Thinking Like an iOS Engineer

by Devmountain Tutorials

WHAT'S COVERED

This section will explore a sample iOS project by discussing:

1. ADDING OTHER PETS TO THE APP
2. STRUCTURING THE DATA
3. CHALLENGES FOR iOS ENGINEERS
4. LEARNING OPPORTUNITIES FOR iOS ENGINEERS

1. ADDING OTHER PETS TO THE APP

When Poodle Jumper first launched, we only provided services for dogs, but we are expanding to include services for cats, rabbits, hamsters, gerbils, and mice. Last week I finished the changes to the screens, or views as they are known in iOS. Today, I am working on pulling in the new data structure into the app so customers can add their additional pets. I'm going to show you some of that process.

STEP BY STEP

I start by looking at:

- 1. The existing data model I currently have in the app
- 2. The new data model from Monique, our Software Engineer
- 3. The updated web service from Ruben, our Web Developer
- 4. The requirements from Jose, our Product Manager
- 5. The designs from Mori, our UX Designer

I compare the changes and figure out how to structure it in the app. Do I need to pull data in a certain sequence? If so, how can I write that code efficiently to be as fast as possible and only include the data I need? When I have a plan for how to execute this in the app, I collaborate with Ruben (Web Dev) and Monique (Software Engineer) because they provide parts of the puzzle that I need. Monique creates the data model and the main database where we store information. Ruben creates a **web service** for the apps to get that data from the database.

TERM TO KNOW

Web service

A type of electronic communication between devices used to send and receive information, or data.

2. STRUCTURING THE DATA

The web service Ruben created has all the pet fields (name, species, breed, size, type of food, attention needed, and special care notes) in the same service as the user's profile. This means that every time I need to refresh one piece of data for a pet, that I have to get all the information for the user's profile and all the other pets. This isn't the most efficient way to structure the data, so I have a quick conversation with Ruben (Web Dev) and Camilla (iOS). We come up with a way to break the one big service into smaller microservices.

BIG IDEA

Remember, the more data coming in and out of the app, the more networking and data usage it eats up while draining the device battery.

Thinking About Web Service

Knowing the structure of the data allows me to plan how and when I make the app send a request to get the data. This comes with its own set of challenges to overcome.

- What should the app do if the user is offline or doesn't have a stable internet connection?
- What data should I save, or cache, to improve the performance?
- If I cache the data, how often does it need to be refreshed?
- Should I refresh it when the view loads, or will users pull down on the screen to make it refresh?
- What do I show the user if the web service sends back an error message?
- Do I need to wait for the data to return before I show the view?
- What do I need to do to make sure the data is in sync with all their other devices?

All of these questions are considerations I have while writing the code. My goal is to create an experience that is as smooth as possible. I can't lock up the user's device while it loads, so I leverage the phone's multitasking abilities to have several things processing at the same time.

3. CHALLENGES FOR iOS ENGINEERS

I don't have this challenge at Poodle Jumper, but many iOS developers have challenges with their company thinking mobile-first. Mobile-first design means you think about the smallest and simplest version of a feature first so that it will provide a good experience.

It is easy to design large complex screens for a computer, but mobile applications need to be simplified to be usable. This goes beyond the interface. It also includes thinking about how we design the integrations and how we leverage mobile capabilities to have features that aren't possible with only a desktop computer.

Many design elements in mobile apps aren't widely used in web experiences. I get to play with animation, transitions, sound, and haptic feedback (that little vibration you feel from your phone.) I love this aspect of mobile devices because it gives me easy ways to create delightful experiences.

4. LEARNING OPPORTUNITIES FOR iOS ENGINEERS

The biggest challenge I face as an iOS engineer is also my greatest learning opportunity: keeping up with the new devices, operating system versions, programming language versions, and capabilities that change every year. If it sounds like a lot, that's because it is.

Every spring, Apple hosts a World Wide Developer Conference (WWDC) where they announce the new operating system capabilities they've built. Anticipation and rumors build for months. WWDC is full of demos and training led by Apple's employees who created the features, but it is not free.

Smart Watches

Developers get early access to the new features, and we look forward to playing around with them every summer. In the fall, the new OS version is released to users along with a line of updated or new devices. We have to make sure we know the changes and test the app on the new OS before it goes live to users. Otherwise, the app could break for them.

HINT

You may be annoyed when your phone or computer continuously reminds you about a software update, but keeping your devices updated is one of the best things you can do to improve the security. When security vulnerabilities are identified, they are fixed or patched in releases, but if you don't update your devices, the known vulnerabilities could be exploited by a hacker.

You don't need to know everything to get started. Apple provides a lot of resources, training, tools, documents, and even a developer community for us to support each other. You just need to learn how to read documentation, experiment, and try. With everything changing so rapidly, it creates a level playing field for people just starting out. Honestly, this constant state of change is what led me to become an iOS developer. I really love what I do!

TERMS TO KNOW

Web service
A type of electronic communication between devices used to send and receive information, or data.