SWE 432 -Web Application Development

Fall 2022

George Mason University

Dr. Kevin Moran

Week 13: Interaction Techniques

Administrivia

- *HW Assignment 4* - Due soon!
- *HW Assignment 5 -* Out now, Due in 2 weeks (December 1st)!

Class Overview

- ***Part 1:*** Interaction Techniques
  - Quick Lecture
  - Designing Alternative Interactions Activity

Interaction Design Overview

Identifying Actions

Goals → Action Sequence

Signifiers

Is this a button?

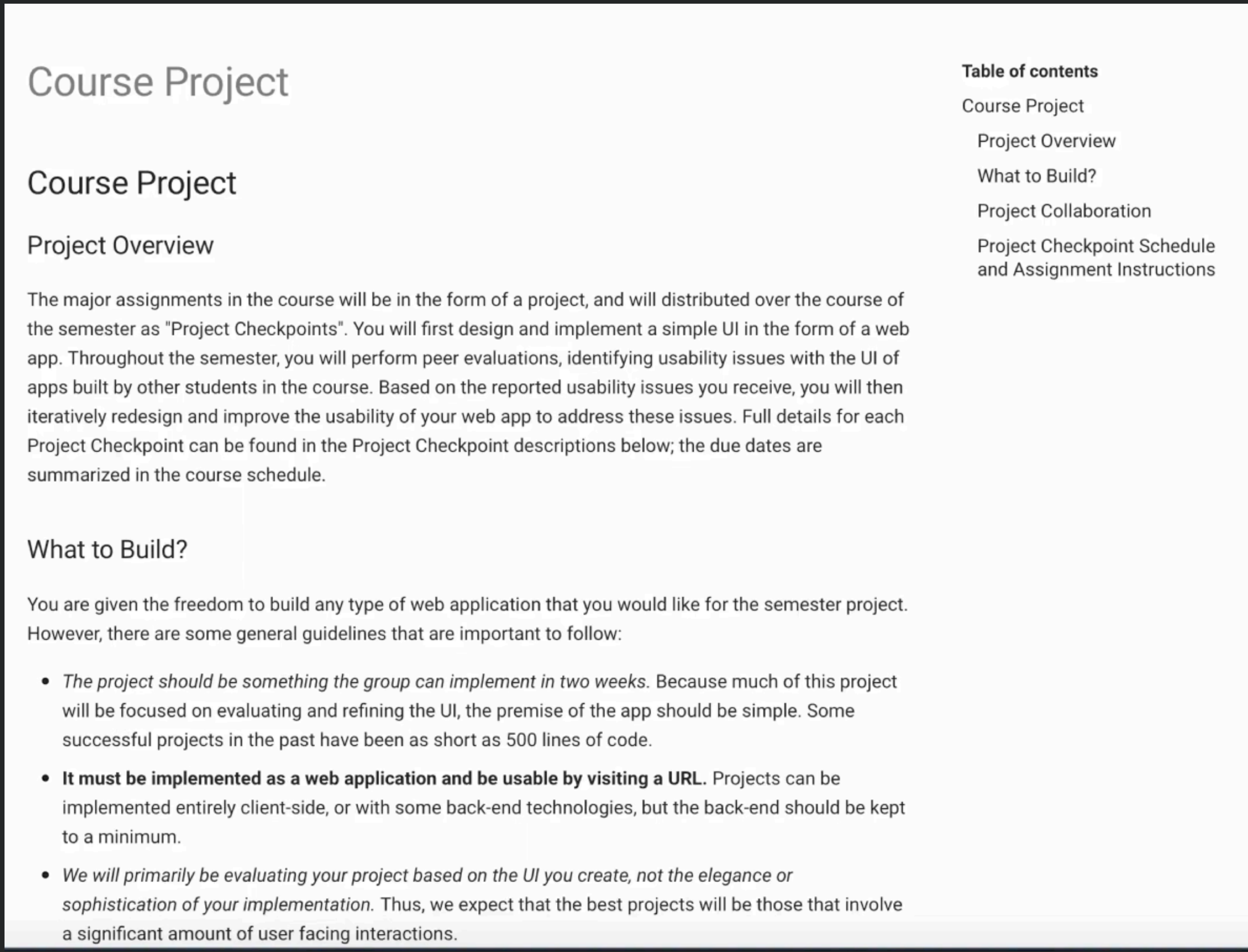
Or a link?

- 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



Course Project

Course Project

Project Overview

The major assignments in the course will be in the form of a project, and will distributed over the course of the semester as "Project Checkpoints". You will first design and implement a simple UI in the form of a web app. Throughout the semester, you will perform peer evaluations, identifying usability issues with the UI of apps built by other students in the course. Based on the reported usability issues you receive, you will then iteratively redesign and improve the usability of your web app to address these issues. Full details for each Project Checkpoint can be found in the Project Checkpoint descriptions below; the due dates are summarized in the course schedule.

What to Build?

