

Estruturas de Linguagens

SWIFT

Prof. Francisco Sant`Anna

Aluno: Felipe Mello Diniz

Introdução

- Swift é uma linguagem de programação compilada criada pela Apple para o desenvolvimento de programas, e aplicativos, para iOS, macOS, watchOS, tvOS e Linux. Ela foi projetada para trabalhar com os frameworks da Apple (Cocoa e Cocoa Touch), e com códigos em Objective-C.

Origens e Influencia

- O projeto para a criação dessa linguagem de programação foi liderado pelo programador Chris Lattner, que contou com a eventual ajuda de alguns dos desenvolvedores da Apple. Esse projeto teve início em 2010, e a primeira versão da linguagem foi apresentada em 2014, na WWDC desse mesmo ano, e teve versões novas apresentadas em 2015 e em 2016. Swift foi desenvolvido, principalmente, como uma alternativa ao Objective-C, porém com uma escrita mais simples, e com maior expressividade, além de outras melhorias.

Origens e Influencia

- Swift foi influenciada por várias linguagens, como: C, C#, Objective-C, D, Haskell, Python, Ruby e Rust. Para compreender melhor como se deram essas influências segue a linha do tempo:
 - 1972 – C
 - 1983 – Objective-C
 - 1990 – Haskell
 - 1991 – Python
 - 1995 – Ruby
 - 2000 – C#
 - 2001 – D
 - 2010 – Rust
- E, apesar de ser uma linguagem de programação nova, ela já influenciou algumas linguagens, como Ruby – onde inspirou o “safe navigation operator” (Operador de navegação segura) – e Rust – onde inspirou o construtor “if let”.

Classificação

- Swift é classificada como uma linguagem multi-paradigma, sendo:
 - Imperativa
 - Funcional
 - Orientada a Objetos
- Quanto a sua tipagem, ela é considerada:
 - Estática
 - Forte
 - Inferida

Avaliação Comparativa

Readability:

- Para comparar a leitura de duas linguagens, precisa-se de um código como exemplo, pois essa comparação depende muito do que o programa deseja fazer. No exemplo escolhido conclui-se que Swift é uma linguagem melhor de se ler do que a linguagem C. Por possuir um recurso que C não possui, Swift permite deixar o código mais simples, o que facilita a leitura do mesmo.

Avaliação Comparativa

Writeability:

- Ainda usando a linguagem C como comparação, podemos dizer que Swift é uma linguagem boa para escrever. Inclusive, esse é um dos atributos que os desenvolvedores da Apple levaram em consideração para a criação da linguagem. A escrita dessa linguagem é mais intuitiva, e de fácil desenvolvimento, como em Python – similaridades na parte de orientação a objetos. Outro motivo para chegar a essa conclusão é a tipagem – forte e inferida – que contribui para que o código seja mais enxuto e mais fácil de ser escrito.
- “Swift is designed to make writing and maintaining *correct* programs easier for the developer.” (Trecho do texto “About Swift” retirado do site oficial da linguagem).

