Old vs. New: iOS Development with Objective-C and Swift

Technology is deeply woven into the fabric of today’s society. Computers have rapidly evolved from clunky, room-sized machines to handheld devices with more computing power than the computers used to send man to the moon. At the front lines of mobile device innovation is Apple, an American technology company founded by Steve Jobs. Apple built their empire on top of two powerful languages which yielded their streamline user interfaces and optimized operating systems: Objective-C and Swift. Debate rages on about whether Objective-C is better than Swift and vice versa; formalizing the pros, cons, and real-life applications of each language will help understand the intricacies of said debate.

Objective-C is a superset of C that intermingled the OOP concepts of SmallTalk with the performance of C, which led to a more powerful version of C. In the 1980s, Steve Jobs founded NeXT to build a new age computer, and while looking for a C-like language to build their machines the company decided to use Objective-C. One of the advantages of Objective-C is that “The methods invoked [are] selected “dynamically,” that is, while the program [is] running”(Hsu 2017). This means that a program can change while running according to user input, which is a necessary feature for Apple products because their operating systems and iPhone apps are based off of user interaction with graphical interfaces. NeXT died off and Steve Jobs went back to Apple with all of his NeXT employees, therefore making the main language of all Apple products to be Objective-C. They wrote powerful frameworks such as Foundation, AppKit and CoreData which essentially transformed the language into what it is today. Objective-C had all the features at the right time in history to become the legacy language of Apple.

If developers wanted to write native iOS apps then they would have to code in Objective-C, which brought a lot of attention to the language however it still never became popular outside of the Apple world. One of the biggest complaints about the language is its horrible syntax. Objective-C creator Brad Cox and Tom Love wanted to make a clear distinction between the OOP and C concepts, which they did by bracketing them off from one another, literally. Love contemplated this stylistic choice, but explained his reasoning for their final decision in an interview when he asked “Do we have a C syntax that is consistently C, or do we create a hybrid language where I describe it as "the square bracket is a gear shift into the object land"?”( Biancuzzi & Warden). Though the language is a superset of C, meaning that it inherits all C functionality while extending some extra features, the two languages look nothing alike. The brackets make a clear distinction between the ideologies, but it does so in an ugly way; complex code is riddled with brackets and begins to look unreadable. This con alone was enough to scare away most programmers as they could choose to use a different OOP version of C that had a less steep learning curve.

Objective-C ruled the Apple world for its entire existence, however, in 2014 it was time for a new language to rise. Unlike Objective-C, Swift is an Apple created language that is meant to tackle the deepest pitfalls of Objective-C. The company felt like they had potential for more developers to create apps and software for iOS, and that the ugliness of Objective-C was holding them back. The arrival of Swift has brought in a new flood of developers as it provides a clean, simple syntax and grammar. According to the Swift documentation, “Swift is designed to make writing and maintaining correct programs easier for the developer.” ( <https://swift.org/about/>). Having simpler code satisfies the main goals of software engineering: reduce cost and time to market while improving productivity and quality. Not only is Swift more syntactically polite, it is also better performing. Apple documentation claims that Swift is up to 2.6 times faster than Objective-C and 8.4 times faster than Python 2.7. The advantages Swift provides to the developer has skyrocketed its popularity, earning it a place in one of the most popular languages, despite its young age.

Given the clear advantages that Swift provides over Objective-C, lots of companies choose to not refactor their Objective-C code into its more modern equivalent. Because of the language’s lack of maturity, old school iOS developers see it as a new, shiny toy rather than a legitimate option. While Objective-C has had decades to mature and become a fully tested language, Swift was released in just 2014, therefore is constantly undergoing updates. These updates tend to be rather drastic, which means that developers need to be writing modular code that can be refactored quickly. For example, in October of 2019 Apple released SwiftUI, which holds the heir for creating user interfaces for Apple products. Before SwiftUI, user interfaces were made by simple drag-and-drop graphical interfaces called Storyboards; the developer did not have to write a line of code to create interfaces. This popularized the model-view-controller architecture in iOS apps, which showed its flaws in complex projects as the view logic became repetitive and inefficient. SwiftUI fixed this problem by switching over completely to programmable user interfaces, thus bringing a new model-view-viewmodel architecture. This fundamentally changed how code was written, and if a company so happened to write non-modular code, the refactor process would be horrific. It is because of these problems that certain companies feel more safe sticking with what they know.

In conclusion, Objective-C provided all the features Apple needed and fueled the legacy all the way until 2014, when a new language challenged its throne. Overall the pros of Swift outweigh the pros of Objective-C as it is nearly better in every aspect: safer, faster, simpler. Though Objective-C was not created specifically for Apple, the history of the language makes it seem as if it was, therefore making both Swift and Objective-C best suited for Apple products; Objective-C because of the powerful frameworks created for it and Swift because it was created by Apple to draw in more developers for iOS apps and such. Huge companies such as Uber, Lyft, Slack and many more use Swift to share their products on the iOS platform, while other companies such as Facebook still use Objective-C for their mobile apps. As time progresses, Swift will become more mature and will become the standard way of developing for iOS.

<https://archive.org/stream/MastermindsOfProgramming/Masterminds%20of%20Programming_djvu.txt>