Nome: Gustavo Padilha, Natan Valim Cardoso e Rodrigo Raupp

Prof: Vinicius Silveira Magnus

***Resumo***. *O trabalho tem objetivo de mostrar todas as sintaxes do ios além de mostrar as grandes novidades que vem chegando no mercado.*

**PALAVRAS CHAVE:***IOS, novidades*

***Abstract.*** *The work aims to show all the ios syntaxes in addition to showing the great news that is coming to the market.*

**KEYWORDS:***IOS, news*

**1.INTRODUÇÃO**

Neste trabalho iremos abordar os temas focados no desenvolvimento iOS, sobre as linguagens de programação, evolução das tecnologias e novidades dessas aplicações.

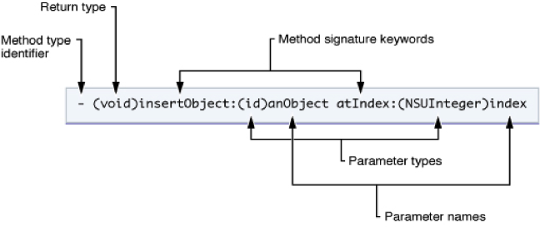
**1.1 QUAL É A LINGUAGEM DE PROGRAMAÇÃO DO IOS?**

Até 2014, a linguagem de programação usada para iOS era a Objective-C (criada no início dos anos 80 e mais tarde adquirida pela Apple) Ela continua sendo usada, só não é mais considerada a favorita para essa tarefa e por fim o appleScript que foi centrado na interface do sistema operacional da Apple.

**2.OBJECTIVE-C**

A linguagem Objective-C, nasceu na empresa NeXT, antiga empresa criada pelo Steve Jobs quando ele saiu da Apple. Quando em 1996 a Apple compra a empresa NeXT começa a desenvolver então na linguagem Objective-C.

Esta linguagem é orientada a objetos, é uma extensão à linguagem C e case-sensitive.Há dois tipos de ficheiros em objective-c: o ficheiro .h (interface); e o ficheiro source: o ficheiro .m.



O “Method type Identifier” é o sinal de “-” ou o sinal de “+”. O tipo de retorno da nossa função.

As keywords são como o nome diz as keywords que irão aparecer quando chamarmos o nosso método para seguido do sinal de “:” indicarmos qual o objecto a mandar por parâmetro.

O “Parameter type” é o tipo da variável esperada para a keyword e o “Parameter names” são os nomes depois que vão ser utilizados para manipularmos os dados quando formos programar no ficheiro .m o que vai fazer o nosso método, processo semelhante ao Java.

A Apple integrou seu framework de desenvolvimento , Cacau, com Objective C para a programação do sistema operacional Mac. O Objective C é útil para desenvolvedores da Apple , mas ele perde um pouco do seu encanto se um aplicativo precisa ser cross- platform. A linguagem utilizada na programação dos aplicativos para o iOS é a Objective-C, a qual é decodificada através de dados. A categoria de linguagem “C” direciona a decodificação através da manipulação de objetos. Os desenvolvedores que já possuem experiência e conhecimentos de Java conseguem trabalhar com a Objective-C facilmente. Vale ressaltar que sem conhecimentos da linguagem Objective-C a criação de funções avançadas não será possível.

Há várias comunidades online que trazem macetes e informações complementares do uso desta linguagem, como o Apple Developer Forums. Uma **vantagem** do Objective C é que é o idioma principal utilizado para desenvolver aplicativos para os produtos da Apple Objective C é a linguagem de programação desenvolvida especificamente para trabalhar com o quadro de programação de cacau , o que significa que ele tem acesso às bibliotecas de desenvolvimento da Apple. Criação de aplicativos para qualquer produto da Apple, como o iPod , iPad ou sistemas OS X , requer algum conhecimento de Objective C.

Objective C + + são Runtime Uma **desvantagem** da programação em Objective C é que ele não contém namespaces. Em outras línguas , em particular C + + , os espaços contêm funcionalidade definida por um nome . Funções em um espaço de nomes só existem no âmbito de aplicação desse namespace, o que significa que uma outra função com um nome idêntico em outro namespace pode ser usado pelo programador sem alterar quaisquer nomes. Isso permite que os programadores a importar uma variedade de bibliotecas e não se preocupar desnecessariamente sobre nomes de funções conflitantes .

