UDC 004.432

SWIFT OBJECT-ORIENTED PROGRAMMING LANGUAGE: FEATURES AND ADVANTAGES

Y. KANIAYEU, I. RUSETSKI
Polotsk State University, Belarus

Swift is a general-purpose programming language built using a modern approach to safety, performance, usability and software design patterns.

Mobile application development industry in the last five years has multiplied in leaps and bounds, changing the way businesses function worldwide. With enterprises aligning mobile apps to their productivity in recent times, and with the rapid innovation in mobile devices across platforms, it calls for mobile app developers to write several versions of an application for many different platforms using a single language and many pieces of reusable code [1].

The iOS platform is a proprietary platform made by Apple. The iOS platform constituents are available for phone devices and tablet devices: Iphone and Ipad. Apps could be developed for the iOS platform, and then target the same app to both an iPhone and an iPad. For building apps for iOS, one must have an Apple developer account and the Xcode IDE on a Mac computer. Xcode comes with all the required Apple development toolkit: SDKs, a code editor, compile/build tools, simulators, and a debugger. Apps can be built for iOS devices either by using the native iOS SDK with Objective-C and Swift or with the various cross platform technologies that are written against the SDK of that framework, but targeted for iOS [2].

Swift is a general-purpose, multi-paradigm, object-oriented, functional, imperative and block structured language. It is the result of the latest research on programming languages and is built using a modern approach to safety, software design patterns by Apple Inc. It is the brand new programming language for iOS application, macOS application, watchOS application and tvOS application. Soon it became one of top 5 programming languages and gained popularity among Apple developer community over the few years of time replacing the old school Objective C [3].

Brief history. Chris Lattner began the development of Swift in the year 2010 and collaborated with other programmers at Apple in the course of the development of this language. The language ideas for Swift were taken from Rust, Objective-C, Ruby, Haskell, C#, CLU, Python, and a range of other programming languages.

Swift was introduced at Apple's 2014 Worldwide Developers Conference (WWDC). It underwent an upgrade to version 1.2 during 2014 and a more major upgrade to Swift 2 at WWDC 2015. Initially a proprietary language, version 2.2 was made open-source software under the Apache License 2.0 on December 3, 2015, for Apple's platforms and Linux.

Through version 3.0 the syntax of Swift went through significant evolution, with the core team making source stability a focus in later versions.

Swift 4.0, released in 2017, introduced several changes to some built-in classes and structures. Code written with previous versions of Swift can be updated using the migration functionality built into Xcode.

Swift 5, released in March 2019, introduced a stable binary interface on Apple platforms, allowing the Swift runtime to be incorporated into Apple operating systems. It is a source compatible with Swift 4. Swift 5.1 was officially released in September 2019. Swift 5.1 is built on the previous version of Swift 5 by extending the stable features of the language to compile-time with the introduction of module stability. The introduction of module stability makes it possible to create and share binary frameworks that will work with future releases of Swift [4].

Key programming features. Swift includes features that make code easier to read and write, while giving the developer the control needed in a true system programming language. Swift supports inferred types to make code cleaner and less prone to mistakes, and modules eliminate headers and provide namespaces. Memory is managed automatically, and you don't even need to type semi-colons. Swift also borrows from other languages, for instance the named parameters brought forward from Objective-C are expressed in a clean syntax that makes APIs in Swift easy to read and maintain [5].

Swift is a protocol-oriented programming language, such programming paradigm is used from the release time of Swift 2.0. in this approach, design protocols are similar to classes but this serves better compared to object-oriented programming. Since the concepts like structs and enums don't work properly as a struct cannot inherit from another struct, neither can an enum inherit from another enum. So inheritance which is one of the fundamental object-oriented concepts cannot be applied to value types. On the other hand, value types can inherit from protocols. The concepts used in protocol-oriented paradigm are: protocol extensions, inheritance and compositions.

Similarly to C# and in contrast to most other OO languages, Swift offers built-in support for objects using either pass-by-reference or pass-by-value semantics, the former using the class declaration and the latter using struct. Structs in Swift have almost all the same features as classes: methods, implementing protocols and using the extension mechanisms. For this reason, Apple terms all data generically as instances, versus objects or values. Structs do not support inheritance, however.

An important new feature in Swift is option types, which allow references or values to operate in a manner similar to the common pattern in C, where a pointer may refer to a value or may be null. This implies that non-optional types cannot result in a null-pointer error; the compiler can ensure this is not possible.

