



# Curso de Programação Mobile Básico

## Passo-a-passo do App



# Índice

TÍTULO	PÁGINA
1. <a href="#">Instalação do NodeJS, SDK do Android e Java Development Kit (JDK)</a>	<a href="#">3</a>
2. <a href="#">Criar o projeto</a>	<a href="#">4</a>
3. <a href="#">Iniciando a Criação da Estrutura de um aplicativo que usa TypeScript</a>	<a href="#">5</a>
4. <a href="#">Criando a página de Login e o provider de autorização</a>	<a href="#">7</a>
5. <a href="#">Começar pelo nosso provider de autorização, nele teremos um login temporário dessa forma:</a>	<a href="#">8</a>
6. <a href="#">Criação da página de registro e alterações no Provider de autorização para registro via Web API</a>	<a href="#">16</a>
7. <a href="#">Exportar para o seu dispositivo em modo debug (testes sem passar pela playstore) e também gerar a release e a keystore para subir o aplicativo na PlayStore como Alfa</a>	<a href="#">44</a>



## 1. Instalação do NodeJS, SDK do Android e Java Development Kit (JDK)

[Guia para o Mac](#)

[Guia para o Windows](#)

Ou...

Faça o download do [Android SDK](#) diretamente do site do [Android](#). Descompacte o arquivo em uma pasta da sua preferencia.

Agora, configure a variável de ambiente **ANDROID\_HOME**, para que os comando dos **Android SDK** fiquem disponíveis de qualquer lugar do seu Sistema Operacional.

Todos os comandos abaixo serão digitados no **Console/Terminal**.

### Linux:

**Edite o arquivo /etc/environment:**

```
sudo vi /etc/environment
```

**Adicione a configuração abaixo dentro do arquivo:**

```
ANDROID_HOME=/<pasta da instalação>/android-sdk-linux
```

```
PATH=${PATH}:$ANDROID_HOME/tools:$ANDROID_HOME/platform-tools
```

### Mas OS:

```
export ANDROID_HOME=/<pasta da instalação>/android-sdk-macosx
```

```
export PATH=${PATH}:$ANDROID_HOME/tools:$ANDROID_HOME/platform-tools
```

### Windows:

```
set ANDROID_HOME=C:\<pasta da instalação>\android-sdk-windows
```

```
set PATH=%PATH%;%ANDROID_HOME%\tools;%ANDROID_HOME%\platform-tools
```

Para testarmos se a configuração foi feita corretamente, digite o comando abaixo no **Console/Terminal** e o **Android SDK Manager** será executado.

```
android
```



Se você seguiu o guia acima ou o vídeo e já tem os seguintes programas instalados:

- NodeJS
- SDK do Android
- JDK (Java Development Kit)

Instale o Ionic e o Cordova com as versões que usamos no curso:

```
npm install -g ionic@2.2.1
```

```
npm install -g cordova@7.0.1
```

Em seguida confirme se a versão está corretamente instalada:

```
ionic -v
```

```
cordova -v
```

```
felipe — bash — 123x23
Last login: Wed Sep 20 15:54:15 on ttys000
MacBook-Pro-de-Think-AM:~ felipe$ sudo ionic -v
[Password:
*****
Dependency warning - for the CLI to run correctly,
it is highly recommended to install/upgrade the following:

Install ios-deploy to deploy iOS applications to devices. `npm install -g ios-deploy` (may require sudo)
*****
2.2.1

MacBook-Pro-de-Think-AM:~ felipe$ sudo cordova -v
7.0.1
MacBook-Pro-de-Think-AM:~ felipe$ ]
```

## 2. Criar o projeto

```
ionic start WakatimeThinkAM2 blank --v2 --appname "Wakatime - Think A.M." --id "net.thinkam.wakatimethinkam"
```

**Nota:** Com o lançamento da CLI do Ionic versão 2.2.3, não precisamos mais do parâmetro --v2!



### 3. Iniciando a Criação da Estrutura de um aplicativo que usa TypeScript



**Observação: Caso esteja no MACOSX abra o VSCode como root usando o comando abaixo**

```
sudo "/Applications/Visual Studio Code.app/Contents/MacOS/Electron"
```



Instruções de Aprendizagem:

- a. Notem a pasta src na raiz do projeto
- b. Foram criadas as subpastas:
  - app
  - pages
  - theme
- c. Criamos também os arquivos na raiz da src:
  - [index.html](#)
  - [manifest.json](#)
  - [service-worker.js](#)
- d. Dentro da pasta src/app teremos:
  - [app.component.ts](#)
  - [app.html](#)
  - [app.module.ts](#)
  - [app.scss](#)
  - [main.ts](#)
- e. Dentro da subpasta src/pages criamos a subpasta home



- f. Na subpasta src/pages/home criamos os arquivos:
  - [home.html](#)
  - [home.scss](#)
  - [home.ts](#)
- g. Dentro da subpasta src/theme criamos o arquivo:
  - [variables.scss](#)
- h. Na pasta raiz criamos dois arquivos e os respectivos conteúdos:
  - i. [tsconfig.json](#)



```
{  
  "compilerOptions": {  
    "allowSyntheticDefaultImports": true,  
    "declaration": false,  
    "emitDecoratorMetadata": true,  
    "experimentalDecorators": true,  
    "lib": [  
      "dom",  
      "es2015"  
    ],  
    "module": "es2015",  
    "moduleResolution": "node",  
    "sourceMap": true,  
    "target": "es5"  
  },  
  "include": [  
    "src/**/*.ts"  
  ],  
  "exclude": [  
    "node_modules"  
  ],  
  "compileOnSave": false,  
  "atom": {  
    "rewriteTsconfig": false  
  }  
}
```



- ii. [tslint.json](#)
- i. No arquivo [package.json](#) na seção devDependencies foram adicionadas as packages a seguir:



```
"@ionic/app-scripts": "2.1.4",
"typescript": "2.3.4"
```

## 4. Criando a página de Login e o provider de autorização

ionic g provider authService  
ionic g page login

Remova o arquivo login.module.ts que foi criado na pasta login dentro de src/pages

Altere o nome da class do arquivo login.ts dentro de src/pages de **Login** para **LoginPage** conforme evidencias abaixo.