Por outro lado cuidado deve ser redobrado,pois a importação de duas bibliotecas com o mesmo nome de função pode causar um erro se o programador não estiver cuidado, porque o compilador não tem como saber qual a função de realmente chamar.

## **3. O QUE É O SWIFT**

*“Swift é uma linguagem de programação robusta e intuitiva criada pela Apple para a construção de aplicativos para iOS, Mac, Apple TV e Apple Watch. Ela foi projetada para oferecer mais liberdade do que nunca aos desenvolvedores. Swift é fácil de usar e de código aberto, então qualquer pessoa com uma ideia pode criar algo incrível.”*

No passado recente, as aplicações móveis viraram o mundo do avesso e já mudaram a forma como usamos a internet para trabalho e lazer. Inúmeras tecnologias surgiram para criar aplicativos móveis e processos de desenvolvimento começaram a considerá-los como cidadãos de primeira-classe. Contudo, embora o Mobile já pareça ser onipresente, o futuro está prestes a começar. Estamos diante de novas gerações de dispositivos móveis de vestir (wearables) ou diversos gadgets que compõem a Internet das Coisas (Internet of Things, IoT). Seremos confrontados com novos tipos de interfaces de usuário para mostrar dados bem como aceitar comandos. E presenciaremos cada vez mais empresas realmente priorizando a mobilidade. Tudo isso influenciará a forma como projetamos, codificamos e testamos software nos anos vindouros.

A Apple já tinha uma linguagem de programação, o Objective-C. Então, por que introduzir uma nova linguagem? Bem, embora o Objective-C possa ter sido bem diferente e moderno quando lançado, ela não se encaixa ao gosto atual por linguagens. Do lado do varejo por exemplo, linguagens de script como o Ruby tem sido amplamente adotadas graças à sintaxe limpa que a linguagem oferece. No mundo corporativo, linguagens fortemente tipadas (type safe) com inferência de tipo parecem ser preferidas e grandes esforços têm sido vistos em linguagens como C# e Java (ou Scala) para trazer conceitos tipicamente encontrados em linguagens funcionais, tais como: funções como objetos e Lambdas. No Objective-C faltam algumas dessas coisas, como sintaxe limpa (e syntatic sugar) e inferência de tipo. O Swift tenta preencher essa lacuna.

Agora vamos focar naquilo que o Swift tem a oferecer. Do ponto de vista da linguagem, Swift é incrível. A Apple observou linguagens modernas, como Scala ou C# e construiu uma linguagem super simples e no entanto poderosa. Ele oferece uma mistura bem interessante entre paradigmas de orientação a objetos e programação funcional - embora dizer que Swift é uma linguagem funcional é um grande exagero. Vamos passar para as cinco melhores características do Swift.

Além disso, como o desenvolvimento da linguagem foi inspirado na Objective-C (além de outras linguagens mais modernas como a Python, Ruby, Rust e outras), ela foi criada para poder ser tranquilamente incorporada em códigos já desenvolvidos em Objective-C.Então, se você já tem apps nessa linguagem, não precisa sofrer nem ficar com medo. Elas conversam entre si numa boa.

Aliás, segundo a Stack Overflow, a Swift ficou em primeiro lugar na categoria Linguagem de Programação Mais Amada de acordo com uma pesquisa com desenvolvedores em 2015 ([*Stack Overflow Developer Survey 2015*](https://insights.stackoverflow.com/survey/2015)) e em segundo lugar em 2016 ([*Stack Overflow Developer Survey 2016 Results*](https://insights.stackoverflow.com/survey/2016)).Como muitos desenvolvedores reclamavam que a Objective-C (antes da criação da Swift) tem uma sintaxe complicada e difícil de debugar, esse é um dos problemas que a Swift veio resolver, pois ela é mais fácil e, segundo o site da própria Apple, “sua sintaxe é concisa mas expressiva”.

Então, além de ser mais fácil de aprender, ela é mais fácil de ensinar. Mas a Apple reforça a cada passo que essa simplicidade não diminui a o poder dessa linguagem. Como ela já existe há alguns anos, isso já pôde ser verificado e atestado pelos desenvolvedores.Ela inclui Closures e Generics também, cuja falta era usada como argumento para alguns programadores não migrarem para o ecossistema de desenvolvimento da Apple.É uma linguagem rápida, como o nome sugere, segura, escalável, com gerenciamento automático de memória e por ser muito fácil de ser lida, facilita também que novos desenvolvedores entrem no time com o projeto em andamento.

