# Ambientes de desenvolvimento de aplicativos móveis

## Apresentação

Atualmente, existem inúmeras possibilidades de configuração de ambientes para desenvolvimento de aplicativos para dispositivos móveis. Para montar seu ambiente, você deve considerar vários fatores, como qual será o sistema operacional, onde o aplicativo irá rodar, as premissas e as restrições do projeto de *software* e as linguagens com as quais você tem mais afinidade.

Embora existam muitas formas para configurar esse ambiente, há algumas linguagens que são mais utilizadas, assim como há alguns *frameworks* e ambientes integrados de desenvolvimento que podem lhe auxiliar, facilitando esse processo.

Nesta Unidade de Aprendizagem, você aprenderá mais sobre desenvolvimento nativo e híbrido para dispositivos móveis. Além disso, conhecerá algumas das linguagens de desenvolvimento mais utilizadas, assim como os ambientes que você pode utilizar como suporte. Por fim, verá como fazer a instalação das aplicações em seu sistema operacional.

Bons estudos.

**Ao final desta Unidade de Aprendizagem, você deve apresentar os seguintes aprendizados:**

- Definir os principais ambientes de desenvolvimento de aplicativos móveis.
- Descrever as principais linguagens de programação utilizadas no desenvolvimento de aplicativos móveis.
- Comparar os tipos de linguagens e os ambientes de desenvolvimento.

# Desafio

A escolha das ferramentas que irão apoiar no desenvolvimento de um aplicativo para dispositivos móveis é uma fase vital em projetos de *software*. Essa fase deve ser definida com base em informações contextuais levantadas sobre o projeto, principalmente em relação aos seus objetivos, em quais dispositivos o aplicativo precisará rodar, principalmente qual será o seu público-alvo.

Em relação ao público-alvo, é importante considerar a quantidade de pessoas que irá utilizar e a diversidade em relação à faixa etária, por exemplo.

Neste Desafio, você vai ter a chance de ajudar na tomada de decisão em relação ao projeto de desenvolvimento de um aplicativo para compartilhamento de informações relacionadas a eventos.

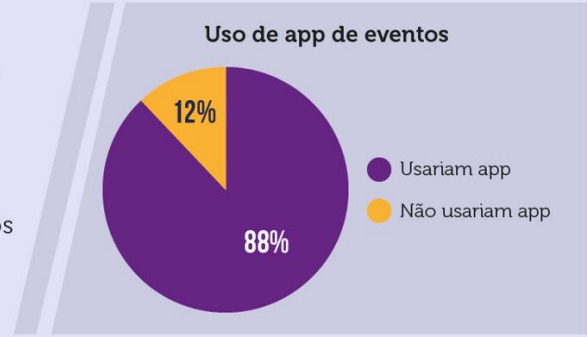
# DESENVOLVIMENTO DE APLICATIVOS DE EVENTOS

A Universidade ABCD procura disponibilizar à comunidade universitária acesso a eventos culturais variados dentro do *campus*, os quais vão desde *shows*, passando por atividades esportivas abertas ao público, palestras, feiras e teatro.

A comunidade é composta por milhares de pessoas, incluindo alunos, funcionários e ex-alunos. Embora alguns desses eventos sejam publicados no Facebook, foi feita uma pesquisa no *Campus*, e 80% dos entrevistados informaram que não têm conhecimento da maioria dos eventos que ocorrem na universidade. Alguns, por preferirem outras redes sociais, como o Twitter e o Instagram; outros simplesmente por não curtirem a página da universidade.

Entre as pessoas que informaram que não sabem dos eventos, 88% disse que utilizaria um aplicativo que mostrasse esses eventos, de acordo com seus interesses.

Pensando nisso, a universidade contratou a empresa na qual você trabalha para criar um aplicativo para centralizar esses eventos culturais, no qual seja possível cadastrar os eventos por tipo, de modo que os usuários interessados possam adicionar o evento diretamente em sua agenda.




## Esse projeto tem algumas restrições:

- [ ] A versão inicial do aplicativo precisa ficar pronta em um mês.
- [ ] Deve operar tanto em Android quanto em iOS.

Você é arquiteto de *software* e vai ajudar na decisão sobre as tecnologias que serão empregadas, assim como na indicação de contratação de desenvolvedores.

Aponte a câmera para o código e acesse o link do conteúdo ou clique no código para acessar.

a) Qual abordagem você seguiria, híbrida ou nativa?

b) Quais tecnologias empregaria?

c) Quanto aos desenvolvedores, especialistas em um dos sistemas operacionais ou com conhecimento de ambos?

Justifique as suas escolhas.

