

# 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 Influências

- 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 Influências

- 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 func makeIterator(from start: Int, step: Int) -> () -> Int {  
2     var i = start  
3     func iter() -> Int {  
4         let currentValue = i  
5         i += step  
6         return currentValue  
7     }  
8     return iter  
9 }  
10  
11 var iterator = makeIterator(from:1, step:1)  
12 var anotherIterator = makeIterator(from:1, step:3)  
13  
14 /*Saidas: 1, 1, 2, 4, 3, 7, 4, 10*/  
15  
16 iterator()  
17 anotherIterator()  
18  
19 iterator()  
20 anotherIterator()  
21  
22 iterator()  
23 anotherIterator()  
24  
25 iterator()  
26 anotherIterator()
```



```
Swift version 3.0.1 (Release)  
iterator: () -> Int = 0x00007ffff7f6e1a0  
$__lldb_expr2`swift_rt_swift_release + 16  
anotherIterator: () -> Int = 0x00007ffff7f6e1a0  
$__lldb_expr2`swift_rt_swift_release + 16  
$R0: Int = 1  
$R1: Int = 1  
=> $R2: Int = 2  
$R3: Int = 4  
$R4: Int = 3  
$R5: Int = 7  
$R6: Int = 4  
=> $R7: Int = 10  
❏
```

# Exemplificando

C:

```
1  #include "callback.h"
2  #include <stdio.h>
3
4  static void step(int * in, int st){
5      *in += st;
6  }
7
8  static void emitter(int * in){
9      printf("%d\n", *in);
10 }
11
12 int main(){
13     int in1 = 10, in2 = 7;
14     int (* step1) () = alloc_callback(&step, &in1, st);
15     int (* emitter1) () = alloc_callback(&step, &in1);
16     int (* step2) () = alloc_callback(&step, &in2, st);
17     int (* emitter2) () = alloc_callback(&step, &in2);
18     step1();
19     step2();
20     emitter1();
21     emitter2();
22     free_callback(step1);
23     free_callback(step2);
24     free_callback(emitter1);
25     free_callback(emitter2);
26 }
```

# 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)