De:

```
export class Login {
```

Para:

```
export class LoginPage {
```

Acrecentar

```
import { NavController, NavParams } from 'ionic-angular';
```

Retirar

```
$IMPORTSTATEMENT
$IONICPAGE
```

Vamos acrecentar em nosso app.module.ts que está localizado na pasta src/app:

Imports

```
import { AuthService } from './providers/auth-service';
```

Providers

```
AuthService
```

Ainda dentro do arquivo app.module.ts teremos que acrecentar o novo componente de login da seguinte maneira:

Imports

```
import { LoginPage } from './pages/login/login';
```



## Declarations

LoginPage

## EntryComponents

LoginPage

Já no arquivo app.component.ts em src/app teremos que acrescentar o seguinte:

### Imports

```
import { LoginPage } from './pages/login/login';
```

E alterar a linha de:

```
rootPage:any = HomePage;
```

Para o seguinte:

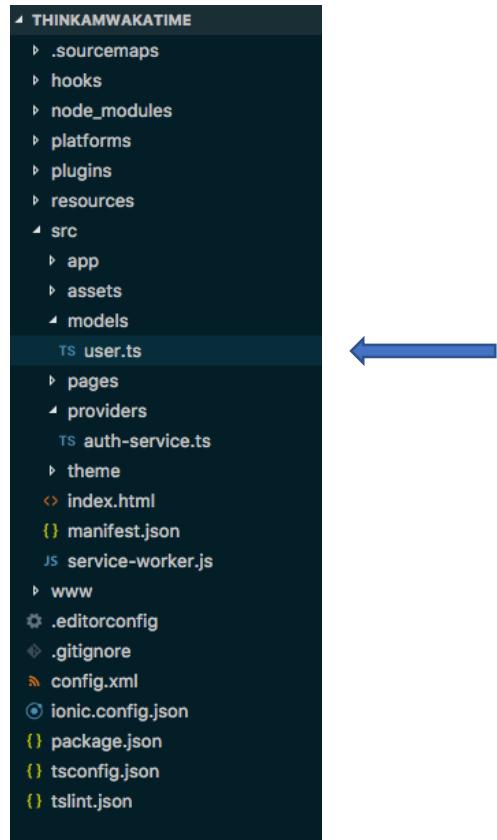
```
rootPage:any = LoginPage;
```

O resultado é que agora abrimos a tela de login, porém ainda não é a tela de login que imaginamos. Para isso vamos...

## 5. Começar pelo nosso provider de autorização, nele teremos um login temporário dessa forma:

```
1 import { Injectable } from '@angular/core';
2 import { Http } from '@angular/http';
3 import 'rxjs/add/operator/map';
4
5 /**
6  * Generated class for the AuthService provider.
7  *
8  * See https://angular.io/guide/dependency-injection for more info on providers
9  * and Angular DI.
10 */
11 @Injectable()
12 export class AuthService {
13
14     constructor(public http: Http) {
15         console.log('Hello AuthService Provider');
16     }
17
18     currentUser: User;
19
20     public login(credentials) {
21         if (credentials.email === null || credentials.password === null) {
22             return Observable.throw("Por favor, informe suas credenciais");
23         } else {
24             return Observable.create(observer => {
25                 // At this point make a request to your backend to make a real check!
26                 let access = (credentials.password === "senhaz" && credentials.email === "email");
27                 this.currentUser = new User('Felipe Almeida', 'felipe.almeida@thinkam.net');
28                 observer.next(access);
29                 observer.complete();
30             });
31         }
32     }
33     public getUserInfo(): User {
34         return this.currentUser;
35     }
36     public logout() {
37         return Observable.create(observer => {
38             this.currentUser = null;
39             observer.next(true);
40             observer.complete();
41         });
42     }
43 }
44
```

Notem que estamos referenciando um modelo que foi inserido na pasta que criaremos, pasta chamada de models e o arquivo é o user.ts, veja:



Dentro dele teremos o seguinte conteúdo:

```
export class User {  
  name: string;  
  email: string;  
  
  constructor(name: string, email: string) {  
    this.name = name;  
    this.email = email;  
  }  
}
```



Agora sim, o nosso provider de autorização será da seguinte forma:

```
import { Injectable } from '@angular/core';
import { Observable } from 'rxjs/Observable';

import 'rxjs/add/operator/map';

import { User } from '../models/user';

@Injectable()
export class AuthService {
  currentUser: User;

  public login(credentials) {
    if (credentials.email === null || credentials.password === null) {
      return Observable.throw("Por favor, informe suas credenciais");
    } else {
      return Observable.create(observer => {
        // At this point make a request to your backend to make a real check!
        let access = (credentials.password === "senha" && credentials.email === "email");
        this.currentUser = new User('Felipe Almeida', 'felipe.almeida@thinkam.net');
        observer.next(access);
        observer.complete();
      });
    }
  }

  public getUserInfo(): User {
    return this.currentUser;
  }

  public logout() {
    return Observable.create(observer => {
      this.currentUser = null;
      observer.next(true);
      observer.complete();
    });
  }
}
```

- Por enquanto temos:
  - Um modelo (user.ts)
  - E nosso provedor de autorização (authService.ts)



Agora resta realizar a chamada deste provedor que usa nosso modelo:

```
1 import { Component } from '@angular/core';
2 -import { NavController, NavParams } from 'ionic-angular';
3
4
5 /**
6  * Generated class for the Login page.
7  *
8  * See https://ionicframework.com/docs/components/#navigation for more info on
9  * Ionic pages and navigation.
10 */
11
12 @Component({
13   selector: 'page-login',
14   templateUrl: 'login.html',
15 })
16 export class LoginPage {
17
18   constructor(public navCtrl: NavController, public navParams: NavParams) {
19
20     ionViewDidLoad() {
21       console.log('ionViewDidLoad LoginPage');
22     }
23   }
24
25 }
26
```

```
1 import { Component } from '@angular/core';
2 +import { NavController, AlertController, LoadingController, Loading, IonicPage } from 'ionic-angular';
3
4 +import { AuthService } from '../../../../../providers/auth-service';
5
6 import { HomePage } from './home/home';
7 +
8 /**
9  * Generated class for the Login page.
10 *
11 * See https://ionicframework.com/docs/components/#navigation for more info on
12 * Ionic pages and navigation.
13 */
14
15 @Component({
16   selector: 'page-login',
17   templateUrl: 'login.html',
18 })
19 export class LoginPage {
20   loading: Loading;
21   registerCredentials = { email: '', password: '' };
22
23   constructor(private navCtrl: NavController, private auth: AuthService,
24             private alertCtrl: AlertController, private loadingCtrl: LoadingController) {
25
26   }
27
28   public login() {
29     this.showLoading();
30     this.auth.login(this.registerCredentials).subscribe(allowed => {
31       if (allowed) {
32         this.navCtrl.setRoot(HomePage);
33       } else {
34         this.showError("Acesso não permitido");
35       }
36     });
37     error => {
38       this.showError(error);
39     });
40   }
41
42   showLoading() {
43     this.loading = this.loadingCtrl.create({
44       content: 'Please wait...',
45       dismissOnPageChange: true
46     });
47     this.loading.present();
48   }
49
50   showError(text) {
51     this.loading.dismiss();
52
53     let alert = this.alertCtrl.create({
54       title: 'Falha ao Entrar',
55       subTitle: text,
56       buttons: ['OK']
57     });
58     alert.present();
59   }
60 }
```



