Why Imitation is the Best Way to Learn

This blog post was translated by ChatGPT 4o.

Reflecting on my journey of learning Android and iOS development, I realized that much of the valuable knowledge I gained was through imitation.

How do you imitate? Start two projects. For example, when imitating Jack' s WeChat Moments implementation, I would look at two lines of his code, then write the same code in my project. If I didn' t understand something, I would go back and look again. If I understood it, I would write more code based on my own ideas.

Using this method, I learned to write Android pull-to-refresh waterfall views and hand-written UIs in iOS, among other things.

The Android pull-to-refresh was the first time I controlled views directly without using XML, and I learned about gestures. It was an improvement. Sometimes, while copying the original author' s code, I would have sudden realizations. Some things cannot be figured out by simply looking at them, no matter how much you think. Because while reading, you overlook a lot of things and miss important details. But by writing the code and handling it yourself, you become more engaged and start to wonder why things work a certain way, paying attention to more details, and eventually understanding them. Like in my previous article on the parallax effect. At first, I didn' t know the background image would move when placed horizontally or vertically. Later, while imitating the code, I noticed many things and learned a method for creating a frosted glass effect.

Previously, I used XIB a lot in iOS. I tried writing code by hand a few times but always gave up. When imitating the Moments project, it felt like I was copying it verbatim. Most of it was identical except for some names. But I still noticed many important aspects of hand-written code. For example, if the height of a list cell changes with the content, you need to calculate its height before drawing it and pass it to the table view. How do you calculate the height? One method is to use a fixed width to calculate the height. These details become apparent when copying the code.

When imitating, I am often more engaged. When reading code, I tend to get sleepy. This is partly because my brain gets tired, and partly because my whole body is still, making it easy to enter a resting state. A friend of mine takes notes while reading code. Someone on Zhihu mentioned solving issues by reading code and getting familiar with the project. These are all good practices; the key is to be actively involved and not just passively reading. Of course, some people can read and think for a long time without getting bored.

Imitating and creating are different. Imitation is easier because if you get stuck, you can simply copy the code as it is. Creating is much harder. Imitating others' apps is easier since you can look at their layouts

and calculate the frames. Because imitation is somewhat easier, it can be sustained longer and doesn't feel discouragingly difficult. However, the knowledge gained from imitation can sometimes be greater than that from creating.

Imitation reduces trial and error and helps you grasp the correct knowledge more quickly. When learning programming, you can work on a project and search Google simultaneously. This often leads to a lot of trial and error, which can be inefficient, especially for beginners. Many trials are meaningless. Sometimes, just knowing the correct approach and why it is correct is enough. Often, during trial and error, you end up randomly trying things, like formatting HTML incorrectly, then making minor adjustments until it looks right. But if you imitate, the correct answers are right there in others' code. You can refer to them and quickly grasp the correct knowledge. For example, with HTML formatting, I would find some well-formatted HTML projects, copy the code line by line, and think about it as I copied. This way, I learn faster and better than if I were to mess around on my own.

Start imitating!