

SERVICES ARCHITECTURE, API  
E MOBILE ARCHITECTURE

# **IMPLEMENTANDO UM APLICATIVO HÍBRIDO COM IONIC – PARTE II**

DIEGO GARCIA E CHRISTIANE DE PAULA REIS



## LISTA DE FIGURAS

Figura 7.1 – Adicionando a plataforma Android no projeto.....	6
Figura 7.2 – Rodando o projeto no Android.....	7
Figura 7.3 – Projeto rodando no <i>smartphone</i> .....	7
Figura 7.4 – Ionic DevApp pronto para ser vinculado com o projeto .....	9
Figura 7.5 – Ilustração da implementação.....	12



**LISTA DE CÓDIGOS-FONTE**

Código-fonte 7.1 – Adicionando a plataforma Cordova no projeto .....	6
Código-fonte 7.2 – Rodando o projeto no Android .....	6
Código-fonte 7.3 – Rodando o projeto com o comando <i>livereload</i> .....	8
Código-fonte 7.4 – Rodando o comando para criar o servidor que vinculará no aplicativo Ionic DevApp .....	8
Código-fonte 7.5 – Implementando botão para abrir a câmera no home.html..	10
Código-fonte 7.6 – Instalando o plugin Cordova em nosso projeto e seu componente no Ionic-Native .....	10
Código-fonte 7.7 – Implementando a scanner no home.ts .....	11

## SUMÁRIO

7 IMPLEMENTANDO UM APLICATIVO HÍBRIDO COM IONIC .....	5
7.1 Vamos gerar aplicativos mobile!.....	5
7.2 Testando no <i>smartphone</i> .....	5
7.3 Utilizando o comando <i>run</i> .....	5
7.4 Utilizando o comando run com o livereload .....	7
7.5 Deixando as coisas ainda mais fáceis com o Ionic Dev App.....	8
7.6 Aplicando funcionalidades nativas .....	9
CONCLUSÃO.....	13
REFERÊNCIAS.....	14

## 7 IMPLEMENTANDO UM APLICATIVO HÍBRIDO COM IONIC

### 7.1 Vamos gerar aplicativos mobile!

No capítulo anterior, vimos Aplicações Híbridas, Cordova e o Ionic, e foi desenvolvido um simples aplicativo que gera QR Codes de games.

Agora, vamos adicionar algumas funcionalidades ao aplicativo, que utilizará recursos nativos e, principalmente, veremos nosso aplicativo rodar em um *smartphone*.

### 7.2 Testando no *smartphone*

É muito simples para rodarmos nosso aplicativo em dispositivos mobile.

Para dispositivos Android, antes de tudo, você precisará do Android Studio instalado em sua máquina com o SDK. Também será necessário o JDK do Java, caso você ainda não o possua.

Links para downloads:

Android Studio – <<https://developer.android.com/studio/>>.

Java JDK 11.0.4 –  
<<https://www.oracle.com/technetwork/java/javase/downloads/jdk11-downloads-5066655.html>>.

Para rodarmos em dispositivos iOS, você precisará de um Mac da Apple e, com isso, basta você ter o XCode instalado nele.