Portanto, esta é a nossa classe do componente de Login:

```
import { Component } from '@angular/core';
import { NavController, AlertController, LoadingController, Loading, IonicPage } from 'ionic-angular';
import { AuthService } from '../../providers/auth-service';
import { HomePage } from '../home/home';

@Component({
  selector: 'page-login',
  templateUrl: 'login.html',
})
export class LoginPage {
  loading: Loading;
  registerCredentials = { email: "", password: "" };
  constructor(private nav: NavController, private auth: AuthService,
    private alertCtrl: AlertController, private loadingCtrl: LoadingController) {}

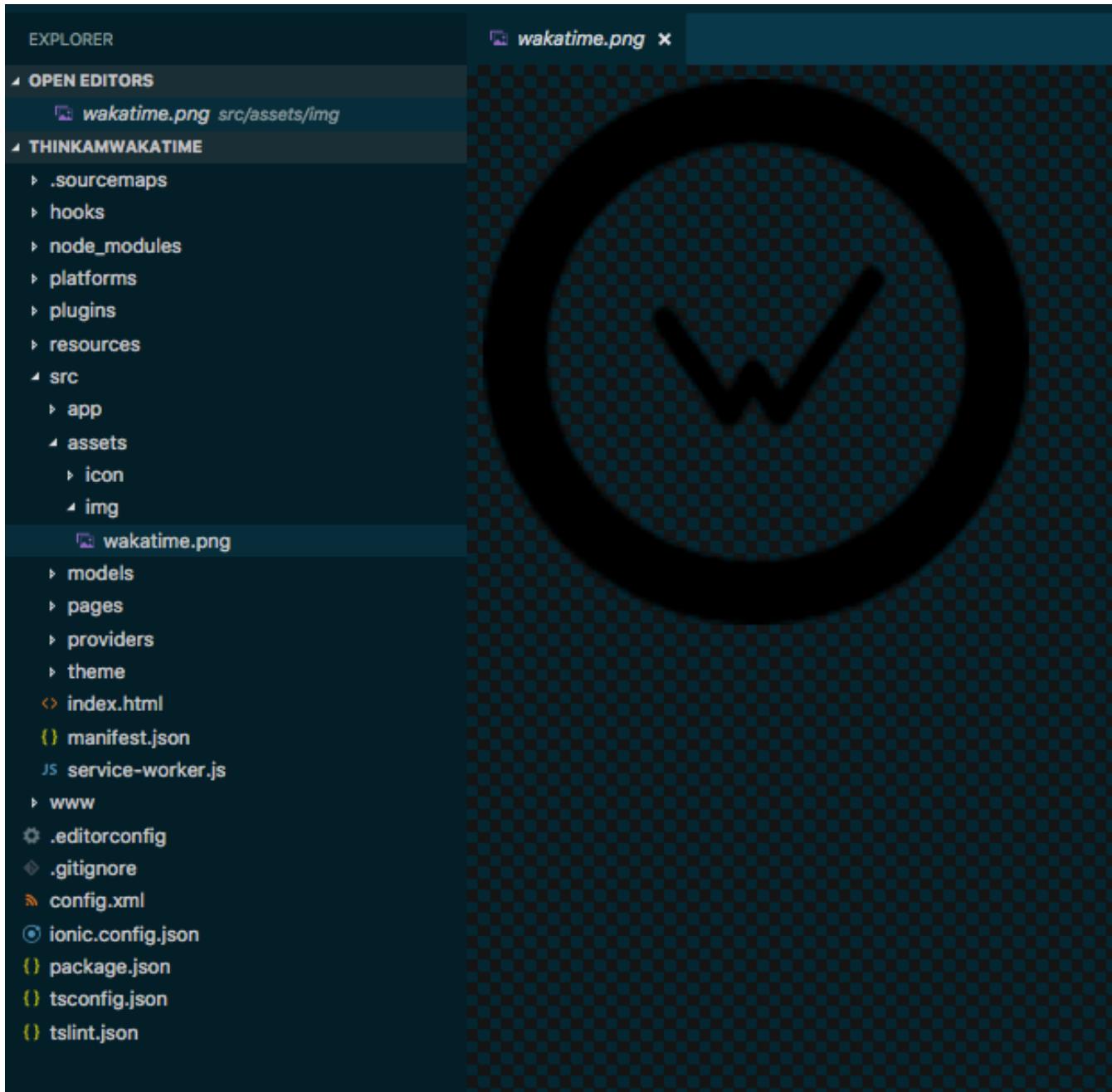
  public login() {
    this.showLoading()
    this.auth.login(this.registerCredentials).subscribe(allowed => {
      if (allowed) {
        this.nav.setRoot(HomePage);
      } else {
        this.showError("Acesso não permitido");
      }
    },
    error => {
      this.showError(error);
    });
  }

  showLoading() {
    this.loading = this.loadingCtrl.create({
      content: 'Please wait...',
      dismissOnPageChange: true
    });
    this.loading.present();
  }

  showError(text) {
    this.loading.dismiss();
    let alert = this.alertCtrl.create({
      title: 'Falha ao Entrar',
      subTitle: text,
      buttons: ['OK']
    });
    alert.present();
  }
}
```



Por fim, precisamos de um HTML, mas antes vamos precisar de uma imagem do Wakatime, coloquei um png que está junto com os anexos e se chama wakatime.png, ele ficou na pasta src/assets, lá dentro criei uma pasta img para colocarmos lá dentro nosso PNG:





Bacana!! Agora só falta o HTML para chegarmos no resultado final em relação a layout e esboço de funcionalidade:

```
1 <!--
2 | Generated template for the Login page.
3 |
4 | See http://ionicframework.com/docs/components/#navigation for more info on
5 | Ionic pages and navigation.
6 -->
7 <ion-header>
8 |
9 <ion-navbar>
10 |   ->ion-title>login</ion-title>
11 </ion-navbar>
12 </ion-header>
13
14
15 <ion-content padding>
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
```

```
<ion-content class="login-content" padding>
  <ion-row class="logo-row">
    <ion-col></ion-col>
    <ion-col width-67>
      
    </ion-col>
    <ion-col></ion-col>
  </ion-row>
  <div class="login-box">
    <form (ngSubmit)="login()" #registerForm="ngForm">
      <ion-row>
        <ion-col>
          <ion-list inset>
            <ion-item>
              <ion-input type="text" placeholder="Email" name="email" [(ngModel)]="registerCredentials.email" required></ion-input>
            </ion-item>
            <ion-item>
              <ion-input type="password" placeholder="Password" name="password" [(ngModel)]="registerCredentials.password" required></ion-input>
            </ion-item>
          </ion-list>
        </ion-col>
      </ion-row>
      <ion-row>
        <ion-col class="signup-col">
          <button ion-button class="submit-btn" full type="submit" [disabled]="!registerForm.form.valid">Login</button>
        </ion-col>
      </ion-row>
    </form>
  </div>
</ion-content>
```



Segue o código sem comparação com o anterior:

```
<ion-content class="login-content" padding>
<ion-row class="logo-row">
<ion-col></ion-col>
<ion-col width-67>

</ion-col>
<ion-col></ion-col>
</ion-row>
<div class="login-box">
<form (ngSubmit)="login()" #registerForm="ngForm">
<ion-row>
<ion-col>
<ion-list inset>

<ion-item>
<ion-input type="text" placeholder="Email" name="email" [(ngModel)]="registerCredentials.email" required></ion-input>
</ion-item>

<ion-item>
<ion-input type="password" placeholder="Password" name="password" [(ngModel)]="registerCredentials.password" required>
</ion-input>
</ion-item>

</ion-list>
</ion-col>
</ion-row>

<ion-row>
<ion-col class="signup-col">
<button ion-button class="submit-btn" full type="submit" [disabled]="!registerForm.form.valid">Login</button>
</ion-col>
</ion-row>
</form>
</div>
</ion-content>
```



## 6. Criação da página de registro e alterações no Provider de autorização para registro via Web API

`ionic g page register`

Remova o arquivo `register.module.ts` que foi criado na pasta `register` dentro de `src/pages`

Altere o nome da class do arquivo `register.ts` dentro de `src/pages` de `Register` para `RegisterPage` conforme evidencias abaixo.