# Infográfico

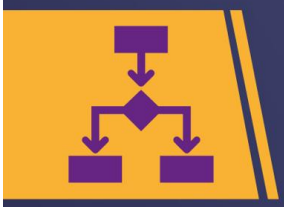
Quando é escolhida a abordagem que melhor se adapta às necessidades do projeto de *software* no qual se está inserido, alguns fatores devem ser considerados. Por exemplo, é importante saber que a abordagem híbrida apresenta um desempenho inferior se comparada à abordagem nativa. Por outro lado, a abordagem híbrida é melhor em relação a facilitar o reuso e a manutenção do código-fonte, por permitir que este rode em várias plataformas.

Existem prós e contras em cada uma das abordagens, e devem ser considerados quais fatores são mais importantes no projeto de *software*.

Neste Infográfico, você vai ver uma comparação entre algumas características do desenvolvimento híbrido em comparação ao nativo. Em seguida, para concluir este No Infográfico, confira o vídeo sobre o tema.

# DESENVOLVIMENTO HÍBRIDO X NATIVO

Vários fatores precisam ser considerados ao desenvolver um aplicativo para dispositivos móveis. Conheça alguns desses fatores, assim como o seu impacto de acordo com a **abordagem escolhida**.



## Fator – Reuso de código

Possibilidade de reaproveitar o mesmo código para diferentes plataformas.

**Híbrido:** reutilizável em qualquer plataforma

**Nativo:** códigos separados para cada plataforma

## Fator – Desempenho

Avaliação da eficiência do sistema.

**Híbrido:** regular

**Nativo:** experiência do usuário é responsiva e rápida

## Fator – Plataformas de desenvolvimento

Linguagens, bibliotecas e plug-ins que podem ser utilizados.

**Híbrido:** HTML, CSS e JavaScript

**Nativo:** IOS: Objective-C e Swift

Android: Java, Kotlin, entre outros

## Fator – Ciclo de desenvolvimento

Pode ser mais ágil, com entregas incrementais; ou mais lento, com entregas somente no final do ciclo.

**Híbrido:** rápido

**Nativo:** lento

## Fator – Custo

Investimento relacionado ao desenvolvimento (tempo, valores, etc.).

**Híbrido:** baixo

**Nativo:** alto

## Fator – Dependência de software de terceiros

O quanto o aplicativo precisará interagir com outros *softwares* para funcionar.

**Híbrido:** alta

**Nativo:** baixa

Aponte a câmera para o código e acesse o link do conteúdo ou clique no código para acessar.

O desenvolvimento *mobile* é um campo dinâmico e empolgante da tecnologia que envolve a criação de aplicativos e soluções para dispositivos móveis, como *smartphones* e *tablets*. Existem basicamente duas abordagens principais de desenvolvimento *mobile*: nativo e híbrido.

No vídeo a seguir, veja um pouco mais sobre cada um desses tipos, suas vantagens e seus desafios.

Aponte a câmera para o código e acesse o link do conteúdo ou clique no código para acessar.

# Conteúdo do Livro

Existem diversos componentes que precisam ser escolhidos e configurados quando um aplicativo vai ser desenvolvido para um dispositivo móvel. Primeiramente, é preciso escolher se será utilizada uma abordagem nativa, somente para Android ou para iOS; ou se será escolhida uma abordagem híbrida, para que o aplicativo possa rodar em ambas as plataformas.

Após escolher o sistema operacional em que o aplicativo será desenvolvido, é necessário definir qual linguagem de programação será utilizada no desenvolvimento, bem como os componentes adicionais, como *plug-ins* e bibliotecas que possam facilitar o trabalho. Por fim, também há diversas opções em relação ao ambiente de desenvolvimento, ou IDE. Uma boa combinação entre sistema operacional, *plug-ins* e IDE pode ajudar a poupar tempo e, consequentemente, dinheiro.

No capítulo Ambientes de desenvolvimento de aplicativos, da obra *Desenvolvimento para dispositivos móveis*, base teórica desta Unidade de Aprendizagem, você vai conhecer os ambientes, as linguagens de programação e os *plug-ins* mais utilizados para desenvolver aplicativos para Android e iOS.

Boa leitura.

# DESENVOLVIMENTO PARA DISPOSITIVOS MÓVEIS

Júlia Mara Colleoni Couto

SAGAH
SOLUÇÕES EDUCACIONAIS INTEGRADAS

# Ambientes de desenvolvimento de aplicativos móveis

## Objetivos de aprendizagem

Ao final deste texto, você deve apresentar os seguintes aprendizados:

- Definir os principais ambientes de desenvolvimento de aplicativos móveis.
- Descrever as principais linguagens de programação utilizadas no desenvolvimento de aplicativos móveis.
- Comparar os tipos de linguagens e os ambientes de desenvolvimento de aplicativos móveis.