## **3.1 VANTAGENS DE SINTAXE**

Sintaticamente falando, Swift é soberbo. É uma linguagem extremamente simples e limpa com uma leitura incrível mesmo para os padrões atuais. Pode-se ver de cara que a simplicidade foi chave na hora de projetar a linguagem. Tome como exemplo o famoso ponto e vírgula ao final de cada declaração. A Apple decidiu torná-lo opcional. Embora isso não seja extremamente relevante, mostra o esforço que eles tiveram para deixar a sintaxe o mais limpa possível.

Outros exemplos são a interpretação de string e o suporte da linguagem para tratar arrays e loops.Esses são apenas alguns exemplos de como a linguagem Swift tem suporte para fazer essas coisas. Note, por exemplo, que concatenar arrays também pode ser feito com o método "append" da classe Array, mas o fato da Apple se superar para construir isso como parte da linguagem demonstra qual é seu objetivo com o Swift.

Se quiser aprender Swift, e testar alguns desses pedaços de código, não poderia deixar de abordar o novo Playground no Xcode 6. O Playground permite testar o código enquanto estamos digitando, em tempo real. Ele executará tudo que digitar no Playground, e apresenta informações detalhadas sobre valores de variáveis, os valores de retorno das chamadas de funções e quantas vezes um determinado bloco de código foi executado.

Até 2014, a linguagem de programação usada para iOS era a Objective-C (criada no início dos anos 80 e mais tarde adquirida pela Apple). Ela continua sendo usada, só não é mais considerada a favorita para essa tarefa.

Na WWDC de 2014 foi apresentada a Swift, essa nova linguagem de programação desenvolvida pela própria Apple (pelo Chris Lattner e com a colaboração de outros programadores da empresa, claro), e de lá pra cá ela só cresceu no gosto do povo e até passou de uma licença proprietária para a Apache 2.0 (licença de software livre).

Então, para programar para iOS os caminhos mais tomados são o Objective-C e o Swift (existe o caminho das linguagens cross-plataform também, para se publicar em iOS e Android). E como Swift foi criada pela própria Apple, é essencial para programadores mobile iOS.

## **3.2 FUNÇÕES SÃO OBJETOS DE PRIMEIRA LINHA**

Funções como objetos de primeira classe e funções de alta ordem, podem ser encontrados em cada vez mais linguagens. Por exemplo, o Java 8 introduziu recentemente Lambdas. O conceito é simples, permitir que funções aceitem outras funções como parâmetros e que também retornem funções. Em sua simplicidade reside o poder deste conceito, uma vez que permite maiores abstrações. Por exemplo, pegue uma função de "filter" aplicada no array. A função de filtro permite filtrar um determinado array com base em qual desses itens satisfazem determinados critérios passados como parâmetro. A chave para ter um método super genérico é ser capaz de receber uma função como parâmetro.Tratar funções como objetos que podem ser referenciados e passados como parâmetro é bastante comum hoje em dia. Mas de novo, pode-se dizer que pela simplicidade como isso é feito no Swift é um conceito básico que fica ainda melhor com inferência de tipo

## **3.3 FORTEMENTE TIPADA COM INFERÊNCIA DE TIPO**

No mundo corporativo, estamos acostumados a trabalhar em cima da nossa rede de segurança fortemente tipada. Enquanto linguagens fortemente tipadas podem dar aquela segurança extra que as coisas (geralmente) não vão quebrar se compiladas, é uma experiência bem melhor quando essas linguagens também fazem inferência de tipo. Dessa forma, por exemplo, podemos fazer:

Repare na ausência do tipo no segundo exemplo. Como o Swift sabe que "bar" é uma String, não é necessário definir explicitamente (embora seja possível, como no primeiro exemplo). Isso pode parecer algo simples, mas certamente torna mais interessante quando é o tipo da função que é inferido

**4 APPLESCRIPT**

AppleScript é uma linguagem de script desenvolvida pela Apple para o sistema operacional Mac. Ele é usado principalmente para trocar dados entre diferentes aplicativos e para automatizar tarefas repetitivas. O exemplo anterior de redimensionar centenas de fotografias digitais e depois carregá-las em um site é um cenário típico em que o AppleScript seria usado no Mac OS.