De:

```
export class Register {
```

Para:

```
export class RegisterPage {
```

Retirar:

```
$IMPORTSTATEMENT  
$IONICPAGE
```

Nas primeiras linhas da class adicione:

```
export class RegisterPage {  
  createSuccess = false;  
  registerCredentials = { name: "", email: "", password: "", secretAPIKey: "" };
```

Imports para acrescentar:

```
import { NavController, AlertController, LoadingController, Loading, IonicPage } from 'ionic-angular';  
import { AuthService } from '../../providers/auth-service';
```

Alterar Construtor para:

```
constructor(private nav: NavController, private auth: AuthService,  
           private alertCtrl: AlertController, private loadingCtrl: LoadingController) {  
}
```

Retirar o método padrão:

```
ionViewDidLoad() {  
  console.log('ionViewDidLoad Register');  
}
```



Acrescentar os seguintes métodos:

```
public register() {
  this.auth.register(this.registerCredentials).subscribe(success => {
    if (success) {
      this.createSuccess = true;
      this.showPopup("Sucesso", "Conta Think A.M. Criada!");
    } else {
      this.showPopup("Erro", "Ocorreu um problema na criação da conta.");
    }
  },
  error => {
    this.showPopup("Erro", error);
  });
}

showPopup(title, text) {
  let alert = this.alertCtrl.create({
    title: title,
    subTitle: text,
    buttons: [
      {
        text: 'OK',
        handler: data => {
          if (this.createSuccess) {
            this.nav.popToRoot();
          }
        }
      ]
    ],
    });
  alert.present();
}
```

No app.module.ts vamos ter as seguintes inserções:

### Imports

```
import { RegisterPage } from './pages/register/register';
```

### Declarations

```
RegisterPage
```

### EntryComponents

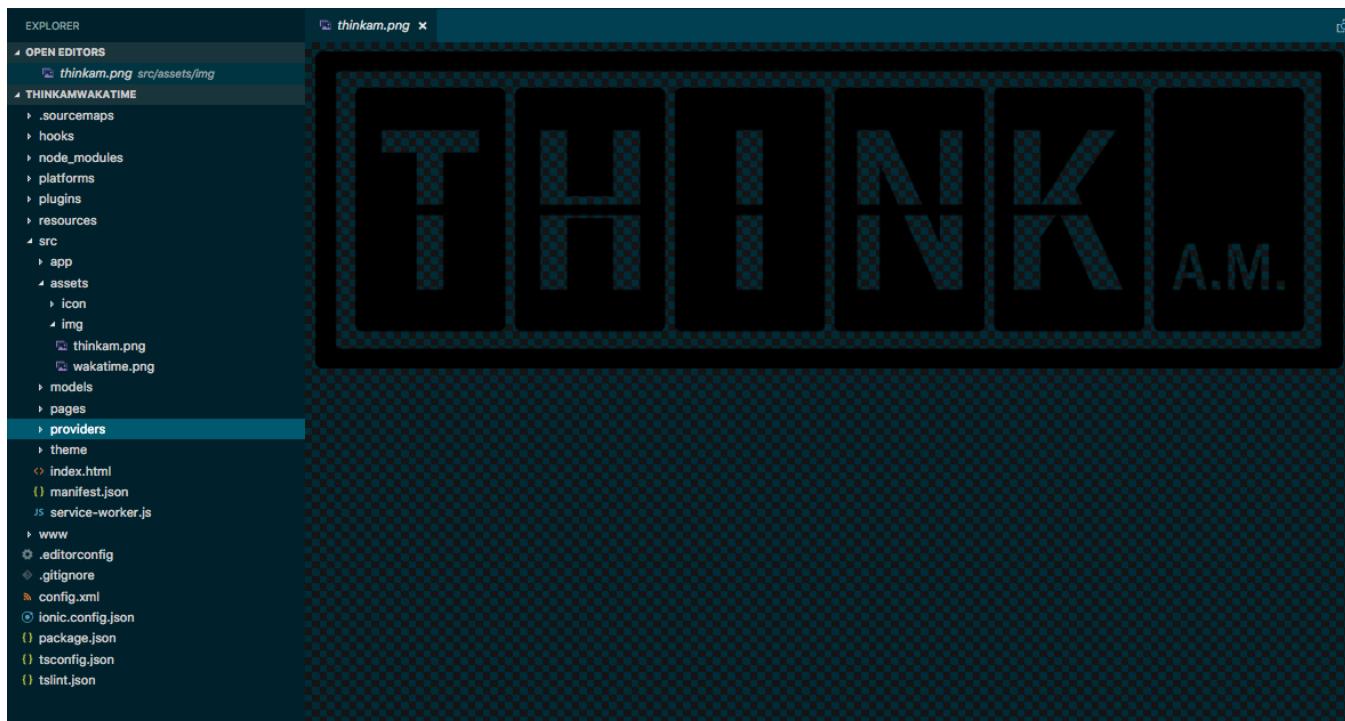
```
RegisterPage
```



Lá no AuthService em providers/auth-service.ts vamos criar o método register temporário até criarmos a chamada pra API da Think A.M.:

```
public register(credentials) {
  if (credentials.name === null ||
    credentials.email === null ||
    credentials.password === null ||
    credentials.secretAPIKey === null) {
    return Observable.throw("Por favor informe todas as informações de registro.");
  } else {
    // Aqui vamos chamar nosso backend mais pra frente!
    return Observable.create(observer => {
      observer.next(true);
      observer.complete();
    });
  }
}
```

Em `src/assets/img` vamos acrescentar a logo da Think A.M., pois de acordo com as regras do Wakatime não devemos colocar a logo deles nesse contexto onde o aplicativo está com registro e login da conta Think A.M. e não Wakatime:





