CSCI 4120 – Mobile App Development

Lab 2: Essential Design Principles Video Discussion

Video: https://developer.apple.com/videos/play/wwdc2017/802/
Project Journal: https://ryansprojectjournal.wordpress.com/

11 core design principles in the video:

- **Wayfinding** – Helps us feel safe by answering these questions: where am I, where can I go, what will I find when I get there, what's nearby, how do I get out?
    - Wayfinding systems help people to navigate complex environments quickly and successfully. They are critical for helping people feel oriented and safe.
    - Good wayfinding systems offer a comprehensive and understandable list of general locations that people can visit. They provide needed detail about what's in those locations. They are highly contextual. They become increasingly specific as we navigate deeper and deeper into the system. They help people to orient themselves by clearly highlighting their current location relative to other locations. And they provide a clear exit path. It's so comforting to know that you can return to a place that is more familiar and start over.
    - The navigation bar, the content area, the tab bar, they're all just tools for providing wayfinding in your app. The navigation bar title and selected tab bar item lets people know where they are. Tab bar in content areas lets people know where they can go and what's nearby.
    - Simply rendered and recognizable tab bar glyphs and understandable labels suggest to people what they will find when they get there. And back buttons offer an exit route, and can help people to identify what area of the app that they're in. Now every screen in your app should answer all of these questions, or else people will feel lost.
- **Feedback** – Should be clear, immediate, and understandable.
    - Showing status clearly and directly is crucial for saving people time and helping them to avoid making mistakes.
    - Completion feedback is discreet, and they're non-invasive, but they're hard to ignore, and they provide that necessary reassurance that our devices are performing as expected.
    - Every action taken in your app should provide some form of confirmation feedback, because it is absolutely necessary for letting people know that actions they've performed were successful.
    - Warnings help people to understand in advance about potential errors. Warnings can be communicated using status indicators, messages, and the instrument cluster, built-in display, sounds, or all of the above.
    - Getting this feedback in real time helps people to course correct so that they don't run into errors later.
    - Clear, timely, and understandable and informative feedback is essential. Feedback answers super important questions: What can I do? What just

happened? What is currently happening? And what will happen in the future?

- **Visibility:**
  - Good feedback does not matter very much if people can't see it.
  - The usability of a design is greatly improved when controls and information are clearly visible.
  - Removing that information might reduce clutter. It could look a little bit cleaner, but it would greatly reduce usability.
  - So it is important to surface key status information at higher levels whenever possible.
  - Densely packed interfaces can be overwhelming and they can slow decision-making, especially for people who are new to your app. So visibility has to be weighed against other considerations.
  - Visibility improves usability!
- **Consistency:**
  - The principle of consistency is about representing similar design features in similar ways.
  - You have to make a conscious effort to be consistent, and you have to be consistent with the right things. You have to fully consider what expectations people have when they come to your app.
  - Those expectations are mostly informed by their experiences with the other apps that they use. So you should pay attention to platform conventions. Things like iconography and terminology, navigation schemes, and even common work flows for typical tasks.
  - From the perspective of the person who you are designing for, it is best to be consistent with the symbol that they are most familiar with.
  - Being consistent will go a long way towards making your app more approachable and usable. Now, look, there is always the impulse to, you know, do things a little bit different.
  - And to be clear, this is a really good thing. You should absolutely be trying out new ideas. That's how innovation happens. But inconsistencies about simple things, like icons and labels, can really trip people up. So it's always best to be consistent unless you have a very strong justification to do otherwise.
  - Internal consistency is about designing controls to share a similar look and feel, that match each other. Your app's glyphs should have a consistent visual style. Text in your app should have a limited number of font faces and sizes and colors and so forth. Internal consistency helps to make an app feel cohesive or whole. When everything matches, when everything fits, people are given a deeper sense of a product's integrity. We intuitively believe that design choices were deliberate and thoroughly considered. And with very good reason. Being consistent takes self-control and restraint.
- **Mental Models** - They don't fully capture the inner workings of these systems. Though, the more experience you have with a system, the more sophisticated your model becomes. Mental models are developed through personal

experience, and they're based on an incomplete set of facts, so everyone's mental model is unique.
    - System Model - Our system model might involve thinking about two independent sources of water, one hot, and one cold. And the system allows those inputs to be mixed, to create a range of temperatures. The system is not immediately responsive. Changes to temperature could take a little bit of time, especially when we first turn on the faucet, right? Our system model includes this understanding of variable latency and temperature.
    - Interaction Model – A model about how we interact with the system. Our interaction model is that we adjust the temperature and water flow with the handles that are provided.
    - When this happens, and we're not really conscious of our own expectations, we perceive the system to be intuitive. Conversely, when a system doesn't match our mental model, it breaks our expectations, and we perceive it to be unintuitive.
    - Labels and subtle variations of form are design cues that can easily be overlooked, especially when people have deeply ingrained notions about how a system works, and how they interact with it.
    - Changes to any long-lived existing product will inevitably be hard for people to adjust to. It really doesn't matter how good or how necessary those changes may be. When considering big changes to an existing app, you have to be 100 percent positive that those changes will lead to clear wins for the people who currently use your app.
- **Proximity** - Proximity is about the distance between a control and the object that it affects. The closer a control is to that object, the more we assume there to be a connection between the two.
    - Of course the bathroom light switch is in the bathroom. Of course the hallway light switch is in the hallway. Of course the bedroom light switch is in the bedroom. Greater proximity also makes ergonomic sense. In general, the closer you are to an object or region of interest, the more likely you are to want to interact with it and control it.
    - Proximity is also useful for expressing relationships between controls.
    - So, for example, if you see a set of switches on the wall, and you know that one of them controls lights, then it's reasonable for you to assume that every switch is a light switch.
- **Grouping** - Grouping helps people to understand the relationships between elements, and it is key for giving a design structure.
    - Typically goes hand in hand with proximity
    - If one of the previously mentioned light switches controls the shade, then it would be better to separate it out from the rest. This configuration makes it easier for people to remember which switch controls the shade, and which ones control the lights.
- **Mapping** - Mapping is about designing controls to resemble the objects that they affect.

