Core+ with Swift

Group Members: Avi Patel, Ayush Patel, Casey Williams, Ethan Krasznai

Introduction - Motivation:

We have created a fitness app that allows users to view, create and manage their workout plan for the week. We have all been going to the gym consistently for a couple of years and want to challenge our knowledge while encouraging others to enjoy a hobby we all share.

Language Specifications:

Arrows → are used to mark grammar productions and stand for 'can consist of'. Boldface constant width text shows literals and punctuation and appears only on the right-hand side of a grammar production rule. Subscripts identify syntactic categories and literals.

> GRAMMAR OF A GETTER-SETTER BLOCK
>
> *getter-setter-block* → **{** getter-clause setter-clause$_{opt}$ **}** | **{** setter-clause getter-clause **}**

> GRAMMAR OF A GETTER-SETTER BLOCK
>
> *getter-setter-block* → **{** getter-clause setter-clause$_{opt}$ **}**
> *getter-setter-block* → **{** setter-clause getter-clause **}**

Paradigm:

Swift has allowed programmers to jump barriers in iOS and OSX development. Its unique syntax and language limitations allow programmers to jump through various obstacles. It has very similar similarities and readability to C++ and Python. Swift is an object-oriented

programming language that is used in the majority of the current industry. Swift is a functional programming language which means that its syntax has upgraded, compared to C's block syntax style. Swift was designed with "lambda calculus" which means inputs are sent to functions that produce outputs. Swift is also a generic programming language which means it is trying to overcome the "shortcoming" of object oriented programming. The use of generic programming allows for generic functions to work at compile time, but do not get ordered by the function definition. The special thing about generic types is that they can use the same code that was used previously. Swift is good to use for programmers because it is an open-source language and it is type-safe. Swift has also introduced a protocol-oriented programming style. First, there are many base classes; a couple examples are BaseViewController, BaseTableViewController, and BaseCollectionViewController. Second, there is code duplication which means that the logic would be created in the same way for all of the classes. If there was more code this would lead to more bugs in the program. If we compare this to C we would have to use subclasses of the UIViewController which would make base classes and this would lead to bugs. Swift has changed this and created pre-existing classes so that the protocols can be accessed via add-ons. Swift is interesting because it can be object-oriented but it can also be functional and generic.

History of Swift:

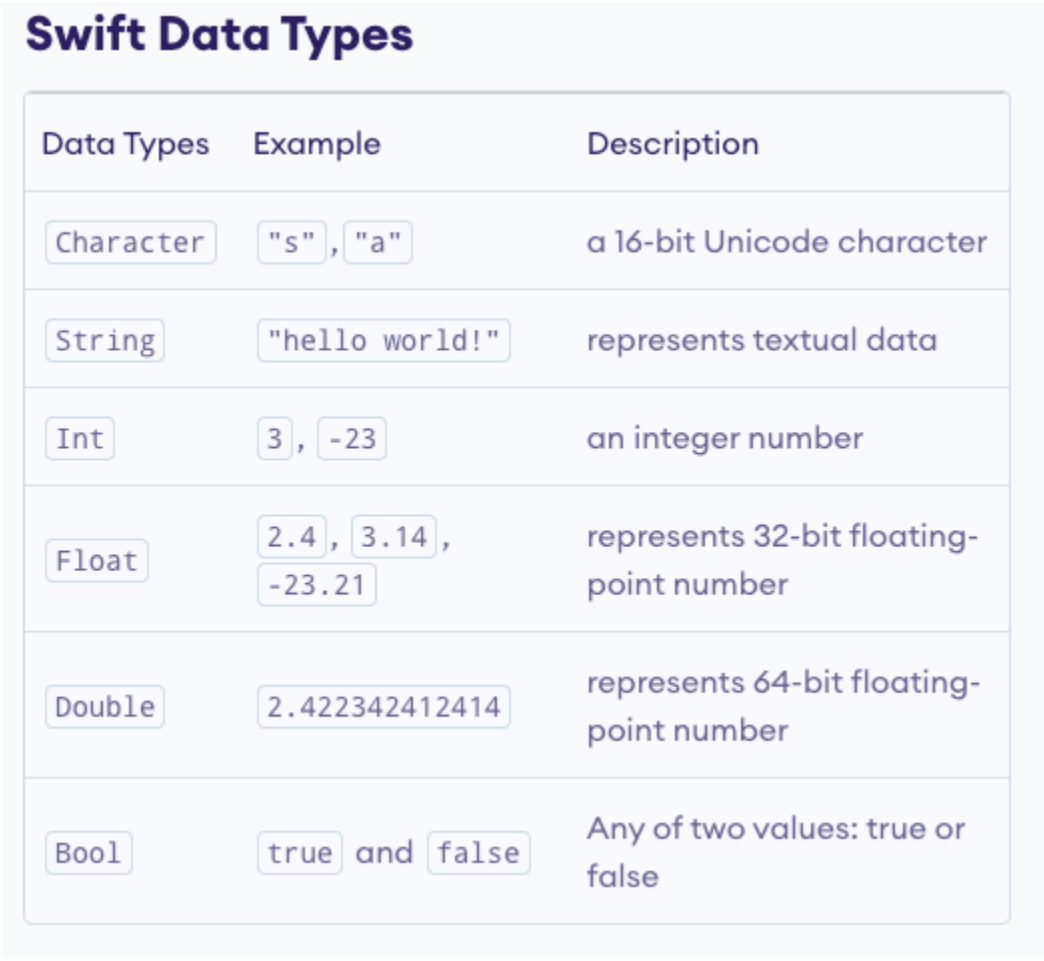
Swift was started in 2010 by Chris Lattner. He was the creator of LLVM and Clang. He was experimenting with a C-like compiler and wanted to create a better developing tool which we call Swift now. With the help of the head of the Apple software team, Bertrand Serlet, they started creating an alternative to Objective-C programming called Shiny. Years later, Lattner collaborated with other members of the team and continued the project. More features were added like ARC to the Objective-C programming and changed the name from Shiny to Swift. They wanted to make sure one of the main features was Swift's memory safety. Unlike C, where