O AppleScript usa uma metáfora de linguagem natural, o que significa que pode ser facilmente compreendido por usuários de computador sem experiência em programação. Por exemplo, um script que precisa fechar um determinado aplicativo de software seria algo assim:

Diga ao aplicativo 'Photoshop' para sair

A linguagem usa termos que agem como substantivos e verbos, tornando-a muito mais fácil de trabalhar do que outras línguas. AppleScript é fácil de usar para tarefas simples. Tarefas mais complexas são normalmente realizadas por aplicativos de software dedicados que você pode chamar usando um script. No exemplo das fotos digitais, o script pegaria uma fotografia, abriria um aplicativo como o Photoshop e faria com que esse aplicativo redimensionasse a fotografia. O poder de um script reside no fato de que você pode automatizar esse processo e redimensionar centenas de fotos dessa forma, usando apenas algumas linhas de código.

*AppleScript* é uma [linguagem de script](https://pt.wikipedia.org/wiki/Linguagem_de_script) que age sobre a interface do sistema operacional da Apple ([Mac OS X](https://pt.wikipedia.org/wiki/Mac_OS_X)) onde é possível realizar diversas mudanças e alterações de funcionamento e inclusive mesclar ferramentas e funções de um programa para outro com toda facilidade de programar uma linguagem intuitiva que é de se esperar de um [script](https://pt.wikipedia.org/wiki/Linguagem_de_script).

Apple Script foi projetada para ser usada como linguagem de usuário final, oferecendo aos usuários um mecanismo inteligente de controlo de aplicações, informações e documentos a fim de que se automatize fluxos de trabalho. Serve para controlar outras aplicações. Automatizar um fluxo de trabalho com AppleScript frequentemente reduz o tempo de performance de várias tarefas, reduz as chances de erro humano, proporciona resultados consistentes e cria um sistema de produção administrável pra trabalhar dentro dos prazos.

O projecto AppleScript é resultante do projecto [HyperCard](https://pt.wikipedia.org/wiki/HyperCard). O HyperCard tem uma linguagem em script com base na [língua inglesa](https://pt.wikipedia.org/wiki/L%C3%ADngua_inglesa) e noutra chamada [HyperTalk](https://pt.wikipedia.org/w/index.php?title=HyperTalk&action=edit&redlink=1), podendo assim usar a lógica e o comportamento. Os engenheiros da [Apple Inc.](https://pt.wikipedia.org/wiki/Apple_Inc.) viram que a linguagem em script podia ser usada em qualquer outra aplicação, e foi assim que o projeto da AppleScript nasceu. O [Mac OS](https://pt.wikipedia.org/wiki/Mac_OS) necessitou que upgrades (renovações) extensos fossem colocados na AppleScript. A maior parte das modificações tecnológicas foram feitas com o lançamento do [System 7](https://pt.wikipedia.org/wiki/System_7), mudando assim o conceito da Apple. AppleScript foi introduzido em [1992](https://pt.wikipedia.org/wiki/1992)[[1]](https://pt.wikipedia.org/wiki/AppleScript#cite_note-1) e lançado em outubro de [1993](https://pt.wikipedia.org/wiki/1993)[[2]](https://pt.wikipedia.org/wiki/AppleScript#cite_note-2) como parte do System 7.1.1 (System 7 Pro, o primeiro e principal upgrade do System 7). [QuarkXPress](https://pt.wikipedia.org/wiki/QuarkXPress) (ver.3.2) foi um dos primeiros e principais aplicativos de software que sustentou o AppleScript, e como consequência, o AppleScript foi amplamente adotado no segmento de divulgação do mercado Apple.

Um aspecto importante da implementação do AppleScript foi o Open Scripting Architecture (OSA). Apple fornece OSA para scripts / produtos de terceiros, tais como automação e QuicKeys Userland Frontier, para funcionar igualmente ao AppleScript. AppleScript foi implementado como um componente scripting, e as especificações de interface básicas são públicas, permitindo que outros desenvolvedores adicionem seus próprios scripts para componentes do sistema.Ex: AppleScript existe em várias aplicações do Mac OS X, da [Apple Inc.](https://pt.wikipedia.org/wiki/Apple_Inc.) e desenvolvedores terceiros. Aplicações com scripts estão no [Finder](https://pt.wikipedia.org/wiki/Finder), [Safari](https://pt.wikipedia.org/wiki/Safari), [iPhoto](https://pt.wikipedia.org/w/index.php?title=IPhoto&action=edit&redlink=1) e [iTunes](https://pt.wikipedia.org/wiki/ITunes), também no Adobe [Illustrator](https://pt.wikipedia.org/wiki/Illustrator) e [Photoshop](https://pt.wikipedia.org/wiki/Photoshop). [Bare Bones BBedit](https://pt.wikipedia.org/w/index.php?title=Bare_Bones_BBedit&action=edit&redlink=1) e [TextWrangler](https://pt.wikipedia.org/w/index.php?title=TextWrangler&action=edit&redlink=1), [Microsoft Word](https://pt.wikipedia.org/wiki/Microsoft_Word) e [Excel](https://pt.wikipedia.org/wiki/Microsoft_Excel)[[3]](https://pt.wikipedia.org/wiki/AppleScript#cite_note-3), [VMWare](https://pt.wikipedia.org/wiki/VMWare), Fusion e muitos outros.