You are given the freedom to build any type of web application that you would like for the semester project. However, there are some general guidelines that are important to follow:

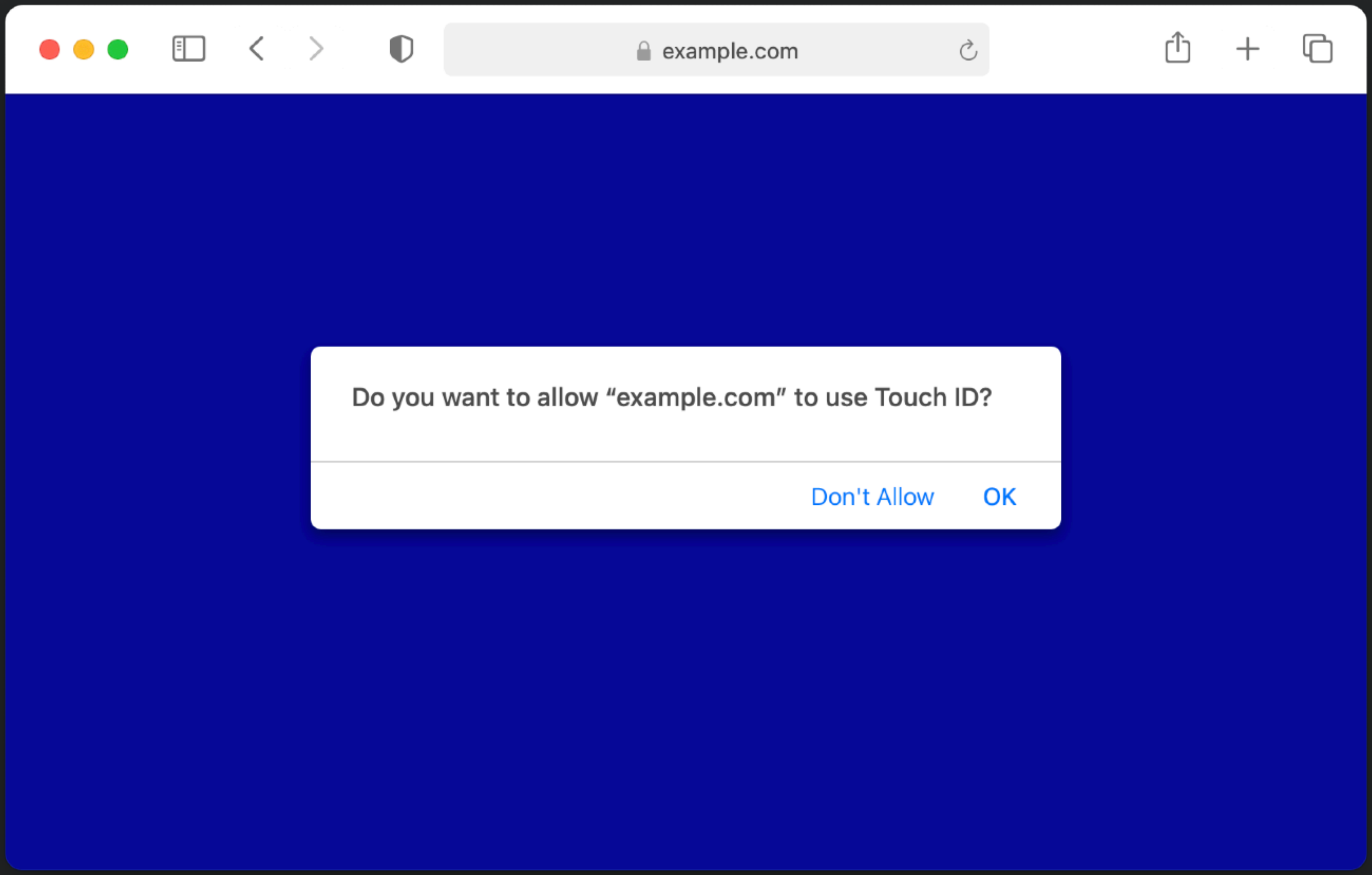
- *The project should be something the group can implement in two weeks.* Because much of this project will be focused on evaluating and refining the UI, the premise of the app should be simple. Some successful projects in the past have been as short as 500 lines of code.
- **It must be implemented as a web application and be usable by visiting a URL.** Projects can be implemented entirely client-side, or with some back-end technologies, but the back-end should be kept to a minimum.
- *We will primarily be evaluating your project based on the UI you create, not the elegance or sophistication of your implementation.* Thus, we expect that the best projects will be those that involve a significant amount of user facing interactions.

Help Users Predict Outcome of Actions

- What does this do?
- Should I click it?

Clarity of Wording (Bad Example)

- Design for clarity & precision



Do you want to allow "example.com" to use Touch ID?