## Introdução

Existem diversos sistemas operacionais para dispositivos móveis, cada um com linguagens e ambientes de programação específicos. Uma minoria utiliza Windows Phone, da Microsoft, RIM, da Blackberry, e o Symbian OS, da Nokia, porém, a maioria dos dispositivos atualmente executa os sistemas operacionais Android e *iPhone Operating System* (iOS). O Android teve sua versão Alpha lançada em 2008, e é um sistema baseado em Linux, desenvolvido e mantido pela Google. Sua licença é de código aberto, embora as decisões sobre o que será desenvolvido estejam sob o controle da Google (DEITEL; DEITEL; WALD, 2016). Seu núcleo de desenvolvimento é em C, mas também usa C++, PHP e Java. Já o iOS é desenvolvido e mantido pela Apple desde sua primeira versão em 2007. É um sistema operacional de código fechado, sendo desenvolvido nas linguagens C, C++, Objective-C, Swift e Java.

Existem dezenas de ferramentas, ambientes de programação e linguagens que podem ser utilizadas para o desenvolvimento de aplicativos móveis (BIØRN-HANSEN; GRØNLI; GHINEA, 2019) e, mesmo que você não utilize todas elas, deve saber que existem. Os comandos que apre-

sentaremos baseiam-se na instalação das aplicações usando o sistema operacional Ubuntu Linux. No entanto, vamos indicar a página Web de cada ferramenta, para que você possa fazer a instalação caso opte por outro sistema operacional.

Neste capítulo, você vai conhecer os principais ambientes, bem como as principais linguagens de programação utilizadas no desenvolvimento de aplicativos para dispositivos móveis. Verá, também, como instalar algumas das *Integrated Development Environment* (IDEs), linguagens de programação e *frameworks* mais utilizados.

## Ambientes integrados de desenvolvimento

Ambientes integrados de desenvolvimentos, ou IDEs são *softwares* que facilitam o desenvolvimento de sistemas. Eles são compostos por editor de código fonte, compilador, depurador e outras funcionalidades. Quando se trata do desenvolvimento para dispositivos móveis, os mais utilizados são Android Studio, Visual Studio Code, IntelliJ, Atom e Xcode. Um desenvolvedor iniciante, deve começar escolhendo apenas um, portanto, não é necessário instalar todos.

**Fique atento**

Cada aplicação possui requisitos específicos em relação a *hardware* e *software*, para que possa ser executada. Consulte o *site* do desenvolvedor para saber se seu computador atende aos requisitos, pois o não atendimento pode impactar em perda de desempenho ou em não conseguir executar a aplicação.

Para iniciar a instalação das aplicações, o primeiro passo é atualizar seu sistema operacional conforme indicações a seguir:

```
$ sudo apt-get update && apt-get upgrade
```

- Android Studio — plataforma desenvolvida pela Google com ferramentas para facilitar o desenvolvimento de aplicativos móveis que rodam o sistema Android. Disponibiliza um editor de código nativo e ferramentas para emulação e análise das aplicações. É um dos am-

bientes mais utilizados para desenvolvimentos voltados para Android (DEITEL; DEITEL; WALD, 2016). Roda em ambientes Windows, Mac, Linux e Chrome OS. Suporta o desenvolvimento de aplicativos em Java, Kotlin e C/C++.

***Link***

Para instalação do Android Studio no Linux faça o *download* disponível no *link* a seguir e siga os passos que constam no *site*.

**https://qrgo.page.link/3kPUA**

- Visual Studio Code — é um editor de código fonte desenvolvido pela Microsoft. Possui a funcionalidade de autocompletar inteligente, além de ferramentas para depuração de código, integração com repositórios de código fonte (como o Git) e possibilita a instalação de extensões e serviços adicionais. Pode ser instalado em ambientes Windows, Linux e MacOs, e suporta o desenvolvimento usando diversas linguagens de programação, como Java, JavaScript, Lua, Jade, CSS, Objective-C, TypeScript, Python, PHP, C++, Go, Markdown, entre outras. O comando para sua instalação no Linux é `$ sudo apt-get install code`.
- IntelliJ IDEA — é uma IDE para *Java Virtual Machine (*JVM), que suporta o desenvolvimento de linguagens como Java e Kotlin. Possui uma versão de código aberto denominada "*Community*" para desenvolvimento Android. Roda em ambientes Windows, MacOs e Linux, e na versão *Community*, permite o desenvolvimento de aplicativos em Java, Scala, Groovy e Kotlin. A versão *Ultimate* também suporta JavaScript e Typescript. Para instalar o IntelliJ via linha de comando no Linux, uma das alternativas é utilizar o Make, um utilitário de linha de comando que facilita a instalação de ambientes de desenvolvimento: `$ sudo apt install ubuntu-make`. Instale o IntelliJ usando o Make: `$ umake ide idea`.
- Atom IDE — desenvolvido pela equipe do Facebook, com o apoio de uma comunidade de desenvolvimento de código aberto, roda em ambientes Windows, MacOs e Linux. Oficialmente, suporta o desenvolvimento de aplicativos em Java, JavaScript, Typescript, C#, Flow e

PHP. Além disso, a comunidade desenvolveu pacotes para utilização de linguagens como R, Go, C++, Python e muitas outras. Para instalação no Linux é preciso adicionar o repositório (`$ sudo add-apt--repository ppa:webupd8team/atom`), atualizar o repositório: `$ sudo apt-get update`) e instalar o Atom (`$ sudo apt-get install atom`).

