Material didático referente à avaliação 1

Swift

Instituto Federal de Educação, Ciência e Tecnologia do Ceará (IFCE)

Maracanaú - CE – Brasil

matheus.mota.freitas61@aluno.ifce.edu.br

Abstract. *This material aims to complement the proposed video for Evaluation 1 of Programming Fundamentals, addressing some characteristics of the Swift programming language, as well as a little of its development history, comparing it with other languages and demonstrating the positive and negative points it has.*

Resumo. *Este material visa complementar o vídeo proposto para a Avaliação 1 de Fundamentos de Programação, abordando algumas características da linguagem de programação Swift, bem como um pouco da história do seu desenvolvimento, comparando-a com outras linguagens e demontrando os pontos positivos e negatos que ela possui.*

1. Introdução

Swift é uma linguagem para devenvimento nativo de sistemas da Apple. Ela teve seu desenvolvimento iniciado em julho de 2010 por Chris Lattner junto com a participação de outros programadores da Apple, tendo sido anunciada em 2014 na "Worldwide Developers Conference (WWDC)", a conferência anual da empresa para desenvolvedores. A sua versão 1.0 foi lançada em setembro de 2014 em conjunto com a versão 6.1 do Xcode, sempre vindo embutida nele desde então. Em 2017 o Swift ficou entre as 10 linguagens mais populares do mundo de acordo com o Índice Tiobe.

O Swift tornou-se open source em dezembro de 2015, e com isso foi desenvolvido o Swift.org com o intuito de hospedar esse projeto, tendo seu Código fonte disponivel para todos no GitHub. Além disso, durante a WWDC de 2016 houve o lançamento do Swift playgrounds que possui a finalidade de ensinar a linguagem de programação de forma interativa.

Tabela 1. Histórico de versões

Data de lançamento	Versão
09/09/2014	Swift 1.0
22/10/2014	Swift 1.1
08/04/2015	Swift 1.2
21/09/2015	Swift 2.0
13/09/2016	Swift 3.0

19/09/2017	Swift 4.0
29/03/2018	Swift 4.1
17/09/2018	Swift 4.2
25/03/2019	Swift 5.0

2. Caracteristicas da linguagem

O Swift possui diversas caracteristicas que o tornam bastante atraente para os desenvolvedores , dentre as quais podemos citar:

2.1. Sintaxe simples

A escrita da linguagem desenvolvida pela Apple foi feita pensando em plicidade para torná-la mais prática, como por exemplo o fato de o ponto sime vírgula ser algo opcional.

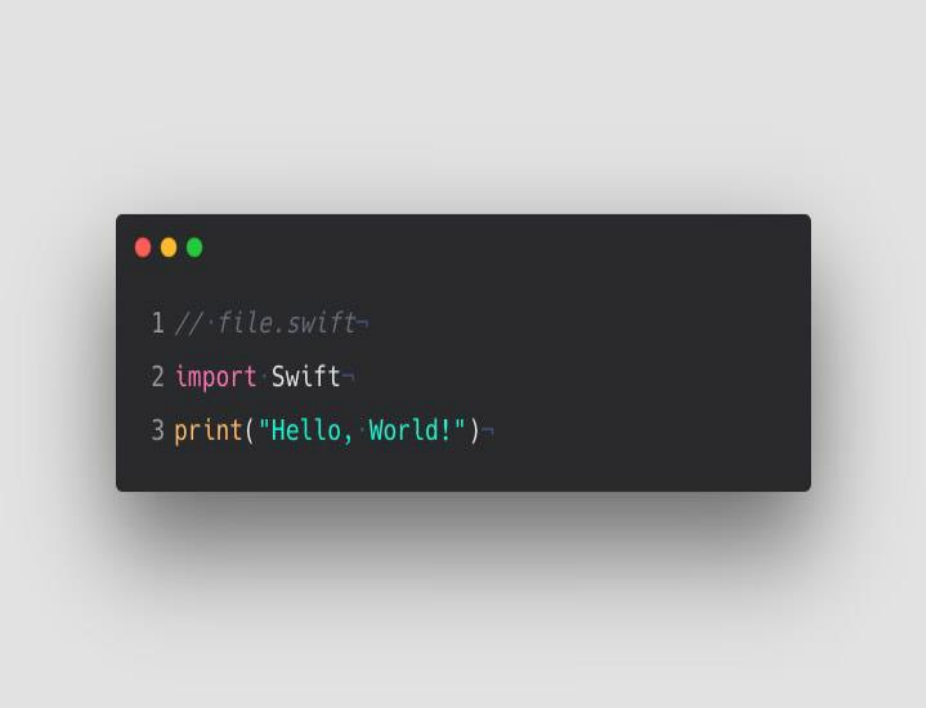

Figura 1. “Hello World” em swift

2.2. Ter Tupla

Um espaço de tuplas tem como função criar uma abstração de memória compartilhada sobre um sistema distribuído, onde todos podem ler e escrever no mesmo.

```
var currency = ("EUR", 0.81)
var time = (NSDate(), "This is my message.")
var email = ("Bart Jacobs", "bart@example.com")
```

Figura 2. O primeiro exemplo declara uma tupla com o nome de currency do tipo (String, Int). A segunda tupla, time, contem uma instância NSDate e uma string. Os valores armazenados em email são ambos do tipo String, o que significa que email é do tipo (String, String).