Clarity of Wording

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

Likely & Useful Defaults

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

Modes

- Vary the effect of a command based on state of system
- Examples
  - caps lock
  - insert / overtype mode
  - vi / emacs command modes
  - keyboard entry used for controlling game and chatting

Challenges with 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
  - Make clear to user which mode they are in and how to change

Command Interactions

- How can a user invoke a command?
- Common examples
    - Menus
    - Buttons
    - Toolbar
    - Dialog box
    - Keyboard shortcut
    - Gesture

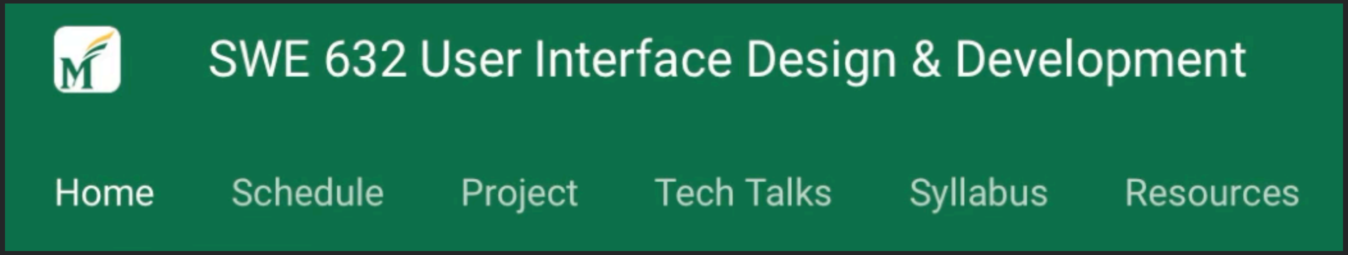
- What are some advantages and disadvantages of each approach?

Physical Actions

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

Moving the Mouse

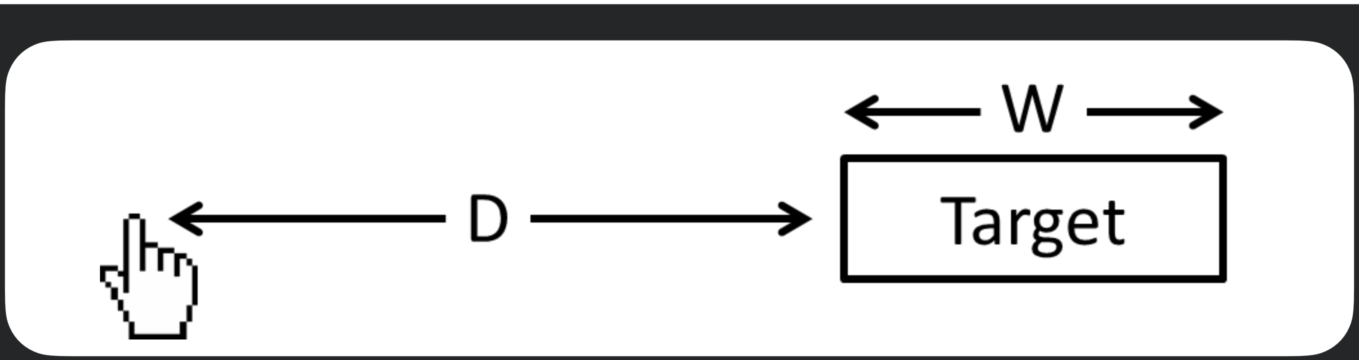

SWE 632 User Interface Design & Development

- After a user has (1) realized that a region is interactable, (2) decided that it will cause the desired action to be invoked

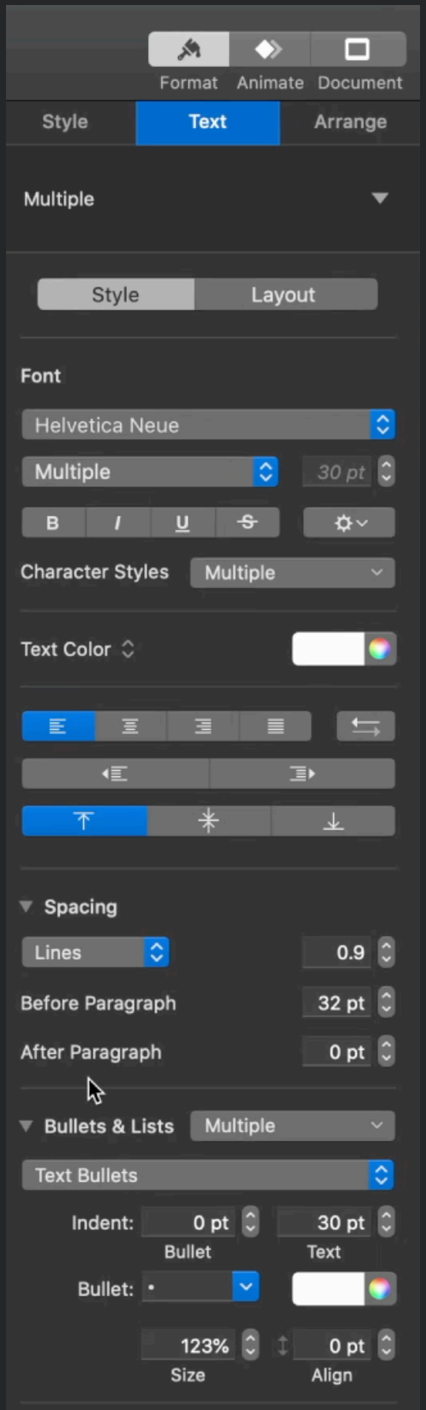
- How long does it take for a user to move the cursor to click on it?

- What factors might influence this time?