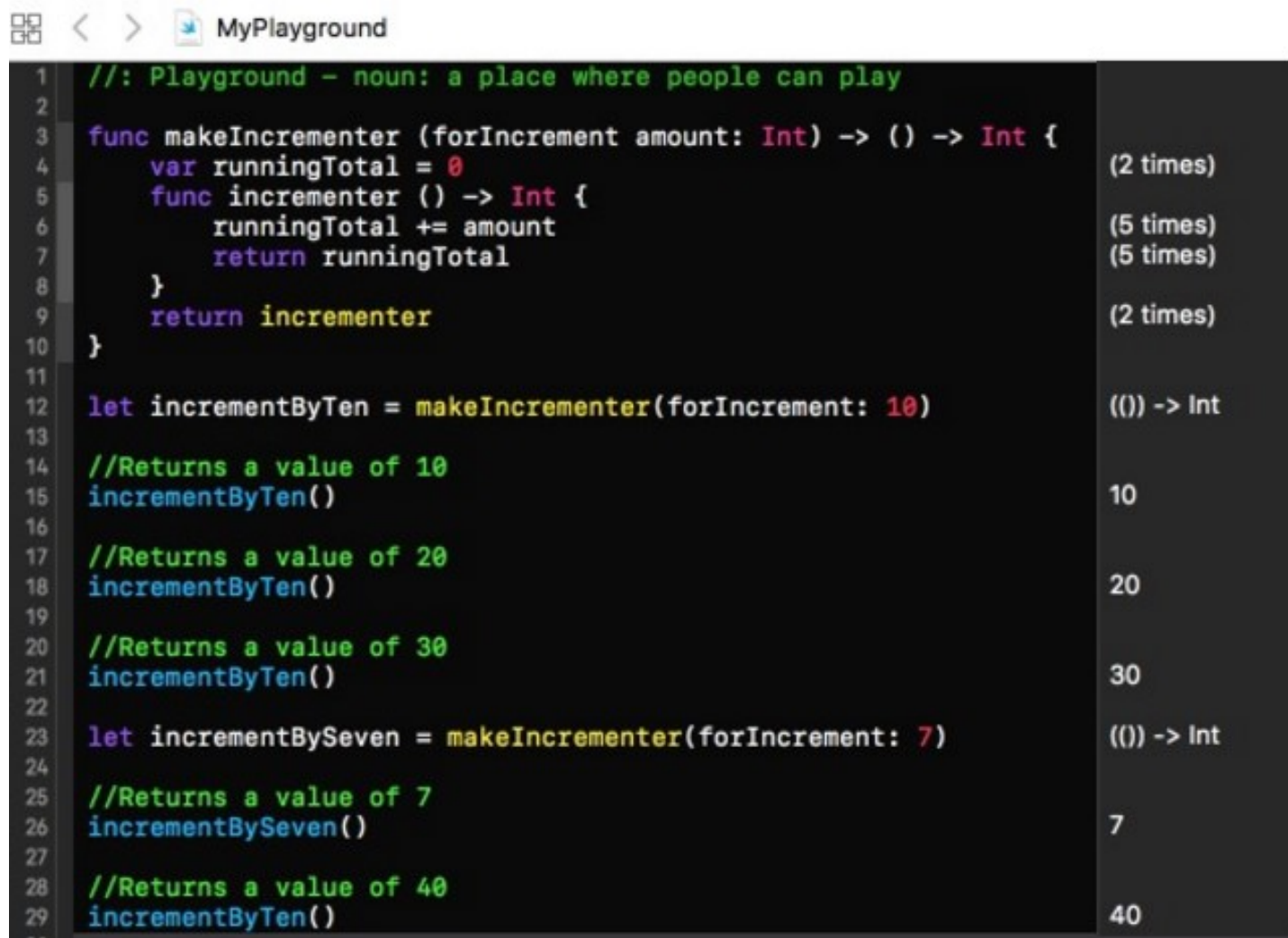
Avaliação Comparativa

Expressiveness:

- Por possuir muitos recursos – Closures, Orientação a objetos, padrões de programação funcional, entre outros -, Swift pode ser considerada mais expressiva do que muitas outras linguagens. Continuando a comparação dela com a linguagem C, conseguimos concluir que Swift tem maior poder de expressão do que C usando o recurso de Closures como exemplo. Por possuir esse recurso, Swift mantém o código limpo, enxuto e com fácil compreensão, enquanto em C, para reproduzirmos esta mesma função, necessita-se de um código consideravelmente mais extenso, e de difícil compreensão.

Exemplificando

Swift:



```
1  //: Playground - noun: a place where people can play
2
3  func makeIncrementer (forIncrement amount: Int) -> () -> Int {
4      var runningTotal = 0
5      func incrementer () -> Int {
6          runningTotal += amount
7          return runningTotal
8      }
9      return incrementer
10 }
11
12 let incrementByTen = makeIncrementer(forIncrement: 10)
13
14 //Returns a value of 10
15 incrementByTen()
16
17 //Returns a value of 20
18 incrementByTen()
19
20 //Returns a value of 30
21 incrementByTen()
22
23 let incrementBySeven = makeIncrementer(forIncrement: 7)
24
25 //Returns a value of 7
26 incrementBySeven()
27
28 //Returns a value of 40
29 incrementByTen()
```

(2 times)
(5 times)
(5 times)
(2 times)
(()) -> Int
10
20
30
(()) -> Int
7
40

Exemplificando

C:

```
#include <callback.h>
#include <stdio.h>
static void incrementer_(int *in) {
    ++*in;
}
static void emitter_(int *in) {
    printf("%d\n", *in);
}
int main() {
    int in1 = 10, in2 = 20;
    int (*incrementer1)() = alloc_callback(&incrememnter_, &in1);
    int (*emitter1)() = alloc_callback(&emitter_, &in1);
    int (*incrementer2)() = alloc_callback(&incrememnter_, &in2);
    int (*emitter2)() = alloc_callback(&emitter_, &in2);
    incrementer1();
    incrementer2();
    emitter1();
    emitter2();
    free_callback(incrementer1);
    free_callback(incrementer2);
    free_callback(emitter1);
    free_callback(emitter2);
}
```

Conclusão

A escolha desses códigos ajuda a exemplificar o que foi dito na comparação entre as linguagens. Neles é possível ver que o código escrito em Swift – usando Closure – é consideravelmente mais enxuto, fácil de escrever, e de entender. Não há nada que atrapalhe o entendimento do código. Já em C, observa-se um código maior, mais complicado de escrever – pois precisa de um bom conhecimento da linguagem (uso de ponteiros) – e mais difícil de compreender o que está sendo feito.

Após essa análise, pode-se concluir que em alguns casos, quando uma linguagem possui um poder maior de expressão, ela se torna uma linguagem mais fácil de ler (o que auxilia na manutenção do código) e mais fácil de escrever (o que auxilia na prototipação).

Bibliografia

- Wikipedia PT:
[https://pt.wikipedia.org/wiki/Swift \(linguagem de programa%C3%A7%C3%A3o\)](https://pt.wikipedia.org/wiki/Swift_(linguagem_de_programa%C3%A7%C3%A3o))
- Wikipedia EN:
[https://en.wikipedia.org/wiki/Swift \(programming language\)#Comparisons to other languages](https://en.wikipedia.org/wiki/Swift_(programming_language)#Comparisons_to_other_languages)
- Swift (site oficial):
<https://swift.org/about/>
- Stackoverflow:
<http://stackoverflow.com/questions/4393716/is-there-a-a-way-to-achieve-closures-in-c>
- Developer Apple:
[https://developer.apple.com/library/content/documentation/Swift/Conceptual/Swift Programming Language/Closures.html](https://developer.apple.com/library/content/documentation/Swift/Conceptual/Swift_Programming_Language/Closures.html)