it takes memory management, Swift has completely changed the way memory functionality works. Swift was formally introduced at the Apple WWDC as “Objective-C without C”, but this was not open source. Later on December 3, 2015, Swift was released as version 2.2 and made open-source under the Apache License 2.0.

The elements of Swift (reserved words, primitive data types, structured types):

Reserved words: In swift “identifiers that begin with two underscores are reserved for the standard library and Swift compiler. To actually use the reserved words as an identifier we have to use a backtick (`) before and after it.”

Primitive data types:



Structured type:

Array:

Dictionary:

Set:

Syntax of Swift:

Swift utilizes a “clean” syntax that makes it easier to read. There is no need for semicolons. You can also use emojis in strings which is interesting because it is an iOS/iOSX programming language. Swift has made it easy to define your variables/intent. Var is used to define a variable, let is used to define a constant. With the use of var and let it will allow for less errors in the code.

Basic control abstraction of Swift:

In swift the programmer will make their own control flow constructs. The use of user defined functions and syntactic sugar . An example would be *Test = Test + 1.* Control abstraction can be used to do the same thing by *Test += 1. Swift does not support Test ++/–. In for loops the language doesn't require you to iterate and make a variable a number, and increment for the amount of times you want the for loop to run. Instead you can do the following.*

```
let names = ["Anna", "Bob", "Jace"]
for name in names {
    print(name)
}
```

How the language handles abstraction:

Being an object-oriented language, swift uses abstract data types to model and manipulate real-world objects. Models contain primitive data types which represent attributes of the object. Objects can be constructed as classes, structures, or enumerations. A class' attributes stay the same, but the properties can change. For example, while creating our fitness app we needed to show the user exercises. These exercises will require equipment, work a specific body part, and

An evaluation of the language's writability, readability, and reliability:

Writability: How easy it is to write a program. Swift allows for programmers to write a whole program a lot faster compared to other languages because it requires less text to write the same program. This leads to faster development time because more work can be done compared to other programming languages.

Readability: How easy it is to read a program. Swift has safe and clean syntax which prevents errors. Swift is close to the natural English language which makes it easier to read. Swift has self-documenting code which makes it easier for developers to understand the code.

Reliability: How a program follows with its specifications, under a variety of conditions. The reliability of swift was a major design component. Swift was designed to specifically be safer than C-based languages, aiming to create a more streamlined environment with fewer runtime errors.

Strengths and weaknesses:

Strengths:

Swift has a simplified syntax and grammar which makes it easier to read, write, and understand the programming language. It does not require lots of code which requires less effort to write it manually. Swift projects are also easier to "scale which means faster development time". Swift is very close to natural English which makes it easier for other team members to

read previous written code. Swift has a 40% increase in performance compared to Objective-C programming language. Similarly claims from Apple state that search algorithms in swift require significantly less run time than C or Python. Swift has static and dynamic memory libraries. Swift uses ARC which makes automatic memory management, which speeds up the development process. Swift is an open-source programming language and easy to learn. It has been “ranked in the top 5 most starred languages on Github, TypeScript, and Rust”.

Weaknesses:

It is a fairly new language therefore swift has many updates and changes. Developers fear that it will not be compatible with previous versions, and if it can be compiled at all. Also due to frequent updates it may be hard to find certain tools for specific tasks. It is known to have lagging issues with specifically tooling and support with official Apple IDE and Xcode. Swift was initially made for iOS development, but now supports Apple Platforms, as well as linux, and windows. This means cross-platform does not work as well as Apple Platforms. Another downside is that Swift cannot be used in older iOS versions. Projects can only be worked on if the operating system is operating on iOS7 or higher.

Program Overview:

Core+ is a simplistic fitness app with no ads that allows users to read about exercises and create a personalized workout plan. It features 6 different pages - the home page, the list of exercises, exercise descriptions, a list of workout plans based on muscle group, a view to manage personal workouts, and a page to add workouts.