- Xcode — é a IDE oficial para iOS. Suporta o desenvolvimento de Objective-C, Swift e Apple-Script. Embora seja desenvolvido pela Apple e rode apenas em seu sistema operacional, é um *software* livre, para permitir que mais desenvolvedores de aplicações tenham acesso. A instalação padrão já inclui o Swift, mas também suporta o desenvolvimento de aplicativos em Objective-C. O Xcode é uma aplicação específica para iOS e MacOS.

Outras IDEs também bastante utilizadas incluem o NetBeans e Eclipse. Além dessas, se você tem bastante conhecimento de programação, sabe que é possível desenvolver aplicações utilizando editores de texto simples, embora seja mais complexo para identificar problemas no código.

## Linguagens de programação para o desenvolvimento de aplicativos móveis

Existe uma grande quantidade de linguagens de programação que podem ser utilizadas no desenvolvimento de aplicativos móveis. Segundo Vilete e Lopes (2018), algumas linguagens são nativas, ou seja, são desenvolvidas para operar em um sistema operacional específico, seja ele Android ou iOS. Contudo, há também outras linguagens, que permitem abordagens mais híbridas e resultam em aplicativos que funcionam em ambos os sistemas.

No desenvolvimento nativo, as aplicações são desenvolvidas pensando especificamente em um sistema operacional (VILETE; LOPES, 2018). Aplicações nativas acessam diretamente todos os recursos disponíveis no dispositivo, como *global positioning system* (GPS), acelerômetro e giroscópio, garantindo um melhor desempenho. Alguns aplicativos desenvolvidos com essa abordagem incluem Facebook Messenger, WhatsApp e Skype. Algumas linguagens são mais específicas para Android, entre elas, o Java e o Kotlin; já para o iOS, as linguagens mais utilizadas são Objective-C e Swift.

No desenvolvimento híbrido, as aplicações são desenvolvidas para que funcionem tanto em Android como em iOS. Essas aplicações são executadas

em um *container* que utiliza um recurso chamado *webview* para simular a execução de uma aplicação nativa. *Webview* é um navegador que é executado quando o usuário inicializa uma aplicação híbrida (VILETE; LOPES, 2018). Neste caso, os aplicativos se baseiam em HTML, CSS e Javascript. Comparando com o desenvolvimento nativo, o desenvolvimento híbrido tem a possibilidade de ter um único código para executar em ambos os sistemas, reduzindo o tempo e o custo com desenvolvimento e manutenção. Por outro lado, abordagens híbridas perdem em relação ao desempenho quando comparadas com abordagens nativas.

Para que você possa programar em determinada linguagem, é necessário ter o *software development kit* (SDK) instalado. Um SDK possui um conjunto de *frameworks* e ferramentas que serão utilizados junto da linguagem de programação, para que código possa ser reconhecido. Quando você instala uma IDE, já pode escolher em quais linguagens vai programar e, se necessário, adicionar os respectivos *plug-ins*. Porém, em algumas linguagens, como Java e Python, você pode instalar o SDK separadamente. A seguir, você verá algumas das principais linguagens para desenvolvimento de aplicativos móveis, bem como algumas de suas características.

- Java — mantido pela Oracle, o Java é composto por uma linguagem de programação e uma plataforma computacional, que é utilizada como base por muitas aplicações. Está disponível em duas versões: *Java Runtime Environment* (JRE) e *Java Development Kit* (JDK). O JRE é utilizado na execução das aplicações e o JDK, no seu desenvolvimento. Por esse motivo, é necessário instalar o JDK. Java é uma linguagem de programação popular, sendo considerada a base para o desenvolvimento Android. É orientada a objetos, o que significa ser baseada na modelagem e comunicação entre os objetos. Java também é uma linguagem estaticamente tipada, ou seja, o usuário precisa declarar o tipo de dados que será armazenado em cada variável declarada. Por exemplo, se a variável for tipada com "INT", só será permitido guardar números inteiros nessa variável. Seu comando para instalação no Linux é `$ sudo apt-get install default-jdk`.
- JavaScript — é uma linguagem de programação usada principalmente para controlar o *Hypertext Markup Language* (HTML) e o *Cascading Style Sheets* (CSS) e manipular comportamentos em uma página Web. É mantido pela *European Computer Manufacturer's Association* (ECMA). Originalmente, o JavaScript foi criado para o desenvolvimento de aplicações no lado cliente, mas já evoluiu para possibilitar

o desenvolvimento de aplicações *desktop* e no lado servidor também. O Node.js, por exemplo, é um ambiente JavaScript que é utilizado no lado servidor. Para instalação no Linux, você deve instalar o *plug-in* na IDE que estiver utilizando para o desenvolvimento.