## **5. O QUE É PRECISO PARA PROGRAMAR PARA IOS**

Você vai precisar criar [uma conta de desenvolvedor na Apple](https://developer.apple.com/pt/support/enrollment/), com ela você vai poder [baixar o Xcode IDE](https://developer.apple.com/xcode/ide/) no site deles mesmo e precisa estar usando um computador da Apple que tenha configuração adequada para usar o programa. Naturalmente, você vai precisar de uma boa conexão com a internet.

Para programar para iOS, utilizamos um SDK que só está disponível para Mac. Você que ainda não conhece muito bem o mundo Apple, como já disse, tudo tem que ser do Jeito Apple. Não adianta usar os famosos “Hachintoshs”, uma hora você vai precisar de um Mac. Recomendo que seja no mínimo um MacBook Pro, mas se você não tiver condições – um Mac Mini toma conta do recado, mas se puder comprar no mínimo um MacBook Pro.Para testarmos vários recursos, precisaremos de um dispositivo. Pode ser um iPhone, iPad ou mesmo um iPod Touch. No começo não haverá necessidade, o SDK tem um iOS Simulator, mas para alguns recursos realmente temos que testar em um device. Outro ponto é que, o iOS Simulator roda as aplicações com a capacidade de processamento do Mac. Então, você roda sua aplicação em um dispositivo com 8GB de RAM e Processador i7 e ela roda “lisinha”, agora quando você testa no device – as coisas mudam.A Apple tem um programa de desenvolvimento que custa US$99/ano. Para testar o app num device, utilizar alguns recursos e publicar na App Store você precisa estar inscrito.O Xcode é o SDK que utilizaremos para desenvolver para o iOS. É Grátis, e está disponível na Mac App Store. Tem em torno de 2GB, lembrando com o seu Mac OS deve estar atualizado.

O NSObject (Next Step Object) é, basicamente, o “modelo” de objeto do Objective-C (Linguagem de programação que utilizamos para desenvolver para iOS).Só explicando, a “Next Step” foi a empresa que Steve Jobs abriu quando foi mandado embora da Apple. Depois a Next Step foi comprada pela Apple e Steve Jobs voltou para o cargo de CEO.

**5.1 XCODE**

Xcode é um ambiente de desenvolvimento integrado e software livre da Apple Inc. para gerenciamento de projetos relacionados com o sistema operacional macOS. O Xcode possui ferramentas para o usuário criar e melhorar seus aplicativos. É um software poderoso e mais simples de utilizar para o desenvolvimento de aplicativos grandes. Antigamente o XCode era chamado de Project Builder.

O Xcode já vem com as ferramentas necessárias para desenvolver aplicações para o macOS, e suporta, por padrão, Objective-C, Swift e Apple-Script, que são linguagens de programação.

O Xcode tem um conjunto de "extras" para desenvolvimento, esses "extras" são chamados de SDK, e são fornecidos pela Apple Inc. no website de desenvolvimento de iOS deles.

Empresas e desenvolvedores independentes devidamente cadastrados no programa para desenvolvedor iOS da Apple, chamado iOS Developer Program, podem distribuir os aplicativos na App Store.Conforme é possível ver, além de ser uma IDE confiável e prática para programadores, ela possui suporte à diversas linguagens para a escolha do programador. Antes de existir o Swift, o Xcode se concentrava principalmente em Objective-C e C como desenvolvimento.

Além disso, O Xcode tem um conjunto de “extras” para desenvolvimento, que são chamados de SDK (Software Development Kit), e são fornecidos pela própria Apple no website de desenvolvimento. Empresas e desenvolvedores independentes que estejam devidamente cadastrados no programa para desenvolvedor iOS da Apple, chamado iOS Developer Program, podem publicar seus aplicativos na App Store.