Advantages. The features of Swift are designed to work together to create a language that is powerful, yet fun to use. Some advantages of Swift include:

- Safety. Swift was designed to improve the code safety for iOS products. It was created as a type-safe and memory-safe language. Type safety means that the language itself prevents type errors. The importance of type memory safety is that it helps avoid vulnerabilities associated with dangling or uninitialized pointers. These types of errors are the most common in development and difficult to find and debug. These advantages of the Swift language make it more attractive. Swift, on the other hand, doesn't use pointers. If you miss a pointer in the code, perhaps nil value, the app will crash. This approach allows programmers to find and fix bugs quickly. As a result, the code will be cleaner and easier to understand. Such features as generics, optionals, and type interference make an app developed in Swift less inclined to contain unnoticed bugs. [6]

- Dynamic libraries support. Dynamic libraries are the executable parts of code that can be linked to an app. The difference between dynamic libraries and static libraries is that dynamic libraries can be linked to any program during run-time. The shared code is loaded once and can be used by a large number of programs. This code can be updated, changed or recompiled without recompiling the application that uses this library. Dynamic libraries are automatically included in the AppStore's download package. Static libraries are linked at the last step of the compilation process after the program is placed in memory.

- Perfomance. From its earliest conception, Swift was built to be fast. Using the high-performance LLVM compiler technology, Swift code is transformed into optimized native code that gets the most out of modern hardware. The syntax and standard library have also been tuned to make the most obvious way to write your code to perform the best whether it runs in the watch on your wrist or across a cluster of servers. Swift is a successor to both the C and Objective-C languages. It includes low-level primitives such as types, flow control, and operators. It also provides object-oriented features such as classes, protocols, and generics, giving developers the performance and power they demand.

- Open Source. Swift is developed in the open at Swift.org, with source code, a bug tracker, forums, and regular development builds available for everyone. This broad community of developers, both inside Apple as well as hundreds of outside contributors, work together to make Swift even more stable and powerful. Open-source Swift can be used on Linux to build Swift libraries and applications. The open-source binary builds provide the Swift compiler and standard library, Swift REPL and debugger (LLDB), and the core libraries, so one can jump right in to Swift development.

Despite its tender age and the attendant controversy, Swift already has a number of prominent success stories. Some of the companies that chose the new language are Lyft, LinkedIn, Coursera, Pandora, Vimeo, Twitter, Fitbit, and Groupon. Moreover, Facebook and Uber are reported to have shown significant interest in Swift. Swift has become a more mature language with the latest update, but there are a lot of things to fix. Apple is creating its own ecosystem with a stable ABI over its platforms, but it still lacks tooling and support for earlier versions, which might be fixed in the next releases. Thus, Swift adoption will continue to grow, which soon might lead to a complete displacement of Objective-C as the leading first-class language for iOS mobile application development. Though Swift and Objective-C can coexist, that is libraries written in Objective-C and Objective-C utilities can be used in Swift, Apple is making it very obvious that Swift is the new default choice for developing iOS apps. Swift is an easier, simpler, and a more compact language compared to Objective-C. Objective-C developers should not have any trouble moving over to Swift.

REFERENCES

1. Dzone - Reasons Why Startups Choose Swift. – [Electronic resource] – Mode of access: https://dzone.com/articles/7-reasons-why-startups-choose-swift-over-objective/. – Date of access: 03.03.2020.

2. IBM Developer - Choosing the best programming language for mobile app development. – [Electronic resource] – Mode of access: https://developer.ibm.com/technologies/mobile/articles/choosing-the-best-programming-language-for-mobile-app-development/. – Date of access: 03.03.2020.
3. GeeksForGeeks - Introduction to Swift Programming; - [Electronic resource] – Mode of access: https://www.geeksforgeeks.org/introduction-to-swift-programming/. – Date of access: 03.03.2020.
4. Wikipedia – Swift programming language. – [Electronic resource] – Mode of access: https://en.wikipedia.org/wiki/Swift/. – Date of access: 03.03.2020.
5. Swift – About Swift. – [Electronic resource] – Mode of access: https://swift.org/about/. – Date of access: 03.03.2020.
6. Altexsoft - The Good and the Bad of Swift Programming Language. – [Electronic resource] – Mode of access: https://www.altexsoft.com/blog/engineering/the-good-and-the-bad-of-swift-programming-language/. – Date of access: 03.03.2020.