**Fique atento**

Java e JavaScript não são a mesma coisa; não há nenhuma relação entre as duas linguagens de programação. No entanto, elas possuem recursos parecidos, como objetos, variáveis, operadores e métodos. Sendo assim, se você souber desenvolver em uma delas, isso poderá ajudá-lo a aprender a outra de forma mais rápida.

- Kotlin — é uma linguagem mais recente e que tem ganhado força nos últimos tempos. É desenvolvida e mantida pela JetBRains e 100% compatível com Java e JavaScript. Assim como Java, Kotlin utiliza o paradigma orientado a objetos e tem suporte ao paradigma funcional, com o uso de expressões lambda (anônimas). Seu comando para instalação no Linux é `$ sudo snap install—classic kotlin`.
- TypeScript — é uma linguagem de programação desenvolvida pela Microsoft, e que adiciona tipagem e outros recursos ao JavaScript. A tipagem possibilita que o desenvolvedor declare o tipo de uma variável, como numérico, textual ou data, por exemplo. Ele permite desenvolver aplicações tanto do lado do cliente como do lado do servidor. Seu comando para instalação no Linux é `$ sudo npm install -g typescript`.
- Objective-C — é um superconjunto da linguagem de programação C, ou seja, agrega recursos ao C. Ele possibilita o uso do paradigma programação orientada à objetos, contendo sintaxe para a criação de métodos e classes. Instale o *plug-in* na IDE que estiver utilizando para o desenvolvimento.
- Swift — foi desenvolvido baseado no Objective-C, que é a linguagem oficial da Apple atualmente. O Swift possui uma sintaxe mais concisa e apresenta um melhor desempenho em relação ao seu antecessor.

***Link***

Para instalação do Swift no Linux, verifique as instruções no *link* a seguir.

**https://swift.org/download/**

Além das linguagens de programação apresentadas, ainda é possível utilizar Python, PHP, Pearl, Rust e muitas outras. A escolha da linguagem mais adequada depende de seus conhecimentos técnicos, aptidões, e, principalmente, do contexto e do foco que será dado ao aplicativo que você deseja desenvolver.

***Link***

O GitHub é um dos repositórios de código fonte com controle de versionamento mais utilizados pelos desenvolvedores de todo o mundo. O *site* disponibiliza uma ferramenta para visualização de dados relacionados às linguagens de programação mais utilizadas nos repositórios hospedados no nele. Acesse o *link* a seguir e saiba mais.

**https://github.com/**

## Linguagens e ambientes de desenvolvimento

No Quadro 1, você pode observar uma correlação entre as principais linguagens de programação e as IDEs que suportam seu desenvolvimento. Note que o Atom e o Code estão entre as IDEs que suportam a maior quantidade de linguagens.

**Quadro 1.** Correlação entre IDEs e linguagens de programação suportadas para o desenvolvimento de aplicativos móveis

| IDE linguagem | Android Studio | Visual Studio Code | IntelliJ | Atom IDE | Xcode |
|---|---|---|---|---|---|
| Java | X | X | X | X | |
| JavaScript | | X | X | X | |
| Kotlin | X | | X | X | |
| TypeScript | | X | X | X | |
| Objective-C | | X | | X | X |
| Swift | | X | | X | X |

## Outros *frameworks* e aplicações úteis

Muitas vezes, é necessário instalar outros *plug-ins*, *frameworks* e aplicações complementares, além das IDEs. O desenvolvimento de aplicativos depende da instalação de algumas plataformas, editores de código e *frameworks*, além de pacotes e módulos. Existem diversas ferramentas gratuitas que podem ser utilizadas pelos programadores durante o desenvolvimento de suas aplicações. Para ajudá-lo na configuração do seu ambiente de desenvolvimento, vamos mostrar como proceder a instalação de algumas das ferramentas mais utilizadas.