3. Declaração de variáveis

Uma variável é um espaço em memória que pode ser alterado no decorrer do programa. Uma constante não pode ser acedida para escrita.

Para se declarar uma variável em Swift deve-se utilizar a keyword var. O tipo de variável é definido de forma implícita de acordo com o valor definido, mas também pode,os defini-la de forma explicita. Além disso pode-se declarar diversas variáveis simultaneamente utilizando”,”.

```
var nome = "Márcio Fábio Althmann"
var idade = 30

var anoDeNascimento: Int
anoDeNascimento = 1983

var primeiroNome: String
primeiroNome = "Márcio"

var dia = 8, mes = 10, ano = 1983
```

Figura 3. Declaração de variáveis de forma implícita, explicita e simultânea

4. Vantagens e desvantagens com o objective C

4.1. Vantagens

- O swift é mais fácil de aprender para novos programadores.

- Suporte para namespaces, uma sintaxe de mutabilidade clara, padrões funcionais e sintaxe concisa.
- O Swift é mais seguro devido à digitação estática e ao uso de opcionais e encadeamento opcional.
- Desenvolvimento interativo usando Playgrounds.
- A visualização do SwiftUI está disponível diretamente com o Xcode sem executar o projeto no simulador (dando feedback visual instantâneo). Além disso, o dispositivo de visualização pode ser alternado rapidamente adicionando um modificador para um dispositivo de visualização, sem o incômodo de construir e executar o projeto em um simulador diferente.

```
# se DEBUG
struct  ContentView_Previews : PreviewProvider {
     visualizações de var estáticas : alguns Exibir {
        ContentView (). previewDevice ( " iPhone SE " )
    }
}
# endif
```

Figura 4. Linha de código em Swift

- A alocação de memória tornou-se mais eficiente para eliminar algumas alocações quando o objeto nunca é usado fora do local de escopo. O compilador pode alocar um objeto na stack onde o acesso é muito mais rápido, também pode optar por não alocar caso ele nunca seja usado.

```
1  for(int i = 0; i < 1000000; i++)
2
3    [[[NSObject alloc] init] self
4    ];
```

Figura 5. A alocação desse exemplo em Objective-C seria executada um milhão de vezes mandando milhares de mensagens para construir e destruir o objeto. Mas passando esse código para Swift com o novo compilador, ele não executaria nada se percebesse que self nunca é usado e é apenas referenciado dentro do loop podendo também retirar o código no processo de compilação.

4.2. Desvantagens

- Maior tempo de compilação.
- Nenhuma maneira direta de usar bibliotecas C ++.
- A estabilidade do formato do módulo ainda não foi alcançada e é necessária para desenvolvedores que desejam compartilhar seu código como uma estrutura binária.

5. Gerenciador de pacotes

O Swift Package Manager é uma ferramenta para gerenciar a distribuição de código Swift. Ele é integrado ao sistema de construção Swift para automatizar o processo de download, compilação e vinculação de dependências.

5.1. Pacotes

Um pacote consiste em arquivos de origem Swift e um arquivo de manifesto. O arquivo de manifesto, chamado Package.swift, define o nome do pacote e seu conteúdo usando o PackageDescriptionmódulo.

Um pacote possui um ou mais destinos. Cada destino especifica um produto e pode declarar uma ou mais dependências.

```
19  19      "repositoryURL": "https://github.com/vapor/console.git",
20  20      "version": "1.0.2"
21  21    },
22  22    {
23  23      "package": "Core",
24  24      "reason": null,
25  25      "repositoryURL": "https://github.com/vapor/core.git",
-26         "version": "1.1.1"
    +26     "version": "1.1.2"
27  27    },
28  28    {
29  29      "package": "Crypto",
30  30      "reason": null,
31  31      "repositoryURL": "https://github.com/vapor/crypto.git",
32  32      "version": "1.1.0"
33  33    },
```

Figura 6. Exemplo de package no Swift.

6. Referências

Petr Pavlík(2017), https://theswiftwebdeveloper.com/swift-package-pinning-4fc7ab769a51.

Lucas Longo(2014), https://imasters.com.br/back-end/swift-linguagem-que-aproxima-o-mundo-da-programacao.

Henrique Marques Fernandes(2020), https://marquesfernandes.com/tecnologia/o-que-e-a-linguagem-swift-e-para-que-serve-desenvolvendo-aplicativos-ios/

Gustavo Machado(2015), https://www.infoq.com/br/articles/apple-swift/

Mike Wuerthele(2017) , https://appleinsider.com/articles/17/01/13/new-swift-project-head-ted-kremenek-said-to-be-running-the-show-behind-the-scenes-for-some-time

Rachel Metz(2014), https://www.technologyreview.com/2014/06/03/250717/apple-seeks-a-swift-way-to-lure-more-developers/

Hrrison Weber(2014), https://venturebeat.com/2014/06/02/apple-introduces-a-new-programming-language-swift-objective-c-without-the-c/

Márxio Althmann(2014), https://www.marcioalthmann.net/2014/08/swift-introducao-constantes-e-variaveis/

Erbi Silva(2016), https://www.techemportugues.com/2016/01/07/programacao-swift-variaveis/

Código Fonte Tv(2019), https://youtu.be/ELa-PgWlYDI

Apple Inc(2020), https://swift.org/