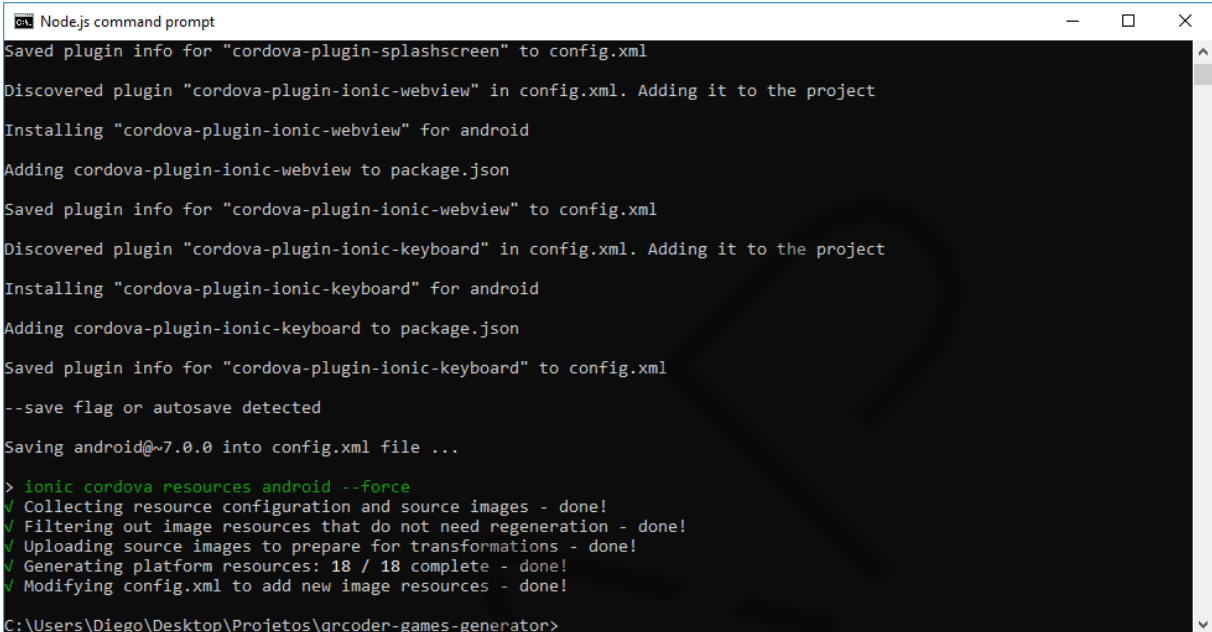
Vamos dar mais foco aos dispositivos Android, porém, você pode testar normalmente em seu iOS com um Mac.

### 7.3 Utilizando o comando *run*

Em dispositivos Android, você poderá testar por um emulador no Android Studio ou pelo seu celular com o modo depuração ativado e conectado à USB. Dessa forma, vamos adicionar a plataforma Android ao nosso projeto Ionic.

```
ionic cordova platform add android
```

Código-fonte 7.1 – Adicionando a plataforma Cordova no projeto  
Fonte: Elaborado pelo autor (2018)



```
Node.js command prompt
Saved plugin info for "cordova-plugin-splashscreen" to config.xml
Discovered plugin "cordova-plugin-ionic-webview" in config.xml. Adding it to the project
Installing "cordova-plugin-ionic-webview" for android
Adding cordova-plugin-ionic-webview to package.json
Saved plugin info for "cordova-plugin-ionic-webview" to config.xml
Discovered plugin "cordova-plugin-ionic-keyboard" in config.xml. Adding it to the project
Installing "cordova-plugin-ionic-keyboard" for android
Adding cordova-plugin-ionic-keyboard to package.json
Saved plugin info for "cordova-plugin-ionic-keyboard" to config.xml
--save flag or autosave detected
Saving android@7.0.0 into config.xml file ...
> ionic cordova resources android --force
✓ Collecting resource configuration and source images - done!
✓ Filtering out image resources that do not need regeneration - done!
✓ Uploading source images to prepare for transformations - done!
✓ Generating platform resources: 18 / 18 complete - done!
✓ Modifying config.xml to add new image resources - done!
C:\Users\Diego\Desktop\Projetos\qrcoder-games-generator>
```

Figura 7.1 – Adicionando a plataforma Android no projeto  
Fonte: Google Imagens (2018)

Com esse comando, nosso projeto está pronto para ser compilado em um aplicativo Android. Após isso, basta abrir seu emulador ou *plugar* seu celular em sua máquina e utilizar o comando:

```
ionic cordova run android
```

Código-fonte 7.2 – Rodando o projeto no Android  
Fonte: Elaborado pelo autor (2018)

