The Basics

Swift is a new programming language for iOS, macOS, watchOS, and tvOS app development. 그럼에도 불구하고 Nonetheless, many parts of Swift will be familiar from your experience of developing in C and Objective-C.

Swift provides its own versions of all fundamental C and Objective-C types, including `Int` for integers, `Double` and `Float` for floating-point values, `Bool` for Boolean values, and `String` for textual data. Swift also provides powerful versions of the three primary collection types, `Array`, `Set`, and `Dictionary`, as described in Collection Types.

Like C, Swift uses variables to store and refer to values by an identifying name. Swift also makes 광범위한 extensive use of variables whose values can't be changed. These are known as constants, and are much more powerful than constants in C. Constants are used throughout Swift to make code safer and clearer 의도적으로 in intent when you work with values that don't need to change.

In addition to familiar types, Swift introduces advanced types not found in Objective-C, such as tuples. Tuples enable you to create and pass around groupings of values. You can use a tuple to return multiple values from a function as a single compound value.

Swift also introduces optional types, which handle the 부재 absence of a value. Optionals say either "there *is* a value, and it equals *x*" or "there *isn't* a value 조금도, 조금도 ~아니다 at all". Using optionals is similar to using `nil` with pointers in Objective-C, but they work for any type, not just classes. Not only are optionals safer and more 표현이 풍부한 expressive than `nil` pointers in Objective-C, they're at the heart of many of Swift's most powerful features.

Swift is a *type-safe* language, which means the language helps you to be clear about the types of values your code can work with. If part of your code requires a `String`, type safety prevents you from passing it an `Int` by mistake. Likewise, type safety prevents you from accidentally passing an optional `String` to a piece of code that requires a non-optional `String`. Type safety helps you catch and fix errors as early as possible in the development process.