**6.1 XCODE CLOUD**

Desenvolvimento de softwares que se baseia na colaboração entre as áreas de programação, segurança da informação/compliance e operações de forma simultânea. Esse trabalho colaborativo permite que novos produtos sejam criados com a mentalidade privacy by design, ou seja, com foco na proteção de dados do usuário desde as primeiras linhas de código. Contudo, nem sempre é fácil garantir que todos da equipe tenham essa visibilidade.

E aí que entra o Xcode Cloud — agora, além de usar o Xcode de forma solitária para trabalhar em seus projetos, é possível criar um ambiente colaborativo na nuvem no qual todos os membros do seu time possam visualizar compilações, realizar testes, enviar commits e assim por diante. Trata-se de uma funcionalidade que será adicionada ao IDE; não será necessário fazer o download de qualquer pacote adicional ou assinar uma mensalidade para ter acesso a tal privilégio, o que é uma excelente notícia para empresas de grande porte que hoje contratam soluções dispendiosas de colaboração.Tudo isso, é claro, com as devidas proteções contra eventuais ataques que podem comprometer a qualidade de seu trabalho e a segurança das informações,

**7. FRAMEWORK FOUNDATION**

O framework Foundation é o pão com manteiga da caixa de ferramentas do desenvolvedor iOS. Ele fornece a classe raiz NSObject e um grande número de blocos de construção fundamentais para o desenvolvimento iOS, de classes a números e strings, de arrays a dicionários. O framework Foundation pode parecer um pouco enroscado à primeira vista, mas ele aproveita bem a potência e é indispensável para o desenvolvimento de aplicativos iOS.

Enquanto o framework Foundation é implementado em Objective-C, o framework Core Foundation é baseado em C. Além desta diferença, o framework Core Foundation implementa um modelo de objeto limitado. Este modelo de objeto permite a definição de uma coleção de tipos opacos que são frequentemente referidos como objetos - apesar do fato de que eles não são, estritamente falando, objetos.

O framework Foundation é muito mais que uma coleção de classes para se trabalhar com números, string e coleções (arrays, dicionários e sets). Ele também define uma dezena de protocolos, funções, tipo de dados e constantes.

### **7.1 PROTOCOLOS**

Muitas linguagens, tais como Perl, Python e C++, fornecem suporte a [herança múltipla](https://en.wikipedia.org/wiki/Multiple_inheritance). Isso significa que uma classe pode ser descendente de mais que uma classe.Mesmo que o Swift e o Objective-C não forneçam suporte para herança múltipla, elas suportam herança múltipla através de uma especificação na forma de protocolos.Um protocolo define um esquema de métodos, propriedades e outros requisitos que se adequam a uma tarefa ou funcionalidade específica. O protocolo pode então ser adotado por uma classe, estrutura ou enumeração para fornecer uma implementação real desses requisitos. Qualquer tipo que satisfaça os requisitos de um protocolo está em conformidade com esse protocolo.

Um protocolo pode exigir que qualquer tipo em conformidade forneça uma propriedade de instância ou propriedade de tipo com um nome e tipo específicos. O protocolo não especifica se a propriedade deve ser armazenada ou computada - ele apenas especifica o nome e o tipo da propriedade necessária. O protocolo também especifica se cada propriedade deve ser gettable ou gettable e setable.

Se um protocolo requer que uma propriedade seja obtida e configurável, esse requisito de propriedade não pode ser cumprido por uma propriedade armazenada constante ou uma propriedade computada somente leitura. Se o protocolo requer que apenas uma propriedade seja obtida, o requisito pode ser satisfeito por qualquer tipo de propriedade, e é válido que a propriedade também seja configurável se isso for útil para seu próprio código.Os requisitos de propriedade são sempre declarados como propriedades variáveis, prefixadas com a palavra-chave var. As propriedades gettable e setable são indicadas escrevendo {get set} após sua declaração de tipo, e as propriedades gettable são indicadas escrevendo {get}.

Um protocolo pode exigir que qualquer tipo em conformidade forneça uma propriedade de instância ou propriedade de tipo com um nome e tipo específicos. O protocolo não especifica se a propriedade deve ser armazenada ou computada - ele apenas especifica o nome e o tipo da propriedade necessária. O protocolo também especifica se cada propriedade deve ser gettable ou gettable e setable.

Se um protocolo requer que uma propriedade seja obtida e configurável, esse requisito de propriedade não pode ser cumprido por uma propriedade armazenada constante ou uma propriedade computada somente leitura. Se o protocolo requer que apenas uma propriedade seja obtida, o requisito pode ser satisfeito por qualquer tipo de propriedade, e é válido que a propriedade também seja configurável se isso for útil para seu próprio código.