Caso as variáveis de ambiente do JDK do Java e do SDK do Android Studio estejam corretas, será gerado o aplicativo no seu dispositivo.

```
Selecionar Node.js command prompt
:app:assembleDebug UP-TO-DATE
:app:cdvBuildDebug UP-TO-DATE

BUILD SUCCESSFUL in 5s
46 actionable tasks: 1 executed, 45 up-to-date

Built the following apk(s):
  C:\Users\Diego\Desktop\Projetos\qrcoder-games-generator\platforms\android\app\build\outputs\apk\debug\app-debug.apk
Using apk: C:\Users\Diego\Desktop\Projetos\qrcoder-games-generator\platforms\android\app\build\outputs\apk\debug\app-debug.apk
Package name: io.ionic.starter
LAUNCH SUCCESS

[OK] Your app has been deployed.
    Did you know you can live-reload changes from your app with --livereload?

C:\Users\Diego\Desktop\Projetos\qrcoder-games-generator>
```

Figura 7.2 – Rodando o projeto no Android  
Fonte: Google Imagens (2018)

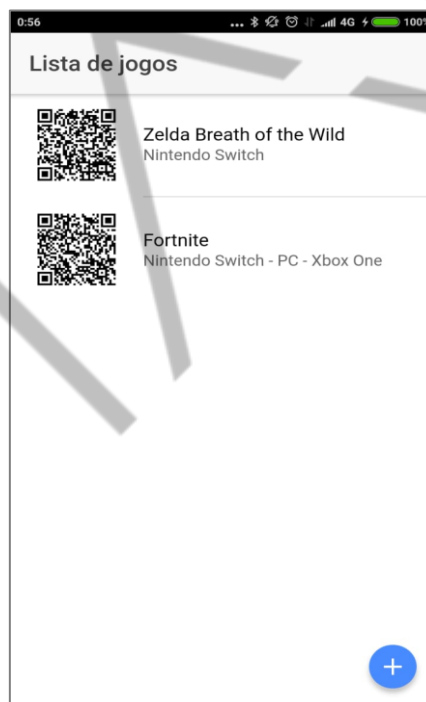


Figura 7.3 – Projeto rodando no *smartphone*  
Fonte: Google Imagens (2018)

## 7.4 Utilizando o comando run com o livereload

Podemos melhorar nossa experiência de desenvolvimento com o comando *livereload*, cujo uso no terminal o Ionic já sugere. Para isso, basta utilizar o comando:

```
ionic cordova run android --livereload
```

Código-fonte 7.3 – Rodando o projeto com o comando *livereload*

Fonte: Elaborado pelo autor (2018)

Para utilizar o *livereload*, seu *smartphone* deve ser conectado na mesma rede do computador em que você está desenvolvendo.

Assim, nosso aplicativo executará as mudanças que realizamos no código em tempo real de execução.

## 7.5 Deixando as coisas ainda mais fáceis com o Ionic Dev App

Caso haja muitos problemas para rodar no seu *smartphone*, você tem, ainda, a opção de rodar em um aplicativo desenvolvido pelo time do Ionic para realizar teste de aplicações de uma maneira muito simples e rápida.

Entre no Google Play ou Apple Store e baixe o aplicativo Ionic DevApp. Depois de instalado, abra o aplicativo que já fornecerá as instruções de como utilizá-lo. Se você leu rapidamente, basta o *smartphone* e o computador estarem conectados na mesma rede, permitirmos a conexão e utilizarmos o comando:

```
ionic serve -c
```

Código-fonte 7.4 – Rodando o comando para criar o servidor que vinculará no aplicativo Ionic DevApp

Fonte: Elaborado pelo autor (2018)

Desse modo, temos nosso aplicativo rodando funcional no seu *smartphone*, sem nenhuma dor de cabeça.