**Na página de login além da nova logomarca teremos agora um link que vai nos direcionar para esse novo componente (register):**

```
1 <ion-content class="login-content" padding>
2   <ion-row class="logo-row">
3     <ion-col></ion-col>
4     <ion-col width-67>
5       | 
6     </ion-col>
7     <ion-col></ion-col>
8   </ion-row>
9   <div class="login-box">
10    <form (ngSubmit)="login()" #registerForm="ngForm">
11      <ion-row>
12        <ion-col>
13          <ion-list inset>
14            <ion-item>
15              | <ion-input type="text" placeholder="E-mail" name="email" [(ngModel)]="registerCredentials.email" required>
16            </ion-item>
17            <ion-item>
18              | <ion-input type="password" placeholder="Senha" name="password" [(ngModel)]="registerCredentials.password" required>
19            </ion-item>
20            <ion-item>
21              | <ion-input type="password" placeholder="Senha" name="password" [(ngModel)]="registerCredentials.password" required>
22            </ion-item>
23          </ion-list>
24        </ion-col>
25      </ion-row>
26    </div>
27  </ion-content>
28
29  <ion-row>
30    <ion-col classe="signup-col">
31      | <button ion-button classe="submit-btn" full type="submit" [disabled]="#!registerForm.form.valid">Login</button>
32      | </ion-col>
33    </ion-row>
34  </form>
35 </div>
36 </ion-content>
37
```

```
1 <ion-content class="login-content" padding>
2   <ion-row class="logo-row">
3     <ion-col></ion-col>
4     <ion-col width-67>
5       | 
6     </ion-col>
7     <ion-col></ion-col>
8   </ion-row>
9   <div class="login-box">
10    <form (ngSubmit)="login()" #registerForm="ngForm">
11      <ion-row>
12        <ion-col>
13          <ion-list inset>
14            <ion-item>
15              | <ion-input type="text" placeholder="E-mail" name="email" [(ngModel)]="registerCredentials.email" required>
16            </ion-item>
17            <ion-item>
18              | <ion-input type="password" placeholder="Senha" name="password" [(ngModel)]="registerCredentials.password" required>
19            </ion-item>
20            <ion-item>
21              | <ion-input type="password" placeholder="Senha" name="password" [(ngModel)]="registerCredentials.password" required>
22            </ion-item>
23          </ion-list>
24        </ion-col>
25      </ion-row>
26    </div>
27  </ion-content>
28
29  <ion-row>
30    <ion-col classe="signup-col">
31      | <button ion-button classe="submit-btn" full type="submit" [disabled]="#!registerForm.form.valid">Login</button>
32      | <button ion-button classe="register-btn" type="button" block clear (click)="createAccount()">Crie uma conta na Think A.M.</button>
33    </ion-col>
34  </ion-row>
35 </form>
36 </div>
37 </ion-content>
38
```

Logo

```

```

## Botão para nova conta

```
<button ion-button class="register-btn" type="button" block clear (click)="createAccount()>  
  Crie uma conta na Think A.M.  
</button>
```

**No componente de Login vamos adicionar as seguintes novidades:**

login.ts (Working Tree) ×

```
1 import { Component } from '@angular/core';
2 import { NavController, AlertController, LoadingController, Loading, IonContent } from '@ionic/angular';
3 import { AuthService } from '../../../../../providers/auth-service';
4 import { HomePage } from '../home/home';
5
6 @Component({
7   selector: 'page-login',
8   templateUrl: 'login.html',
9 })
10 export class LoginPage {
11   loading: Loading;
12   registerCredentials = { email: '', password: '' };
13   constructor(private nav: NavController, private auth: AuthService,
14             private alertCtrl: AlertController, private loadingCtrl: LoadingController) {}
15 }
16
17
18 public login() {
19   this.showLoading();
20   this.auth.login(this.registerCredentials).subscribe(allowed => {
21     if (allowed) {
22       this.nav.setRoot(HomePage);
23     } else {
24       this.showError("Acesso não permitido");
25     }
26   });
27 }
28
29
30 public createAccount() {
31   this.nav.push(RegisterPage);
32 }
33
34
35 public login() {
36   this.showLoading();
37   this.auth.login(this.registerCredentials).subscribe(allowed => {
38     if (allowed) {
39       this.nav.setRoot(HomePage);
40     } else {
41       this.showError("Acesso não permitido");
42     }
43   });
44 }
```

**Portanto serão as seguintes inserções:**



## Imports

```
import { RegisterPage } from './register/register';
```

## Métodos

```
public createAccount() {
  this.nav.push(RegisterPage);
}
```

Por enquanto o HTML ainda está simples, por isso teremos que adicionar o seguinte código HTML para que fique de acordo com o que esperamos:

The screenshot shows a mobile application interface for account creation. At the top, there is a header bar with a back arrow icon and the title "Registro - Conta Think A.M.". Below the header, there are four input fields: "Nome" (Name), "E-mail" (Email), "Senha" (Password), and "Secret API Key Wakatime". At the bottom of the screen is a large blue button labeled "ENVIAR" (Send).

```
<ion-header>
```



```
<ion-navbar color="dark">
  <ion-title>Registro - Conta Think A.M.</ion-title>
</ion-navbar>
</ion-header>

<ion-content class="login-content" padding>
  <div class="login-box">

    <form (ngSubmit)="register()" #registerForm="ngForm">
      <ion-row>
        <ion-col>
          <ion-list inset>

            <ion-item>
              <ion-input type="text" placeholder="Nome" name="name" [(ngModel)]="registerCredentials.name" required></ion-input>
            </ion-item>

            <ion-item>
              <ion-input type="text" placeholder="E-mail" name="email" [(ngModel)]="registerCredentials.email" required></ion-input>
            </ion-item>

            <ion-item>
              <ion-input type="password" placeholder="Senha" name="password" [(ngModel)]="registerCredentials.password" required></ion-input>
            </ion-item>

            <ion-item>
              <ion-input type="text" placeholder="Secret API Key Wakatime" name="secretAPIKey" [(ngModel)]="registerCredentials.secretAPIKey" required></ion-input>
            </ion-item>

          </ion-list>
        </ion-col>
      </ion-row>

      <ion-row>
        <ion-col class="signup-col">
          <button ion-button class="submit-btn" full type="submit" [disabled]="!registerForm.form.valid">Enviar</button>
        </ion-col>
      </ion-row>

    </form>
  </div>
</ion-content>
```

Se você quiser se aprofundar melhor nos componentes do Ionic, desde o HTML e CSS até exemplos de uso do componente no Typescript, recomendo a seguinte [página](#).



Vamos agora criar as validações que comentei anteriormente de nome, e-mail, e secret API key. Em seguida vamos subir para o GitHub porque já acumulamos bastante coisa para o nosso changeset.

Criem uma pasta chamada utils em src, dentro da pasta utils crie um arquivo chamado email.ts, o caminho ficará o seguinte, src/utils/email.ts, neste arquivo acrescente o conteúdo que fará a validação de e-mail:

```
import { Injectable } from '@angular/core';

@Injectable()
export class Email {
  validateEmail(email: string) {
    if (/^[\w+([.-]?\w+)*@\w+([.-]?\w+)*(\.\w{2,3})+$/.test(email)) {
      return (true);
    }
    return (false);
  }
}
```

Estamos dessa forma acrescentando um novo módulo que é um provider de utilidades, sendo assim acrescentaremos no app.module.ts o conteúdo a seguir:

#### Imports

```
import { Email } from './utils/email';
```

#### Providers

```
Email
```

No nosso outro provedor AuthService vamos ter novidades:

#### Imports

```
import 'rxjs/add/observable/throw';
import { Email } from './utils/email';
```

Um constructor para que tenhamos a instância do emailUtils através de injeção de dependência:

```
constructor(private emailUtils : Email){  
}
```



E alterações no método register:

```
public register(credentials) {  
    if (credentials.name === null ||  
        credentials.email === null ||  
        credentials.password === null ||  
        credentials.secretAPIKey === null) {  
        return Observable.throw("Por favor informe todas as informações de registro.");  
    } else {  
        if (credentials.name.length < 3){  
            return Observable.throw("Por favor informe um nome com no mínimo 3 caracteres.");  
        }  
        if (!this.emailUtils.validateEmail(credentials.email)){  
            return Observable.throw("Por favor informe um e-mail válido.");  
        }  
        if (credentials.secretAPIKey.length < 8){  
            return Observable.throw("Por favor informe uma secret API key com no mínimo 8 caracteres.");  
        }  
        // Aqui vamos chamar nosso backend mais pra frente!  
        return Observable.create(observer => {  
            observer.next(true);  
            observer.complete();  
        });  
    }  
}
```

Vamos testar e se estiver tudo certo é só subir pro [GitHub!](#) =]

Agora vamos concentrar nossos esforços em testar a API de registro da Think A.M. no Postman, pois bem! Vamos para alguns cenários de inputs...