Os protocolos podem exigir métodos de instância específicos e métodos de tipo a serem implementados por tipos em conformidade. Esses métodos são escritos como parte da definição do protocolo exatamente da mesma maneira que para métodos normais de instância e tipo, mas sem chaves ou um corpo de método. Parâmetros variáveis ​​são permitidos, sujeitos às mesmas regras dos métodos normais. Os valores padrão, no entanto, não podem ser especificados para parâmetros de método dentro da definição de um protocolo.Tal como acontece com os requisitos de propriedade de tipo, você sempre prefixa os requisitos de método de tipo com a palavra-chave estática quando eles são definidos em um protocolo. Isso é verdade, embora os requisitos do método de tipo sejam prefixados com a classe ou palavra-chave estática quando implementados por uma classe:

Além de especificar os requisitos que os tipos em conformidade devem implementar, você pode estender um protocolo para implementar alguns desses requisitos ou implementar funcionalidades adicionais que podem aproveitar as vantagens dos tipos em conformidade.

#### **7.2 MÉTODOS OBRIGATÓRIOS E OPCIONAIS**

Relembre que todo método de um protocolo Swift é obrigatório. Se um tipo em conformidade com um protocolo Swift não implementar cada propriedade e método definido pelo protocolo, o compilador irá lançar um erro. Isso já não é igual para os protocolos do Objetive-C.

Em Objective-C, os métodos de um protocolo podem ser marcados como opcionais. Se um método de protocolo for opcional, o tipo em conformidade com o protocolo não é obrigado a implementar o método.

Por conta disso, a grande maioria dos frameworks do SDK do iOS são escritos em Objective-C, onde iremos encontrar uma grande quantidade de protocolos que definem métodos opcionais.

O que é importante de se lembrar é que toda propriedade e método de um protocolo Swift é obrigatório.

#### **7.3 BENEFÍCIOS**

Os protocolos têm múltiplos benefícios. Quando uma classe adota ou fica em conformidade com um protocolo, espera-se que a classe implemente o método declarado pelo protocolo.

Isso significa que um protocolo pode ser usado para declarar a interface de um objeto sem revelar o tipo do objeto.

A vantagem dos protocolos é que tipos, enums, classes e estruturas não relacionados, ainda podem compartilhar um comportamento semelhante por meio do uso de protocolos.

Apesar de termos coberto uma grande área de terreno neste artigo, nós mal arranhamos a superfície do que o framework Foundation tem para nos oferecer. Não é necessário saber os detalhes de todas as classes ou funções definidas no framework Foundation para começar com o desenvolvimento iOS.

**8. IOS 15**

A Apple anunciou o iOS 15 em setembro deste ano, uma atualização importante com recursos poderosos que melhoram a experiência do sistema operacional do iPhone. O iOS 15 torna as ligações do FaceTime mais naturais, apresenta o Memoji

Agora, você escolhe as roupas do seu Memoji e se expressa com novos adesivos. Mostre todo o seu estilo com várias cores de chapéus e acessórios.Foco

O recurso Foco ajuda você a se concentrar ou até a se desconectar um pouco. Escolha um Foco existente ou crie o seu para receber apenas as notificações que quiser enquanto se dedica totalmente ao trabalho ou termina um jantar sem distrações.

Com este recurso, as vozes parecem vir de onde cada pessoa está posicionada na tela, e as conversas fluem de forma ainda mais natural.

**8.1 APIS**

A companhia norte-americana apresentou ainda uma série de kits e APIs para facilitar a criação de aplicativos para seus dispositivos. No Apple Watch, uma API permite que usuários acessem os seus serviços de streaming direto no relógio inteligente. No Apple ID, o novo botão de assinatura Sign-In With Apple permitirá acesso a uma funcionalidade antifraude, confirmando que aquela pessoa é real. Também há novidades no PencilKit, SiriKit, MapsKit e Metal.