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



Multiple

Font

Helvetica Neue

Multiple 30 pt

Character Styles Multiple

Text Color

Spacing

Lines 0.9

Before Paragraph 32 pt

After Paragraph 0 pt

Bullets & Lists Multiple

Text Bullets

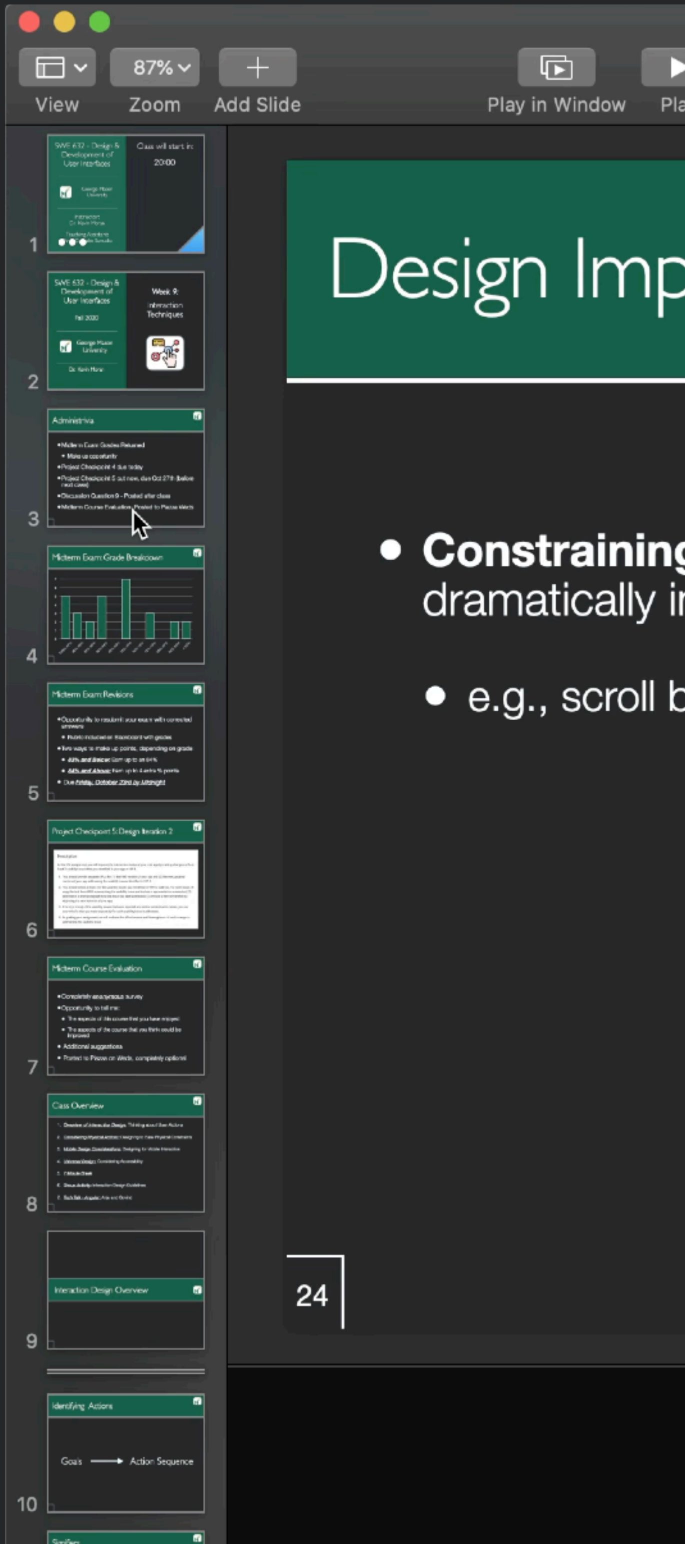
Indent: 0 pt Bullet 30 pt Text

Bullet: •

123% Size 0 pt Align

Design Implications of Fitt's Law

- ***Constraining*** movement to one dimension dramatically increases speed of actions
    - e.g., scroll bars are 1D