- Node.js — é um interpretador de código JavaScript, usado no desenvolvimento de aplicações baseadas em rede. É conhecido por ser um ambiente de execução *server-side* e por possibilitar que as aplicações rodem independentemente de um navegador (de forma independente em uma máquina). Comando para instalação no Linux: `$ sudo apt-get install nodejs`.
- Node Package Manager (Npm) — é um gerenciador de pacotes do Node, que facilita a instalação de ferramentas para desenvolvimento de aplicações usando JavaScript. Atualmente, é mantido pela Node.js Foundation, como um *software* livre de código aberto. Seu comando para instalação no Linux é `$ sudo apt-get install npm`.
- Apache Cordova — é um *framework* mantido pela Apache Foundation, que permite usar CSS3, HTML5 e JavaScript para o desenvolvimento

de aplicações para dispositivos móveis multiplataforma, com base em componentes do tipo *webView*. Possibilita que o código acesse os recursos nativos do dispositivo, como o GPS, acelerômetro, câmera. Seu comando para instalação no Linux é `$ sudo npm install -g cordova`.

- Ionic — é um *framework* para desenvolvimento de aplicações para dispositivos móveis híbridos. É responsável pela parte mais visual da aplicação, tratando o código para que ele fique com a aparência do dispositivo no qual está rodando (iOS, Android, etc). Seu comando para instalação no Linux é `$ sudo npm install -g ionic`.
- Angular — trata-se de um *framework* e plataforma para desenvolvimento de interfaces de aplicações móveis e Web baseadas em JavaScript, CSS ou HTML. É um *front-end* baseado em Typescript, mantido pela equipe do Google, cujo comando para instalação no Linux é `$ sudo npm install -g @angular/cli`.

Você também pode precisar instalar outras aplicações ou pacotes, de acordo com a aplicação que estiver desenvolvendo. O Npm tem dezenas de pacotes e módulos que você pode explorar visitando o *site* oficial da aplicação. Neste capítulo, você teve uma visão geral sobre os principais ambientes de desenvolvimento, linguagens de programação utilizados no desenvolvimento de aplicativos móveis. Caso tenha interesse em aprender mais sobre determinada ferramenta, consulte o *site* do fabricante.

## Referências

BIØRN-HANSEN, A.; GRØNLI, T. M.; GHINEA, G. A Survey and Taxonomy of Core Concepts and Research Challenges in Cross-Platform Mobile Development. *ACM Computing Surveys (CSUR)*, New York, v. 51, n. 5, Article No. 108, Jan. 2019.

DEITEL, P.; DEITEL, H.; WALD, A. *Android 6 para programadores*: uma abordagem baseada em aplicativos. 3. ed. Porto Alegre: Bookman, 2016. 618 p.

VILETE, A. C. S.; LOPES, T. M. *Frameworks para o desenvolvimento de aplicações mobile multiplataforma*. Orientador: Luiz Gustavo Lourenço Moura. 2018. 80 f. Monografia (Bacharelado em Sistemas de Informação) – Instituto Federal Fluminense, Campos dos Goytacazes, 2018. Disponível em: http://bd.centro.iff.edu.br/xmlui/handle/123456789/2184. Acesso em: 21 jun. 2019.

## Leituras recomendadas

ANDROID Developers. [*S. l.*: *S. n.*], 2019. Disponível em: https://developer.android.com. Acesso em: 21 jun. 2019.

ANGULAR: One framework. Mobile & desktop. *Angular*, [*S. l.*], 2019. Disponível em: https://angular.io. Acesso em: 21 jun. 2019.

APLICATIVOS híbridos e nativos: entenda as principais diferenças. *ComputerWorld*, São Paulo, 28 set. 2018. Disponível em: https://computerworld.com.br/2018/09/28/aplicativos-hibridos-e-nativos-entenda-as-principais-diferencas/. Acesso em: 21 jun. 2019.

ATOM IDE: Community Powered. *GitHub*, [*S. l.*], 2019. Disponível em: https://ide.atom.io. Acesso em: 21 jun. 2019.

CORDOVA: Mobile apps with HTML, CSS & JS. *Apache Software Foundation*, Wakefield, 2015. Disponível em: https://cordova.apache.org. Acesso em: 21 jun. 2019.

CORRÊA, L. C. *Desafios, agilidade e simplicidade*: uma abordagem para desenvolvimento mobile. Orientadores: Milene Serrano; Maurício Serrano. 2018 121 f. Monografia (Bacharelado em Engenharia de Software) – Faculdade do Gama, Universidade de Brasília, Gama, 2018. Disponível em: http://bdm.unb.br/handle/10483/21578. Acesso em: 21 jun. 2019.

INTELLIJ IDEA: Capable and Ergonomic IDE for JVM. *JetBrains*, Praha, 2019. Disponível em: https://www.jetbrains.com/idea. Acesso em: 21 jun. 2019.

IONIC: Cross-Platform Mobile App Development. *Ionic*, Madison, 2019. Disponível em: https://ionicframework.com. Acesso em: 21 jun. 2019.