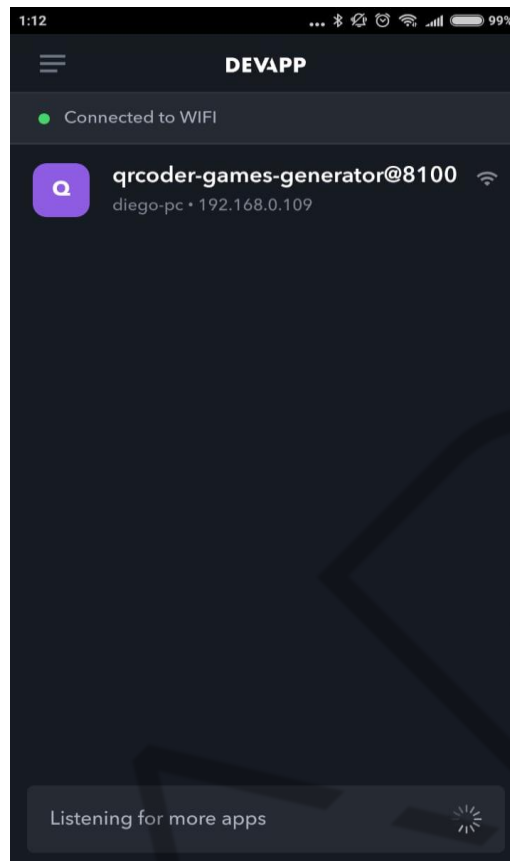


Figura 7.4 – Ionic DevApp pronto para ser vinculado com o projeto  
Fonte: Google Imagens (2018)

## 7.6 Aplicando funcionalidades nativas

Vamos começar a desenvolver!

Agora, aplicaremos a funcionalidade em nosso projeto para permitir a leitura de QRCode que criamos. Para isso, utilizaremos a câmera do *smartphone* para realizar a leitura, mas, antes, adicionaremos um botão no nosso HTML que realizará essa funcionalidade com o seguinte código, no **home.html**:

```
<ion-header>
  <ion-navbar>
    <ion-title>
      Lista de jogos
    </ion-title>
  </ion-navbar>
</ion-header>

<ion-content>
  <ion-list>
    <ion-item *ngFor="let game of games;let i = index"
      (click)="openDetail(game, i)">
```

```

        <ion-thumbnail item-start>
            <ngx-qr-code [qrc-
value]="convertObjectToString(game)"></ngx-qr-code>
        </ion-thumbnail>
        <h2>{{ game.name }}</h2>
        <p>{{ convertPlatformToString(game.platform) }}</p>
    </ion-item>
</ion-list>

<ion-fab bottom right>
    <button ion-fab mini (click)="openDetail()"><ion-icon
name="add"></ion-icon></button>
    <button ion-fab mini (click)="openCamera()"><ion-icon
name="camera"></ion-icon></button>
</ion-fab>
</ion-content>

```

Código-fonte 7.5 – Implementando botão para abrir a câmera no home.html

Fonte: Elaborado pelo autor (2018)

No capítulo anterior, adicionamos o plugin Cordova, que será responsável por realizar a leitura, porém, se você ainda não o tiver instalado no seu projeto, adicione-o, com o seguinte comando:

```

ionic cordova plugin add phonegap-plugin-
barcodescanner
npm install --save @ionic-native/barcode-scanner

```

Código-fonte 7.6 – Instalando o plugin Cordova em nosso projeto e seu componente no Ionic-Native

Fonte: Elaborado pelo autor (2018)

Por fim, implemente o seguinte código no **home.ts**:

```

import { Component } from '@angular/core';
import { NavController, NavParams, AlertController } from
'ionic-angular';
import { DetailPage } from '../detail/detail';

import { BarcodeScanner } from '@ionic-native/barcode-
scanner';
import { GamesProvider } from '../../providers/games/games';
@Component({
    selector: 'page-home',
    templateUrl: 'home.html'
})
export class HomePage {
    games: object[] = []
    constructor(
        public barcodeScanner: BarcodeScanner,
        public gamesProvider: GamesProvider,
        public alertCtrl: AlertController,
        public navCtrl: NavController,

```

```
public navParams: NavParams) {
  this.games = this.gamesProvider.games
}
convertObjectToString(obj) {
  return JSON.stringify(obj)
}

convertPlatformToString(platform) {
  return platform.join(' - ')
}
openDetail(game, index) {
  this.navCtrl.push(DetailPage, { game, index })
}
openCamera() {
  this.barcodeScanner.scan().then(barcodeData => {
    const alert = this.alertCtrl.create({
      title: 'Dados do QrCode',
      message: barcodeData.text
    })
    alert.present()
  }, (err) => {
    console.log('Error: ', err);
  });
}
```

Código-fonte 7.7 – Implementando a scanner no home.ts  
Fonte: Elaborado pelo autor (2018)

No método **openCamera()**, implementamos o serviço **barcodeScanner**, que instalamos em nosso projeto. Nele, chamamos o método *scan*, o qual retornará uma Promise com os dados do QR Code.