### Cenário 1:

```
{  
    name: "Felipe A. de Almeida",  
    email: "f.albuquerquedealmeida@gmail.com",  
    password: "bra",  
    secretAPIKey: "6d2855b7-8f79-463e-b1c9-44ed7e280f40"  
}
```

### Cenário 2:

```
{  
    name: "Felipe A. de Almeida",  
    email: "f.albuquerquedealmeida@gmail.com",  
    password: "brasilmobile",  
    secretAPIKey: "6d2855b7-8f79-463e-b1c9-44ed7e280f40"  
}
```

### Cenário 3:



```
{  
    name: "Felipe A. de Almeida",  
    email: "f.albuquerquedealmeida@gmail.com",  
    password: "bras1lmobile",  
    secretAPIKey: "6d2855b7-8f79-463e-b1c9-44ed7e280f40"  
}
```

#### Cenário 4:

```
{  
    name: "Felipe A. de Almeida",  
    email: "f.albuquerquedealmeida@gmail.com",  
    password: "BRAS1LMOBILE",  
    secretAPIKey: "6d2855b7-8f79-463e-b1c9-44ed7e280f40"  
}
```

#### Cenário 5:

```
{  
    name: "Felipe A. de Almeida",  
    email: "f.albuquerquedealmeida@gmail.com",  
    password: "BRAS1LMobile",  
    secretAPIKey: "6d2855b7-8f79-463e-b1c9-44ed7e280f40"  
}
```

#### Cenário 6:

```
{  
    name: "Felipe A. de Almeida",  
    email: "f.albuquerquedealmeida@gmail.com",  
    password: "BRA$1LMobile",  
    secretAPIKey: "6d2855b7-8f79-463e-b1c9-44ed7e280f40"  
}
```

Se você ainda não tem o Postman, adicione a [extensão do chrome](#).



No sexto cenário obtivemos o resultado positivo com a API retornando o ID do objeto que foi inserido:

The screenshot shows a Postman request configuration for a POST method to the URL `thinkam.azurewebsites.net/api/user`. The 'Body' tab is selected, showing a JSON payload:

```
1 {  
2   name: "Felipe A. de Almeida",  
3   email: "f.albuquerquedealmeida@gmail.com",  
4   password: "BRA$1LMobile",  
5   secretAPIKey: "6d2855b7-8f79-463e-b1c9-44ed7e280f40"  
6 }
```

The response body shows the inserted user's ID:

```
1 "07013810-decc-4813-90d1-7600577ea0d5"
```

A red oval highlights the response body area.

Utilizamos o método POST:

The screenshot shows a Postman request configuration for a POST method to the URL `thinkam.azurewebsites.net/api/user`. The 'Body' tab is selected, showing a JSON payload:

```
1 {  
2   name: "Felipe A. de Almeida",  
3   email: "f.albuquerquedealmeida@gmail.com",  
4   password: "BRA$1LMobile",  
5   secretAPIKey: "6d2855b7-8f79-463e-b1c9-44ed7e280f40"  
6 }
```

The response body shows the inserted user's ID:

```
1 "07013810-decc-4813-90d1-7600577ea0d5"
```

Selecionamos na aba Body a opção raw e escolhemos o tipo JSON, onde colamos os cenários para teste da URL (thinkam.azurewebsites.net/api/user):

```

1  {
2      "name": "Felipe A. de Almeida",
3      "email": "f.albuquerquedealmeida@gmail.com",
4      "password": "BRASILMobile",
5      "secretAPIKey": "6d2855b7-8f79-463e-b1c9-44ed7e280f40"
6  }
7
8
9
10
11
12

```

Body    Cookies    Headers (11)    Tests

Pretty    Raw    Preview    JSON

1 "07013810-decc-4813-90d1-7600577ea0d5"

Fiz uma alteração na API para retornar um objeto ao invés de somente uma string (texto), agora teremos de retornar um objeto com dois atributos, o primeiro é o “ok” que poderá ser verdadeiro ou falso, ou seja, true ou false, o segundo atributo é o “data” que trará os dados, que pode ser o ID ou qualquer outra informação. Mas isso não muda muita coisa agora para nós, vamos para o nosso exemplo de código:

#### Serão 4 arquivos alterados:

1. No primeiro é uma questão de linguagem do aplicativo (Português Brasileiro), vamos alterar o arquivo src/index.html de lang="en", para lang="pt-br", conforme abaixo:

```

1 <!DOCTYPE html>
2 <html lang="en" dir="ltr">
3 <head>
4     <meta charset="UTF-8">
5     <title>Ionic App</title>
6     <meta name="viewport" content="viewport-fit=cover, width=device-width">
7     <meta name="format-detection" content="telephone=no">
8     <meta name="msapplication-tap-highlight" content="no">
9
10    <link rel="icon" type="image/x-icon" href="assets/icon/favicon.ico">
11    <link rel="manifest" href="manifest.json">
12    <meta name="theme-color" content="#4e8ef7">
13
14    <!-- cordova.js required for cordova apps -->
15    <script src="cordova.js"></script>
16

```

```

1 <!DOCTYPE html>
2 + <html lang="pt-br" dir="ltr">
3 <head>
4     <meta charset="UTF-8">
5     <title>Ionic App</title>
6     <meta name="viewport" content="viewport-fit=cov
7     <meta name="format-detection" content="telepho
8     <meta name="msapplication-tap-highlight" conter
9
10    <link rel="icon" type="image/x-icon" href="asse
11    <link rel="manifest" href="manifest.json">
12    <meta name="theme-color" content="#4e8ef7">
13
14    <!-- cordova.js required for cordova apps -->
15    <script src="cordova.js"></script>
16

```

2. No segundo arquivo se trata de um novo módulo que estará em nosso aplicativo e será fundamental para realizar as requisições Web API:

```

1 import { BrowserModule } from '@angular/platform-browser';
2 import { ErrorHandler, NgModule } from '@angular/core';
3
4 import { IonicApp, IonicErrorHandler, IonicModule } from 'ionic-angular';
5 import { SplashScreen } from '@ionic-native/splash-screen';
6 import { StatusBar } from '@ionic-native/status-bar';
7
8 import { AuthService } from './providers/auth-service';
9
10 import { Email } from '../utils/email';
11
12 import { MyApp } from './app.component';
13
14 import { HomePage } from '../pages/home/home';
15 import { LoginPage } from '../pages/login/login';
16 import { RegisterPage } from '../pages/register/register';
17
18 @NgModule({
19   declarations: [
20     MyApp,
21     HomePage,
22     LoginPage,
23     RegisterPage
24   ],
25   imports: [
26     BrowserModule,
27     IonicModule.forRoot(MyApp)
28   ],
29   bootstrap: [IonicApp],
30   entryComponents: [
31     HomePage,
32     LoginPage,
33     RegisterPage
34   ],
35   providers: [
36     StatusBar,
37     SplashScreen,
38     { provide: ErrorHandler, useClass: IonicErrorHandler }
39   ]
40 })
41
42 export class AppModule {}

