correctly?
  - Can unfamiliar users pick it up (or even discover it exists)?
  - Can **power users** learn to become experts with it?

- How can we provide users with guidance on how to use the product / feature
  - They won’t read a whole book!

Learnability

- **Design Constraints**
  - Don’t overwhelm the user
  - Make it easy to find features
  - Let novices learn



Efficiency

- How fast can users perform the correct interaction
- How do you design interfaces for faster interaction
- Let experts be fast

Errors

- Is it unlikely that users will make errors?
- Is it easy to recover from errors?

Make it hard to make mistakes

- Unambiguous buttons
- Clear descriptions
- No surprises!
- Every action should be revertable!
  - Deleting a file: Instead of permanently deleting it immediately, move it to a Recycle Bin where the user can restore it.
  - Editing text: Provide an Undo button (Ctrl + Z) to reverse changes.
  - Sending an email: Provide an option to undo sending for a few seconds.
  - Moving a file: Allow users to move it back to its original location.

Make it hard to make mistakes



Interviews: From opinions to requirements

P4A: Elicit Requirements

- Pick one OpenCode feature (that you implemented)
- Write an interview script
- Interview three different types of users
- Turn what you learned into testable requirements
- **Due Monday November 9th, 11:59 PM**

What makes a requirement useful

- **Testable:** Someone can check it and says “holds” or “doesn’t hold”
- **Traceable:** Points to the interview lines it came from
- **Situated:** Names the input, state, or action
- **About behavior:** What the user sees, not how to build it
- **Acceptance Criteria =** a concrete situation + expected outcome
  - When condition X is met
  - The system performs action Y

Three rules for interviews

- Talk about their life, not your idea
- Ask about specifics in the past, not opinions about the future
- Talk less, listen more

Who should you interview?

- A classmate who is not on your team
- A member of course staff
- Someone outside the course

Today's Activity

- Start discussing the features and interview scripts

- **Deliverable for bonus:** Upload your P4A Workbook as a PDF to gradescope.
  - ***Due by 6pm today.***