No retorno da Promise caso de sucesso, criamos uma caixa de alerta com o serviço **AlertController** do Ionic e apresentamos na tela.

Com tudo isso implementado, ao clicar no botão que inserimos, será aberta nossa câmera com o scanner do QR Code!

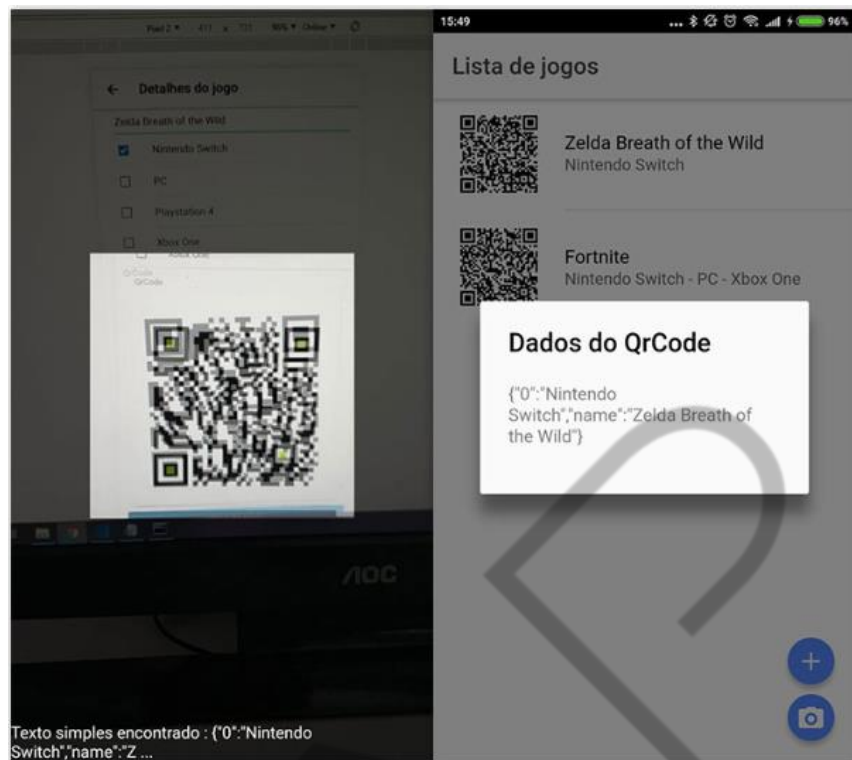


Figura 7.5 – Ilustração da implementação  
Fonte: Google Imagens (2018)

## CONCLUSÃO

Este capítulo é um complemento do nosso projeto. Como foi mostrado, em aplicações híbridas, com apenas poucas linhas de códigos, você pode ter acesso a recursos nativos, facilmente, em ambas as plataformas ao mesmo tempo (iOS e Android).

Chegamos ao final do curso. Tenho plena convicção de que você está preparado para se aventurar no desenvolvimento de microsserviços e de aplicações para mobile.

Divirta-se!

## REFERÊNCIAS

DEVELOPERS. **Android Studio provides the fastest tools for building apps on every type of Android device.** (s.d.). Disponível em: <<https://developer.android.com/studio/>>. Acesso em: 10 out. 2019.

IONIC ACADEMY. **Ionic QR Code Generator & Reader [v3]**. 2017. Disponível em: <<https://ionicacademy.com/ionic-qr-code-generator-reader/>>. Acesso em: 10 out. 2019.