    - Shades go up, and shades go down. So it makes sense to use a control that mirrors that up and down movement. There is no ambiguity about how to raise or lower the shade.
    - Mapping also relates to how controls are arranged relative to each other. Their order should resemble the configuration of the objects that they affect.
    - By focusing on mapping, it is so much easier to make choices about where to position controls, how to order them, and even which controls to use.
    - Providing the ability for people to directly manipulate an object is more straightforward, it's more intuitive, and precise.
- **Affordance:**
    - Our notions about how we interact with this plate are called affordances. In other words, a plate's physical characteristics provide visual and tactile cues about what interactions the plate afford to us. A plate's physical characteristics make certain actions seem possible, and others less possible.
    - Affordance is not an attribute of the object itself. It is more about the relationship that individuals have to an object. Affordance varies based on one's physical abilities. So, therefore, affordance varies from person to person.
    - Now, because affordance is subjective, one person can perceive an affordance when another does not. People are more likely to perceive an affordance when it is related to an action that they are more likely to take.
    - You perceive affordances with every environment and every object that you have ever interacted with. They let people know what actions are possible, and the obviousness and visibility of those cues helps people to know the correct or the intended ways of interacting, and the same should be true for apps.
    - Sliders afford dragging a knob along a track. Dials afford rotating. Buttons afford tapping and clicking. In each of these examples, affordance is communicated with maximum efficiency.
    - Regardless of the exact technique that you use, your app's interface must make clear what action possibilities it affords. If not, people won't know how to properly interact with it. They'll interact in ways that your app doesn't support, and they'll confuse controls for non-interactive objects.
- **Progressive Disclosure** - Progressive disclosure is a technique for managing complexity.
    - Progressive disclosure gradually eases people from the simple to the more complex. And progressive disclosure is also about hiding away complexity, so that people can accomplish basic tasks with simple and more approachable interfaces.
    - Ordering a cheeseburger would be way more complex and daunting if you had to consider all of those options at once. Customizing your cheeseburger is way easier when someone steps you through the process

one decision at a time. And some choices that you make early on could make other choices irrelevant later.
    - Progressive disclosure is a necessary and helpful technique for managing complexity and simplifying decision making. But the same technique can bury information or functionality.
    - Well, discussions about how to properly use progressive disclosure often come down to the 80/20 rule. The 80/20 rule is a design principle that says in essence 80 percent of a system's effects come from 20 percent of its causes. For an app, this might mean that 80 percent of its benefit comes from 20 percent of the actions that it presents, or 80 percent of the people who use an app are only going to use 20 percent of its functions.
    - Not all information or functionality are created equal. Some is much more important than others. So, in order to reduce clutter, and simplify decision-making, it is a really good thing to use progressive disclosure to hide things of lesser importance.
    - Not only does progressive disclosure reduce visual clutter, and make printing less difficult, it also lowers the likelihood of people getting confused and changing a setting that they don't really understand.
    - By keeping things simple for novices, they are less likely to feel intimidated, overwhelmed, or get themselves into trouble, while more experienced users can quickly reveal the options and actions that they require, that they understand, and that they are capable of using.
- **Symmetry:**
    - Symmetrical forms are efficient forms, and we tend to associate them with good health, with stability, balance, and orderliness. And we consider them to be aesthetically pleasing, probably for very good evolutionary reasons. Symmetrical elements, even if they are not physically connected to each other, are perceived as though they are.
    - Attractive interfaces tend to demonstrate a mixture of reflectional and translational symmetry.
    - When laying out your app's interface, look for opportunities to use symmetry to provide a sense of balance and order.
- All of these principles help satisfy the need to feel safe, the need to understand, the need to achieve our goals, and the need to experience beauty and joy.
- It is also possible to have too much of a good thing. Too much feedback is annoying. Too much visibility is distracting. Too much progressive disclosure can make work flows inefficient. So you have to be measured in your approach, and exercise judgment and discretion.
- They will guide you toward making apps that better serve the human beings that you are designing for. And that is the thing that matters most. And in turn, the people that you are designing for will recognize, on some level, all of the hard work that you have invested. They will appreciate the thoughtfulness and care, and they will sense the humanity of the people who made it.