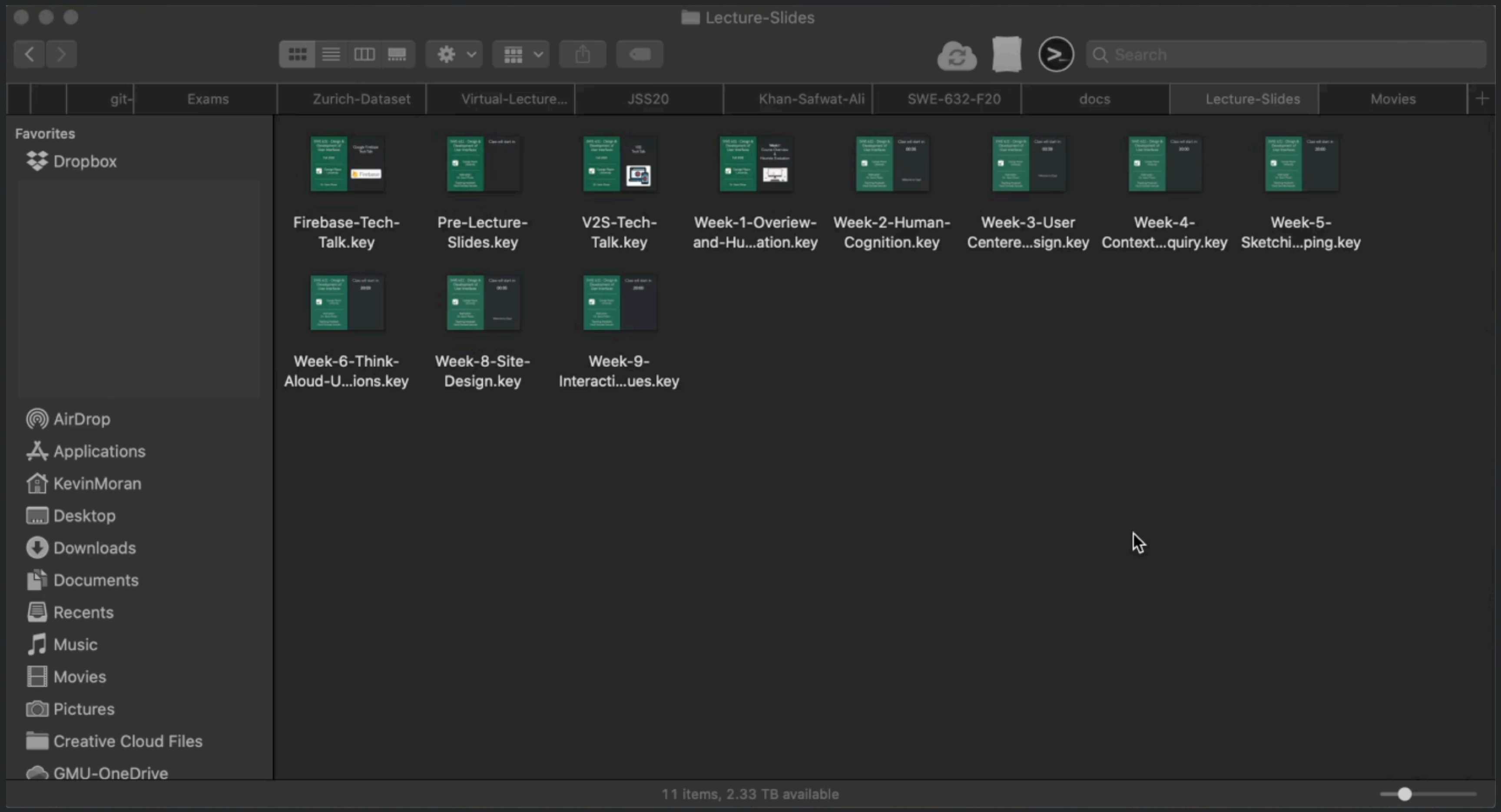
Design Imp

- **Constraining**
  dramatically i
    - e.g., scroll b

24

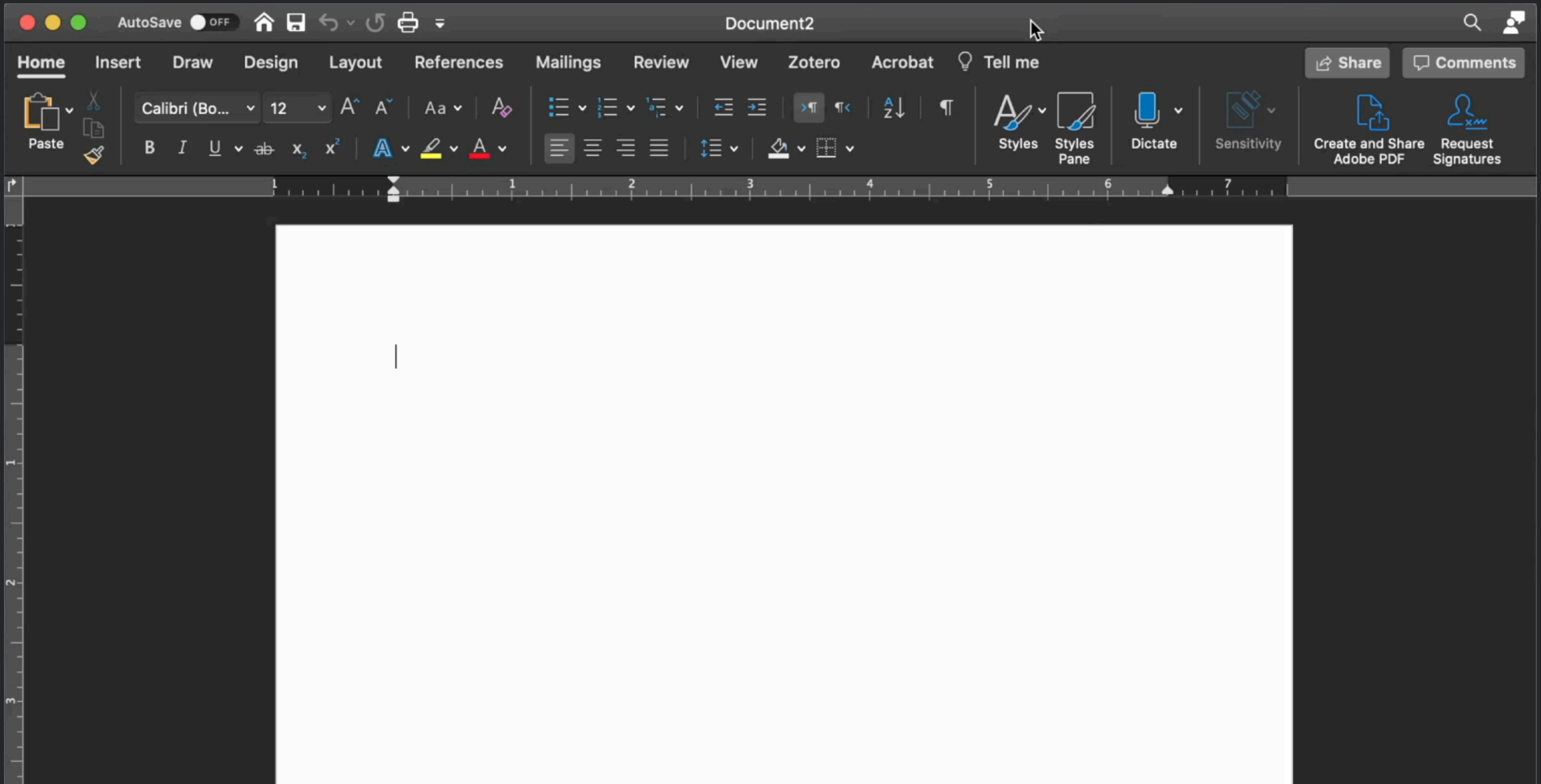
Design implications of Fitt's law

- 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



Mobile Design

Responsive Design

- Mobile devices often have smaller form factor than desktop / laptop OS
- Can design a separate UI
- Or may build a ***fluid*** UI that rescales for different display sizes

Where's the Cursor?

- No cursor on many mobile devices
- Cannot use dynamic hinting to determine which elements can be interacted with
  - May require more use of static hinting
- Fitt's law still applies
  - Fingers are less sensitive, hard to select small buttons, occlude elements