## **9. SISTEMA OPERACIONAL DO IPHONE**

O **iOS** é o [sistema operacional](https://www.techtudo.com.br/informatica/sistemas-operacionais/) móvel da [Apple](https://www.techtudo.com.br/tudo-sobre/apple.html). O software é utilizado no [iPhone](https://www.techtudo.com.br/tudo-sobre/iphone.html) e no [iPod Touc](https://www.techtudo.com.br/noticias/2019/05/ipod-touch-da-apple-ganha-nova-versao-com-chip-a10-veja-preco-no-brasil.ghtml)h, com o variante [iPadOS](https://www.techtudo.com.br/noticias/2019/06/ios-13-no-ipad-apple-anuncia-sistema-ipados-para-tablets.ghtml) no [iPad](https://www.techtudo.com.br/tudo-sobre/ipad.html), além de fazer integração com outros sistemas da marca, como o [macOS](http://www.techtudo.com.br/tudo-sobre/macos.html), do [Mac](http://www.techtudo.com.br/tudo-sobre/mac.html), e o [tvOS](https://www.techtudo.com.br/noticias/2018/06/tvos-12-novo-sistema-para-apple-tv-conta-com-suporte-ao-dolby-atmos.ghtml), da [Apple TV](http://www.techtudo.com.br/tudo-sobre/apple-tv.html). A primeira versão do iOS foi lançada em 2007 com o objetivo de dar praticidade às operações básicas dos smartphones da maçã e ser um sistema mais funcional em relação ao [Android](https://www.techtudo.com.br/tudo-sobre/android.html), principal concorrente

**Prós**

* Leveza
* Visual
* Rapidez
* Padronização
* Simplicidade de uso

**Contras**

* Não flexível
* Custo

**9.1 VÍRUS NO IPHONE?**

A segurança do iOS - o sistema operacional do iPhone - costuma ser maior do que o de seus concorrentes, porque a [Apple](https://canaltech.com.br/empresa/apple/) utiliza as chamadas *sandboxes*, nas quais os aplicativos da App Store são executados quando baixados.

Para entender um pouco melhor como esse processo funciona, no entanto, é necessário compreender o que é uma *sandbox*: em suma, ela se trata de uma área separada, onde programas podem ser executados sem danificar o resto do sistema caso algum deles contenha um vírus. Assim, como em “caixas de areia” (conforme a própria tradução do termo sugere), eles podem ser utilizados sem medo de qualquer interferência no iOS.

# **CONCLUSÃO**

Conseguimos compreender as aplicações, extensões, atualizações e os refatoramentos que o iOS teve ao longo dos anos, vimos o quão importante é ter o conhecimento de uma linguagem nativa para ser um ótimo desenvolvedor.

**Referências Bibliográficas**

Guimarães, Walter. Introdução a Objective-c introdução e motivo da sua utilização e implementação. **DevMedia,** publicado em 26/06/2011. Disponível em: <https://www.devmedia.com.br/introducao-ao-objective-c/23061>. Acesso em 21/10/2021

Ventura, Felipe. Apple Swift motivos pela linguagem de programação popular na época. **TecnoBlog,** publicado em 12/03/2018. Disponível em: <https://tecnoblog.net/236076/apple-swift-ranking-programacao-2018/>. Acesso em 21/10/2021

Alessandro. Meu primeiro aplicativo em Xcode para o desenvolvimento iOS. **DevMedia,** publicado em 19/09/2014. Disponível em: <https://www.devmedia.com.br/xcode-ios-introducao-ao-desenvolvimento-para-ios/29987>. Acesso em: 23/10/2021

Kovacs, Leandro. Swift como funciona o código Swift e o que é capaz de fazer. **TecnoBlog,** publicado em 06/05/2018. Disponível em: <https://tecnoblog.net/424081/o-que-e-o-codigo-swift-transferencias-internacionais/>. Acesso em: 23/10/2021

Medeiros, Henrique. SDK’s para aplicativos iOS e exemplos. **MobileTime,** publicado em 03/06/2019. Disponível em: <https://www.mobiletime.com.br/noticias/03/06/2019/apple-traz-novas-apis-e-sdks-para-apps/>. Acesso em: 23/10/2021

FRANÇA, Renan. Top 20 linguagens de programação. **ClassPath,** publicado em 26/11/2020. Disponível em <https://classpert.com/pt-BR/blog/linguagens->de-programac

ao-mais-usadas#Visual%20Basic..Acesso em: 23/10/2021.

KINAST, Priscila. IOS 15 Veja as novidades. **Oficina da Net,** publicado em 15/09/2021[.https://www.oficinadanet.com.br/apple/36769-ios-15-o-que-esperar-da-proxima-versao-do-sistema-da-apple](https://www.oficinadanet.com.br/apple/36769-ios-15-o-que-esperar-da-proxima-versao-do-sistema-da-apple).Acesso em: 23/10/2021.