```

```

1 import { BrowserModule } from '@angular/platform-browser';
2 import { ErrorHandler, NgModule } from '@angular/core';
3+ import { HttpClientModule } from '@angular/http';
4+
5 import { IonicApp, IonicErrorHandler, IonicModule } from 'ionic-angular';
6 import { SplashScreen } from '@ionic-native/splash-screen';
7 import { StatusBar } from '@ionic-native/status-bar';
8
9 import { AuthService } from './providers/auth-service';
10
11 import { Email } from '../utils/email';
12
13 import { MyApp } from './app.component';
14
15 import { HomePage } from '../pages/home/home';
16 import { LoginPage } from '../pages/login/login';
17 import { RegisterPage } from '../pages/register/register';
18
19 @NgModule({
20   declarations: [
21     MyApp,
22     HomePage,
23     LoginPage,
24     RegisterPage
25   ],
26   imports: [
27     BrowserModule,
28+   IonicModule.forRoot(MyApp),
29+   HttpClientModule
30   ],
31   bootstrap: [IonicApp],
32   entryComponents: [
33     HomePage,
34     LoginPage,
35     RegisterPage
36   ],
37   providers: [
38     StatusBar,
39     SplashScreen,
40     { provide: ErrorHandler, useClass: IonicErrorHandler }
41   ]
42 })
43
44 export class AppModule {}

```

Veja que adicionamos...

## Imports

```
import { HttpClientModule } from '@angular/http';
```

## HttpClientModule

3. Vamos então para o terceiro arquivo que está em src/pages/register.ts que será responsável por intermediar as interações do usuário no HTML e as requisições e respostas da Web API no provedor AuthService:

```

1 import { Component } from '@angular/core';
2 import { NavController, AlertController, LoadingController, Loading, IonicPage } from 'ionic-angular';
3 import { AuthService } from './providers/auth-service';
4
5 @Component({
6   selector: 'page-register',
7   templateUrl: 'register.html',
8 })
9 export class RegisterPage {
10   createSuccess = false;
11   registerCredentials = { name: '', email: '', password: '', secretAPIKey: '' };
12
13   constructor(private nav: NavController, private auth: AuthService,
14   | private alertCtrl: AlertController, private loadingCtrl: LoadingController) {
15   }
16
17   public register() {
18     this.auth.register(this.registerCredentials).subscribe(success => {
19       if (success) {
20         this.createSuccess = true;
21         this.showPopup("Sucesso", "Conta Think A.M. Criada!");
22       } else {
23         this.showPopup("Erro", "Ocorreu um problema na criação da conta.");
24       }
25     },
26     error => {
27       this.showPopup("Erro", error);
28     });
29   }
30 }

```

```

1 import { Component } from '@angular/core';
2 import { NavController, AlertController, LoadingController, Loading, IonicPage } from 'ionic-angular';
3 import { AuthService } from './providers/auth-service';
4
5 @Component({
6   selector: 'page-register',
7   templateUrl: 'register.html',
8 })
9 export class RegisterPage {
10   createSuccess = false;
11   registerCredentials = { name: '', email: '', password: '', secretAPIKey: '' };
12
13   constructor(private nav: NavController, private auth: AuthService,
14   | private alertCtrl: AlertController, private loadingCtrl: LoadingController) {
15   }
16
17   public register() {
18+     this.auth.register(this.registerCredentials).subscribe(res => {
19+       if (res.ok) {
20+         this.createSuccess = res.ok;
21+         this.showPopup("Sucesso", "Conta Think A.M. Criada!");
22+       } else {
23+         this.showPopup("Erro", res.data);
24+       }
25     },
26     error => {
27       this.showPopup("Erro", error);
28     });
29   }
30 }

```

Neste caso precisamos alterar porque mudei o retorno da API para os dois atributos ao invés de só nos retornar um texto (string).



- O quarto e último, mas não menos importante, está em `src/providers/auth-service.ts` e serve para realizar a comunicação com nossa Web API, que neste caso está em Azure Functions, não vamos aprender a criar uma API neste curso, mas é importante vocês saberem que para um aplicativo moderno a API é indispensável.

```
1 import { Injectable } from '@angular/core';
2 import { Observable } from 'rxjs/Observable';
3
4 import 'rxjs/add/operator/map';
5 import 'rxjs/add/observable/throw';
6
7 import { User } from '../models/user';
8
9 import { Email } from '../utils/email';
10
11 @Injectable()
12 export class AuthService {
13
14   constructor(private emailUtils : Email){}
15
16   currentUser: User;
17
18   currentUser$: User;
19
20   public login(credentials) {
21     if(credentials.name === null || credentials.password === null) {
22       return Observable.throw("Por favor, informe suas credenciais!");
23     } else {
24       return Observable.create(observer => {
25         //At this point make a request to your backend to make a real check!
26         let access = [credentials.name === "senha" && credentials.email === "email"];
27         this.currentUser = new User('Felipe Almeida', 'felipe.almeida@thinkkan.net');
28         observer.next(access);
29         observer.complete();
30       });
31     }
32   }
33
34   public register(credentials) {
35     if(credentials.name === null || credentials.email === null || credentials.password === null || credentials.secretAPIKey === null) {
36       return Observable.throw("Por favor informe todas as informações de registro.");
37     } else if(credentials.name.length < 3) {
38       return Observable.throw("Por favor informe um nome com no mínimo 3 caracteres.");
39     } else if(!this.emailUtils.validateEmail(credentials.email)){
40       return Observable.throw("Por favor informe um e-mail válido.");
41     }
42     if(credentials.secretAPIKey.length < 8){
43       return Observable.throw("Por favor informe uma secret API key com no mínimo 8 caracteres.");
44     }
45
46     //Após vossa classe nosso backend manda pra frente!
47     return Observable.create(observer => {
48       observer.next(true);
49       observer.complete();
50     });
51   }
52
53   public logout() {
54     this.currentUser$.next(null);
55     this.currentUser$.complete();
56   }
57
58   public login(credentials) {
59     if(credentials.name === null || credentials.password === null) {
60       return Observable.throw("Por favor, informe suas credenciais!");
61     } else {
62       return Observable.create(observer => {
63         //At this point make a request to your backend to make a real check!
64         let access = [credentials.name === "senha" && credentials.email === "email"];
65         this.currentUser = new User('Felipe Almeida', 'felipe.almeida@thinkkan.net');
66         observer.next(access);
67         observer.complete();
68       });
69     }
70   }
71
72   public register(credentials) {
73     if(credentials.name === null || credentials.email === null || credentials.password === null || credentials.secretAPIKey === null) {
74       return Observable.throw("Por favor informe todas as informações de registro.");
75     } else if(credentials.name.length < 3) {
76       return Observable.throw("Por favor informe um nome com no mínimo 3 caracteres.");
77     } else if(!this.emailUtils.validateEmail(credentials.email)){
78       return Observable.throw("Por favor informe um e-mail válido.");
79     }
80     if(credentials.secretAPIKey.length < 8){
81       return Observable.throw("Por favor informe uma secret API key com no mínimo 8 caracteres.");
82     }
83
84     return this.http.post('http://thinkkan.guruwebsites.net/api/user', credentials)
85       .map(res => res.json())
86       .catch(error => Observable.throw(error.json()));
87   }
88 }
```

## Imports

```
import { Http } from '@angular/http';
```

```
import 'rxjs/add/operator/catch';
```

## Constructor

```
private http: Http){
```

## Requisição

```
return this.http.post("http://thinkam.azurewebsites.net/api/user", credentials)
    .map(res => res.json())
    .catch(error => Observable.throw(error.json()));
```

Muito, muito, muito simples! Vamos testar e se estiver ok, vamos subir para o GitHub!