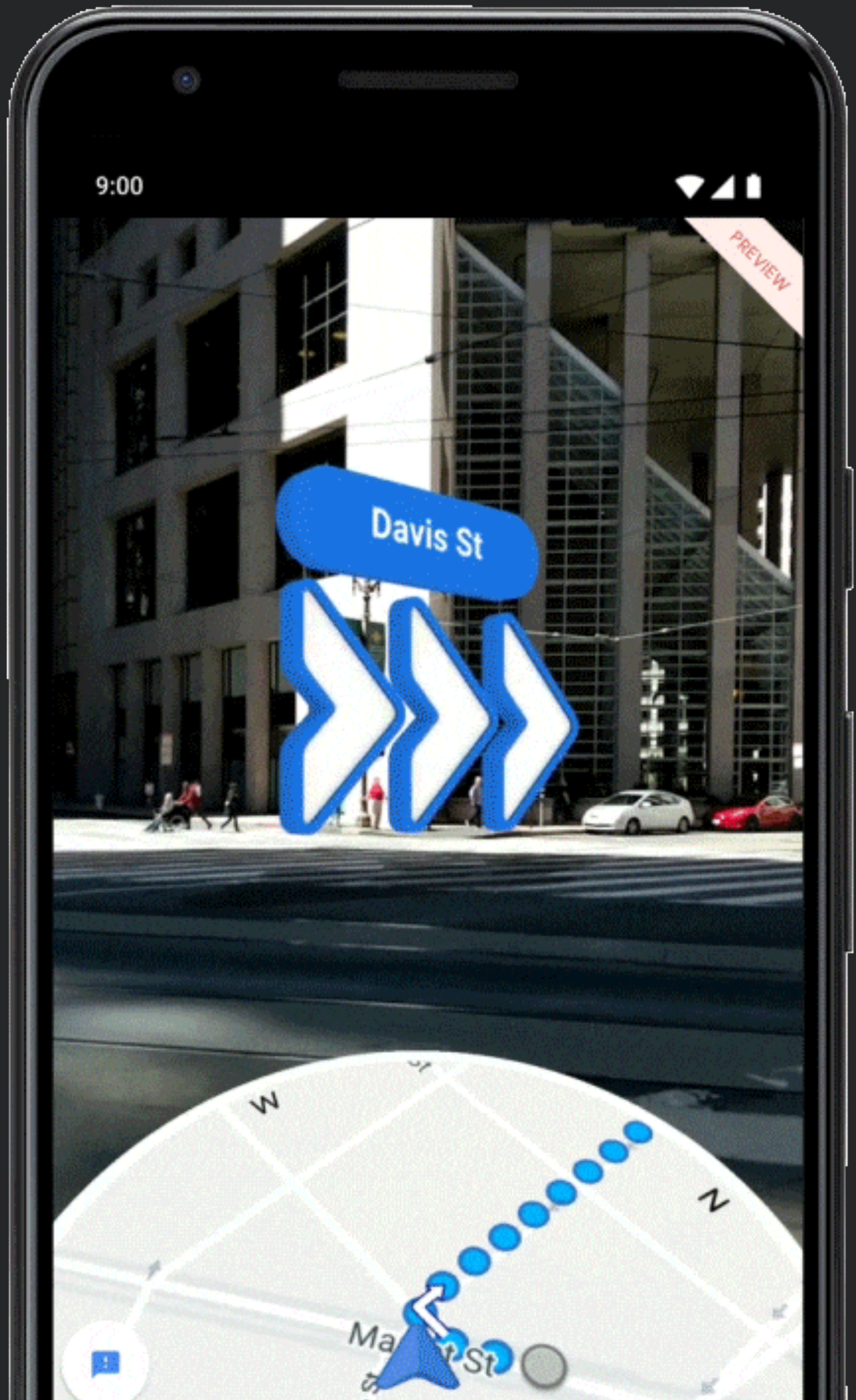
Alternative Inputs

- Modern mobile devices often have a wide range of sensors which can be used for input
  - Camera
  - Microphone
  - Accelerometer
  - Three-axis gyro
  - GPS
  - Barometer
  - Proximity sensor
  - Ambient light sensor
- Enables new interaction techniques

