Leitura I: Introdução à Swift

Observação

Este material foi traduzido e adaptado do curso CS 193P - *iPhone Application Development* oferecido pela Universidade de Stanford através do iTunesU. Ele está protegido pela licença *Creative Commons Attribution-Noncommercial-Share*[1].

Objetivo

O objetivo desta primeira semana de leitura é iniciar os estudos nesta nova linguagem denominada Swift. Cobriremos o básico como variáveis e fluxo de controle, mas também um pouco de um tópico mais não convencional: fechamentos (funções como tipos).

Muitos de vocês não tiveram experiência com Objective-C, mas não se preocupe com isso. Nada na documentação de Swift realmente assume isso. Contudo, se você nunca programou em C (C#, Java, C++ ou qualquer outra variante), então Swift pode ser extremamente nova para você (mas tenho a esperança de que mesmo assim você aprenderá).

Materiais Necessários

Toda a leitura deve ser feita no documento *Swift Programming Language* (Linguagem de Programação Swift). Este documento está disponível em:
https://developer.apple.com/library/mac/documentation/Swift/Conceptual/Swift_Programming_Language/index.html

Seções a Ler

Para que o seu tempo seja utilizado de forma mais útil e para enfatizar os conceitos mais importantes neste momento, as seções foram divididas em quatro categorias:

- **vermelhas** são seções MUITO IMPORTANTES e devem ser mais difíceis para serem

1 http://creativecommons.org/licenses/by-nc-sa/3.0/us/

entendidas. Elas devem ser lidas com bastante atenção;
- **azuis** são seções importantes, mas não tão difíceis de serem entendidas;
- **verdes** são seções importantes, mas cobrem assuntos bastantes básicos e simples (muitos deles são parecidos com os encontrados na linguagem C);
- **cinzas** são seções que não precisam ser lidas neste momento. Provavelmente serão nas próximas semanas.

Não deixe de ler os textos que estão dentro dos retângulos acinzentados (*NOTES*), pois muitos deles são importantes.

Se há uma ligação a outra seção no texto, você não precisa seguir a ligação a não ser que também seja parte desta tarefa de leitura.

Na seção *Language Guide*, leia as seguintes seções dos seguintes capítulos:

The Basics

Constants and Variables
Comments
Semicolons
Integers
Floating-Point Numbers
Type Safety and Type Inference
Numeric Literals
Numeric Type Conversion
Type Aliases
Booleans
Tuples
Optionals
Assertions

Os nomes de constantes e variáveis Unicode (por exemplo,) podem ser divertidos, mas você será responsabilizado pela qualidade dos nomes (em todos os locais) e legibilidade do seu código.

Não use pontos-e-vírgulas nos finais das linhas (somente os utiliza para, raramente, separar dois

comandos em uma única linha).

Basic Operators

Muito desta seção é quase idêntico à linguagem C (por isso muitas seções são **verdes**).

Terminology
Assignment Operator (ignore "references to tuples")
Arithmetic Operators
Compound Assignment Operators
Comparison Operators
Ternary Conditional Operator
Nil Coalescing Operator
Range Operators
Logical Operators

Strings and Characters

Ignore o conteúdo da caixa *NOTE* na introdução do capítulo. Discutiremos sobre as classes do Objective-C como *NSString* posteriormente.

String Literals
Initializing an Empty String (ignore "*initializer syntax*" por enquanto)
String Mutability
Strings Are Value Types (ignore *NOTE*)
Working with Characters
Concatenating Strings and Characters
String Interpolation
Unicode
Counting Characters
Comparing Strings
Unicode Representations of Strings

Uma *String* também pode ser convertida para um *Array <Character>* assim:
let myArrayOfCharacters = Array(myString)
Isso não é mencionado neste capítulo porque ele está antes do capítulo que descreve *Array*.

Collection Types

Mutability of Collections
Arrays (ignore a explicação sobre initializers no final da seção)
Dictionaries

Se você tentar acessar um índice maior do que a quantidade de elementos em um *array*, o programa será terminado com erro (*crash*). Há também uma função *last* em *Array* que retorna um *Optional* (i.e. ela retorna nil se, e somente se, o *Array isEmpty*). Note que o operador sobre *array* += leva um outro *array* no lado direito (não um elemento que será adicionado ao *array*).

Control Flow

For Loops (**for-in** pode ser novo para muitos de vocês, bem como **ranges**, por exemplo 1..5)
While Loops
Conditional Statements (especialmente **Switch**, ignore Tuples, Value Bindings & Where)
Control Transfer Statements (mas ignore Labeled Statements)

O comando *switch* é mais importante em Swift do que em C ou em outras linguagens

Functions

Defining and Calling Functions
Function Parameters and Return Values (ignore Functions with Multiple Return Values e Optional Tuple Return Types)
Function Parameter Names
Function Types (esta seção será o maior desafio para a maior parte de vocês)
Nested Functions

Closures

Em Swift, é muito importante entender que uma descrição de uma função (i.e. os argumentos e valor de retorno) pode ser considerado um tipo de primeira classe (como um *Array* ou um *Int*). Muitas API do iOS tem fechamentos (*closures*) como argumentos.

Closure Expressions (entenda **Function Types** antes)

Trailing Closures
Capturing Values
Closures are Reference Types

Classes and Structures

Comparing Classes and Structures
Structures and Enumerations are Value Types
Classes are Reference Types
Choosing Between Classes and Structures
Assignment and Copy Behavior for Strings, Arrays, and Dictionaries

Note que até nesta aula só conhecemos duas maneiras de criar uma classe ou *struct*: colocando o nome da classe ou struct seguido de parêntesis vazios, por exemplo *let x = VideoMode ()*, ou, apenas para *struct*, usando o nome da *struct* com todas as variáveis tendo um valor, por exemplo, *let hd = Resolution (width : 1920, height : 1080)*.

Aprenderemos tudo sobre meios mais poderosos de iniciar classes e *struct* nas próximas aulas.

Properties

Ignore referências a enumerações (*enum*). Enumerações são importantes e serão vistas nas próximas aulas com detalhes.

Stored Properties (ignore Lazy Stored Properties e Stored Properties and Instance Variables)
Computed Properties
Property Observers
Global and Local Variables
Type Properties