JAVA. *Oracle Corporation*, Redwood Shores, 2019. Disponível em: https://java.com. Acesso em: 21 jun. 2019.

JAVASCRIPT. *Pluralsight*, Farmington, 2019. Disponível em: https://www.javascript.com. Acesso em: 21 jun. 2019.

KOTLIN Programming Language. *Kotlin Foundation*, Delaware, 2019. Disponível em: https://kotlinlang.org. Acesso em: 21 jun. 2019.

NODEJS: a JavaScript runtime built on Chrome's V8 JavaScript engine. *OpenJS Foundation*, [*S. l.*], 2019. Disponível em: https://nodejs.org. Acesso em: 21 jun. 2019.

NPM: the world's largest software registry. *npm, Inc.*, Oakland, 2019. Disponível em: https://www.npmjs.com. Acesso em: 21 jun. 2019.

OBJECTIVE C Runtime. *Apple*, Cupertino, 2019. Disponível em: https://developer.apple.com/documentation/objectivec. Acesso em: 21 jun. 2019.

SWIFT: a general-purpose programming language built using a modern approach to safety, performance, and software design patterns. *Apple*, Cupertino, 2019. Disponível em: https://swift.org. Acesso em: 21 jun. 2019.

TYPESCRIPT: a superset of JavaScript that compiles to clean JavaScript output. *Microsoft; GitHub*, [*S. l.*], 2019. Disponível em: https://github.com/microsoft/TypeScript. Acesso em: 21 jun. 2019.

VENTEU, K. C.; PINTO, G. S. Desenvolvimento Móvel Híbrido. *Interface Tecnológica*, Taquaritinga, v. 15, n. 1, p. 86–96, 2018. Disponível em: https://revista.fatectq.edu.br/index.php/interfacetecnologica/article/view/337. Acesso em: 21 jun. 2019.

VISUAL Studio Code: Code Editing. Redefined. *Microsoft*, Seattle, 2019. Disponível em: https://code.visualstudio.com. Acesso em: 21 jun. 2019.

XCODE. *Apple*, Cupertino, 2019. Disponível em: https://developer.apple.com/xcode. Acesso em: 21 jun. 2019.

Encerra aqui o trecho do livro disponibilizado para esta Unidade de Aprendizagem. Na Biblioteca Virtual da Instituição, você encontra a obra na íntegra.

Conteúdo:

SAGAH
SOLUÇÕES EDUCACIONAIS INTEGRADAS

# Dica do Professor

A escolha das ferramentas que irão compor um ambiente de desenvolvimento depende de vários fatores, entre eles o sistema operacional e a linguagem de programação que você vai utilizar. Caso você pretenda testar várias configurações de ambientes, o ideal é escolher um editor que suporte diferentes linguagens e *frameworks*.

Nesta Dica do Professor, você vai aprender algumas das características do Visual Studio Code, além de como instalar e configurar extensões para facilitar o desenvolvimento de seus aplicativos.

Aponte a câmera para o código e acesse o link do conteúdo ou clique no código para acessar.

# Exercícios

1) **Ambientes integrados de desenvolvimento ajudam os programadores, disponibilizando funções como autocompletar e indicações de problema na sintaxe.**

**Entre os itens a seguir, quais são exemplos de IDEs?**

A) Atom, iOS, IntelliJ e Android Studio.

B) Atom, Xcode, Kotlin e Android Studio.

C) Atom, Xcode, IntelliJ e TypeScript.

D) Atom, Xcode, IntelliJ e Android Studio.

E) Atom, Swift, IntelliJ e Android Studio.

2) **Duas abordagens se destacam quando se fala em desenvolvimento para aplicativos móveis: o desenvolvimento nativo e o híbrido.**

**Quanto ao desenvolvimento nativo, é correto afirmar que:**

A) aplicações são desenvolvidas para rodar em qualquer sistema operacional.

B) o desempenho é inferior se comparado ao desenvolvimento híbrido.

C) utilizam uma *webview*, a qual abre um navegador para exibir conteúdo ao usuário.

D) as aplicações não podem acessar diretamente o GPS e o acelerômetro.

E) as aplicações são desenvolvidas para um sistema operacional específico.

3) **Algumas linguagens de programação têm características mais específicas. Java e Kotlin, por exemplo, são linguagens estaticamente tipadas.**

**O que isso significa?**

A) A declaração dos tipos de dados é opcional, não interferindo na compilação do código.

B) As categorias de dados a serem armazenados precisam ser explicitamente declaradas.

C) Os tipos de dados são implícitos, não sendo permitida a declaração estática.

D) Essas linguagens só trabalham com dados dos tipos textuais e numéricos.

E) Existe apenas um tipo genérico.