Augmented Reality

- Overlaying generated content on top of view of the real world

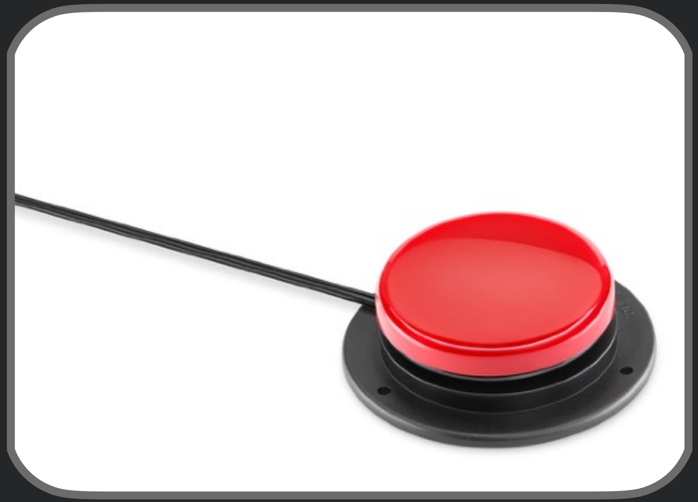
Alternative Inputs + Augmented Reality



Universal Design

A Personal Subject for Me…

+



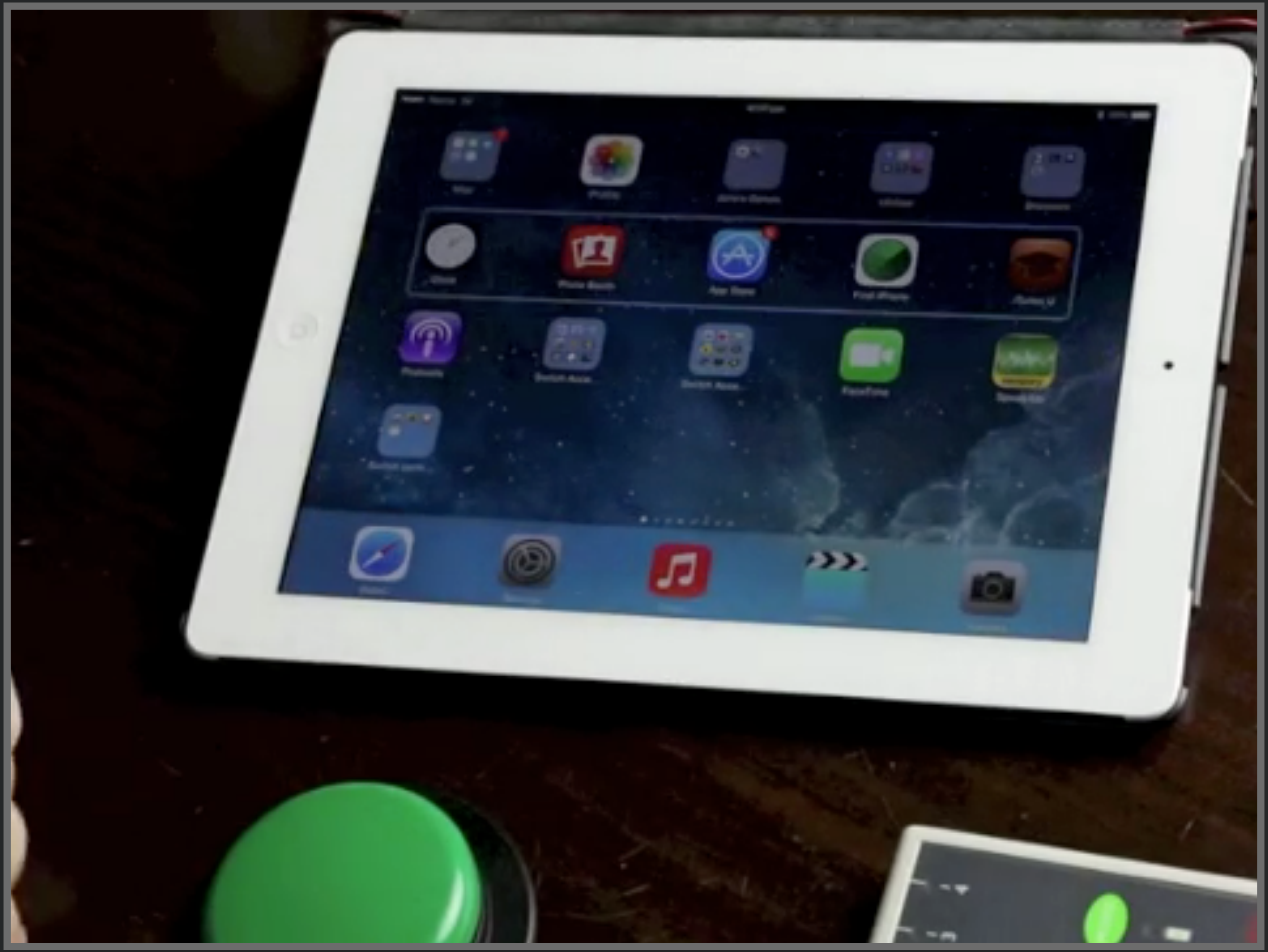
Supporting Users with Disabilities

- **Perception** - visual & auditory impairments
  - Blindness or visual impairments
  - Color blindness
  - Deafness & hearing limitations
- **Motion** - muscle control impairments
  - Difficulties with fine muscle control
  - Weakness & fatigue
- **Cognition** - difficulties with mental processes
  - Difficulties remembering
  - Difficulties with conceptualizing, planning, sequencing actions

Blindness and Visual Impairments

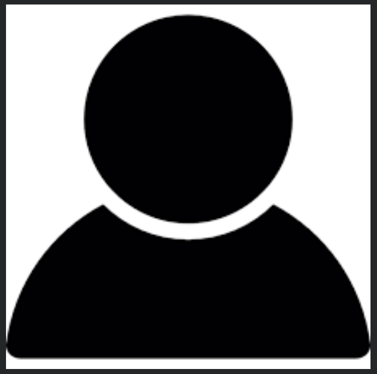
- Users use screenreader to listen to screen elements
- Reads all of the text on the page
  - Through practice, learn to listen to text at 400+ words per minute
- Important to have ***alt-text***
  - Images should have labels that explain them
- Important to have ***hierarchy***
  - Rather than visually skimming page, skims page by listening to section heads to determine which level to navigate to next

Motion Impairments



Universal Design

- How can users with physical disabilities be supported in user interactions?
- Good: ***assistive design*** - offering equivalent actions for disabled users that cannot take normal actions
- Better: ***universal design*** - designing interactions so broadest set of users across age, ability, status in life can use normal actions



Example - Curb cut

- Initially designed for ***<u>accessibility</u>*** - support for disabled & wheel chairs
- But potentially benefits ***<u>all users</u>*** of public spaces - people w/ suitcases, hand carts, roller blades, bikes, …

7 Principles of Universal Design

- ***Equitable use:*** The design is useful and marketable to people with diverse abilities
- ***Flexibility in use:*** The design accommodates a wide range of individual preferences and abilities