Sample Run:

A user downloads our app from the App Store and opens it. The landing page opens and they see a button to see their workout plans, or to view one of 5 muscle groups - legs, chest, back, shoulders, and arms. They click on the legs category and are taken to a list of exercises.

One of these is the leg press, and the user clicks on it. The user sees an animated picture of someone using a leg press, followed by the title, type of workout, and a description of how to perform it.

After viewing a few different exercises the user decides to create a workout plan. They travel back to the landing page and view the list of workout plans. They select their custom leg plan and see it is empty. They click the “add” button in the top right and add the exercises they liked. They are able to move the exercises around and delete one they added by accident. As they begin the workout, they keep track of what they completed by checking the indicators next to the name of each exercise. The next day they do the same thing with the next muscle group.

Problem definition:

The problem we are trying to solve is to allow a user to easily find workouts of a specific muscle group and provide an educational description of said workout. Similarly we want to allow users to then create their own customized workouts based upon the list of workouts we provided to the user. Overall we want to a create user friendly, simple application that helps beginner workout enthusiasts on their exercise journey.

Use Cases:

- The first use case of our application is to allow users to view our workouts as well as the workout descriptions.
- The second use case is to allow users to add workouts to their specific muscle group workout plans.
- The third use case is to allow users to remove specific workouts from their specific muscle group workout plans using a swipe to the left gesture on the selected workout.
- The fourth use case is to allow users to enter edit mode within their specific muscle group workout plans. Edit mode allows users to do two things.

    - The first use of edit mode allows users to delete selected workouts from their specific muscle group workout plans. (This use case is somewhat redundant, however, it allows users to delete items without a swipe gesture)
    - The second use of edit mode allows users to rearrange the order of specific workouts in a specific muscle group workout plan.
- The fifth use case is to allow users to checkmark a specific workout in a specific muscle group workout plan. When a workout is selected with an on tap gesture, the red circle displayed will turn into a green circle with a green checkmark.

Intuition:

Our program has features that would help a beginner/intermediate gym attendant to perform better at the gym.

Description of its Algorithms:

Users are allowed to view exercises based on muscle group and see an example and description of each. Users are allowed to create 5 custom workout plans and manage them by adding, deleting, moving, and marking them complete.

Experiments and Observations:

There were several swift implementations we experimented with. One of our primary obstacles was creating an array or list to store user-selected workouts, and have that array's data be consistent through views, and persist when the application is closed. We first experimented with firebase dictionaries, but we were unable to make them work. Secondly, we tried to use a JSON database, however, we only ran into issues with conformability to Hashable, and Equitable protocols. Finally, we were able to effectively make use of a UserDefault database, using JSONEncoding and JSONDecoding. Although we failed to accurately implement a firebase database or a JSON database it is likely that our failure to implement was more our fault than the IDE or the language. Our eventual success using a UserDefault

database was similar to the implementations of firebase or JSON databases, however, the UserDefault database worked as intended and there was no reason to change back.

Testing:

All exception handling was coded into our CRUD (Create, Read, Update, and Delete) functions using guard, and let try? blocks. These allow us to accurately predict the behavior of our functions, handle unwanted behavior, and avoid errors during execution or simulation.

Conclusions:

Swift is an efficient tool for building mobile applications. We noticed it is similar to Java and Python because it is object-oriented, but in functionality, it shares similarities with web UI languages like CSS and Javascript. The language is easy to use and understand, but at times we struggled because it is harder to find documentation for newer versions compared to other languages we have used.

Sources:

https://www.altexsoft.com/blog/engineering/the-good-and-the-bad-of-swift-programming-language/

https://medium.com/swift-classroom/apples-swift-a-guide-to-data-types-abstraction-and-expressions-382c66af5560

https://www.method.com/insights/swift-blurs-the-lines-of-programming-paradigms/

https://exyte.com/blog/introduciton-to-swift

https://docs.swift.org/swift-book/ReferenceManual/LexicalStructure.html#:~:text=Keywords%20reserved%20in%20particular%20contexts,unowned%20%2C%20weak%20%2C%20and%20willSet%20.

https://www.programiz.com/swift-programming/data-types

https://docs.swift.org/swift-book/LanguageGuide/ClassesAndStructures.html#:~:text=In%20fact%2C%20all%20of%20the,are%20value%20types%20in%20Swift.

https://www.tutorialspoint.com/swift/swift_structures.htm

https://developer.apple.com/swift/

https://www.cs.cornell.edu/jif/swift/doc/

https://www.altexsoft.com/blog/engineering/the-good-and-the-bad-of-swift-programming-language/

https://medium.com/@thisisfordeveloper/programming-language-evaluation-criteria-part-2-writability-reliability-cost-6c3029c9d957

https://www.coursera.org/articles/programming-in-swift

https://docs.swift.org/swift-book/ReferenceManual/AboutTheLanguageReference.html