4) **Algumas linguagens de programação são evoluções, outras adicionam funcionalidades, enquanto outras não apresentam muito em comum.**

**Qual a correlação entre o Java e o JavaScript?**

A) O JavaScript é uma evolução do Java para aplicativos móveis.

B) Ambas são linguagens que possibilitam o desenvolvimento de aplicativos híbridos.

C) Ambas são linguagens de programação orientadas a objetos.

D) JavaScript é um superconjunto de Java, agregando funções ao Java.

E) Javascript é uma linguagem mais antiga e mais utilizada.

5) **Diversos *plug-ins* e *frameworks* podem ser utilizados para apoiar o desenvolvimento para aplicativos móveis.**

**Qual deles pode ser utilizado para facilitar a instalação de outras bibliotecas e componentes?**

A) Node.js.

B) Npm.

C) Apache Cordova.

D) Ionic.

E) Angular.

# Na prática

O desenvolvimento de um aplicativo que funcione em multiplataforma pode ser um desafio para um programador inexperiente. Mesmo um programador mais experiente pode precisar aprender a trabalhar com novas linguagens ou *frameworks*, de modo a poder desenvolver algum projeto específico.

Neste Na Prática, você vai conhecer um estudo de caso no qual o desenvolvedor precisou tomar decisões relacionadas às ferramentas que ele iria utilizar para desenvolver um aplicativo.

# SOLUÇÕES PARA DESENVOLVIMENTO DE APLICATIVO

A escolha das ferramentas mais adequadas para o desenvolvimento de uma aplicação não é tarefa trivial; requer análise do contexto e dos conhecimentos de quem vai desenvolver o sistema. Existem muitas linguagens de programação, *frameworks* e outros *plug-ins*, que podem ser combinados das formas mais diversas.

## Perfil do desenvolvedor

Martin é desenvolvedor de *software* há 3 anos, tendo atuado em diversos projetos usando JavaScript. Ele é autônomo, e prospecta alguns projetos em que ele mesmo é responsável por todo o ciclo de desenvolvimento do *software*, desde a análise até a manutenção do sistema, posterior à entrega.

Aponte a câmera para o código e acesse o link do conteúdo ou clique no código para acessar.

## Contexto

Em seu mais novo projeto, Martin precisa desenvolver um *chatbot*, para auxiliar a matrícula de alunos na AlphaBetaGama, uma instituição que oferece cursos de graduação.

Os principais requisitos da aplicação seriam os seguintes:

O usuário deve interagir com o chatbot, que, após validar a matrícula, vai sugerir disciplinas para o aluno cursar, atualizando as sugestões à medida que o aluno vai escolhendo as disciplinas.

Uma versão funcional deve ser entregue em dois meses.

A aplicação precisa funcionar em dispositivos móveis (Android e iOS), assim como em navegadores web, de modo que possa ser acessada em desktops.

## Problema

Uma das primeiras decisões precisava ser relacionada à arquitetura. Ele sabia que teria que aprender novas tecnologias, mas, ao mesmo tempo, precisava aproveitar os seus conhecimentos, para poupar tempo.

## Solução

Para desenvolver o *chatbot*, Martin fez algumas pesquisas e decidiu por uma abordagem híbrida, com as seguintes ferramentas:

- ☐ **TypeScript:** por ter conhecimento em JavaScript, ele terá maior facilidade em aprender essa linguagem.
- ☐ **Node.js:** apresenta bom desempenho em aplicações *real-time*, como servidores de *chat*.
- ☐ **Ionic:** facilita o desenvolvimento híbrido, permitindo que uma versão da aplicação possa ser usada tanto em iOS como em Android.
- ☐ **Angular:** além de ter uma integração boa com o Ionic, possibilita que o reuso do código sirva para *desktops* e dispositivos móveis.

## Conclusão

Martin conseguiu entregar o projeto, gastando menos tempo em aprender novas tecnologias, e com o custo de manutenção do código reduzido.

# Saiba mais

Para ampliar o seu conhecimento a respeito desse assunto, veja abaixo as sugestões do professor:

## VSCode

Conheça o Visual Studio Code da Microsoft, programa para desenvolvimento de aplicativos para dispositivos móveis.

Aponte a câmera para o código e acesse o link do conteúdo ou clique no código para acessar.

## Aplicativos híbridos e nativos: entenda as principais diferenças

O seguinte artigo traz algumas estatísticas, assim como um resumo das diferenças entre aplicativos híbridos e nativos. Confira.

Aponte a câmera para o código e acesse o link do conteúdo ou clique no código para acessar.

## Desenvolvimento móvel híbrido

O seguinte artigo compara e mostra as vantagens do desenvolvimento híbrido em relação ao desenvolvimento nativo.

Aponte a câmera para o código e acesse o link do conteúdo ou clique no código para acessar.