- ***Simple and intuitive:*** Use of the design is easy to understand, regardless of the user's experience, knowledge, language skills, or current concentration level
- ***Perceptible information:*** The design communicates necessary information effectively to the user, regardless of ambient conditions or the user's sensory abilities
- ***Tolerance for error:*** The design minimizes hazards and the adverse consequences of accidental or unintended actions
- ***Low physical effort:*** The design can be used efficiently and comfortably and with a minimum of fatigue
- ***Size and space for approach and use:*** Appropriate size and space is provided for approach, reach, manipulation, and use regardless of user's body size, posture, or mobility

http://universaldesign.ie/What-is-Universal-Design/The-7-Principles/

Big Topic - Further Reading

Jeff Bigham's Course at CMU: http://www.accessibilitycourse.com

Amy Ko's Book Chapter on Accessibility:
https://faculty.washington.edu/ajko/books/user-interface-software-and-technology/#/accessibility#ref-islam10

In-Class Activity

In-Class Activity: Interaction Design Guidelines

- Envision a fictional app (e.g., a mobile AR tour-guide app for visiting Antartica)
- Build a list of alternative interaction techniques for your category
  - Identify examples from desktop / web / mobile apps
- Describe pros and cons of each for your design context
- Describe how you will support mobile and universal design

- ***(1) Navigating lists of items***
  - Examples: grids, lists, pages of results, infinite scrolling, filtering
- ***(2) Invoking commands on content***
  - Examples: toolbar, floating toolbar, cards, context menu, sidebar pane
- ***(3) Invoking top level commands***
  - Examples: drawers, toolbar, menus, dialog
- ***(4) Entering formatted text***
  - Examples: toolbar commands, Markdown, HTML
- ***(5) Panning and zooming***
  - Example: zoom slider, scrollbars, pinch to zoom, drag to pan
- ***(6) Accelerometer-based control***
  - Examples: shake to undo, rotate to pan, roll / pitch / yaw game control
- ***(7) Chat bots***

Acknowledgements

- Slides adapted from Dr. Thomas Latoza's SWE 432 course