Notamos que está faltando um detalhe importante para esses casos de consumo de API e que faz a diferença para a qualidade de usabilidade do usuário, não sei se vocês notaram antes, mas temos o aguarde no exemplo inicial do Login, e vamos melhorá-lo e relembrá-lo para que possa ser utilizado no componente de registro.



Vejam as melhorias no componente de Login:

```
1 import { Component } from '@angular/core';
2 import { NavController, AlertController, LoadingController, Loading, IonicPage } from 'ionic-angular';
3
4 import { AuthService } from '../../../../../providers/auth-service';
5
6 import { HomePage } from '../home/home';
7 import { RegisterPage } from '../register/register';
8
9 @Component({
10   selector: 'page-login',
11   templateUrl: 'login.html',
12 })
13 export class LoginPage {
14   loading: Loading;
15   registerCredentials = { email: '', password: '' };
16   constructor(private nav: NavController, private auth: AuthService,
17     private navCtrl: AlertController, private loadingCtrl: LoadingController) {
18   }
19
20   public createAccount() {
21     this.navCtrl.push(RegisterPage);
22   }
23
24   public login() {
25     this.showLoading();
26     this.auth.login(this.registerCredentials).subscribe(allowed => {
27       if (allowed) {
28         this.navCtrl.setRoot(HomePage);
29       } else {
30         this.showError("Acesso não permitido");
31       }
32     },
33     error => {
34       this.showError(error);
35     });
36   }
37
38   showLoading() {
39     this.loading = this.loadingCtrl.create({
40       content: 'Please wait...',
41       dismissOnPageChange: true
42     });
43     this.loading.present();
44   }
45
46   showError(text) {
47     this.loading.dismiss();
48     let alert = this.alertCtrl.create({
49       title: 'Falha ao Entrar',
50       subTitle: text,
51       buttons: ['OK']
52     });
53     alert.present();
54   }
55 }
```

```
1 import { Component } from '@angular/core';
2 import { NavController, AlertController, LoadingController, Loading, IonicPage } from 'ionic-angular';
3
4 import { AuthService } from '../../../../../providers/auth-service';
5
6 import { HomePage } from '../home/home';
7 import { RegisterPage } from '../register/register';
8
9 @Component({
10   selector: 'page-login',
11   templateUrl: 'login.html',
12 })
13 export class LoginPage {
14   loading: Loading;
15   registerCredentials = { email: '', password: '' };
16   constructor(private nav: NavController,
17     private auth: AuthService,
18     private navCtrl: AlertController,
19     private loadingCtrl: LoadingController) {
20   }
21
22   public createAccount() {
23     this.navCtrl.push(RegisterPage);
24   }
25
26   public login() {
27     this.showLoading();
28     this.auth.login(this.registerCredentials).subscribe(allowed => {
29       if (allowed) {
30         this.navCtrl.setRoot(HomePage);
31       } else {
32         this.showError("Acesso não permitido");
33       }
34     },
35     error => {
36       this.showError(error);
37     });
38   }
39   showLoading() {
40     this.loading = this.loadingCtrl.create({
41       content: 'Por favor aguarde...',
42       dismissOnPageChange: true
43     });
44     this.loading.present();
45   }
46
47   showError(text) {
48     this.loading.dismiss();
49     let alert = this.alertCtrl.create({
50       title: 'Falha ao Entrar',
51       subTitle: text,
52       buttons: ['OK']
53     });
54     alert.present();
55   }
56 }
```

- Organizamos melhor o constructor para uma melhor leitura do que está sendo instanciado através de injeção de dependência;
- E também realizamos a tradução da mensagem que estava em inglês, já que nosso aplicativo está sendo criado inicialmente unicamente para Brasileiros.

Vamos aplicar esse conceito no register, organizando melhor o constructor e adicionando aquilo que o login já vinha fazendo, porém no caso do register já existe uma comunicação com a Web API da Think A.M. que está no Azure Functions.

```
1 export class RegisterPage {
2
3   createSuccess = false;
4   registerCredentials = { name: '', email: '', password: '', secretAPIKey: '' };
5
6   constructor(private navCtrl: NavController, private auth: AuthService,
7     private navCtrl: AlertController, private loadingCtrl: LoadingController) {
8   }
9
10  public register() {
11    this.auth.register(this.registerCredentials).subscribe(success => {
12      if (success) {
13        this.createSuccess = true;
14
15        this.showPopup("Sucesso", "Conta Think A.M. Criada!");
16      } else {
17        this.showPopup("Erro", "Ocorreu um problema na criação da conta.");
18
19      }
20    },
21    error => {
22      this.showPopup("Erro", error);
23    });
24  }
25
26  showLoading() {
27    this.loading = this.loadingCtrl.create({
28      content: 'Por favor aguarde...',
29      dismissOnPageChange: true
30    });
31    this.loading.present();
32  }
33
34  showPopup(title, message) {
35    let alert = this.alertCtrl.create({
36      title: title,
37      subTitle: message,
38      buttons: ['OK']
39    });
40    alert.present();
41  }
42
43  showError(text) {
44    this.loading.dismiss();
45    let alert = this.alertCtrl.create({
46      title: 'Falha ao Entrar',
47      subTitle: text,
48      buttons: ['OK']
49    });
50    alert.present();
51  }
52
53  
```



Notem que estamos acrescentando o dismiss() que “mata” o carregando pra nós.

Agora será que conseguimos subir para o servidor do GitHub? Opa, [claro!](#)

## 5. Login usando a API da Think A.M.

Incialmente fizemos o [básico](#):

```
6 ThinkAMWakatime/src/pages/login/login.ts
@@ -25,11 +25,11 @@ export class LoginPage {
 25   25
 26   26     public login() {
 27   27       this.showLoading()
 28 -     this.auth.login(this.registerCredentials).subscribe(allowed => {
 29 -       if (allowed) {
 28 +     this.auth.login(this.registerCredentials).subscribe(res => {
 29 +       if (res.ok) {
 30   30         this.nav.setRoot(HomePage);
 31   31       } else {
 32 -         this.showError("Acesso não permitido");
 32 +         this.showError(res.data);
 33   33     }
 34   34   },
 35   35     error => {
 36
 36
 37
 37
```

```
10 ThinkAMWakatime/src/providers/auth-service.ts
@@ -25,13 +25,9 @@ export class AuthService {
 25   25     if (credentials.email === null || credentials.password === null) {
 26   26       return Observable.throw("Por favor, informe suas credenciais");
 27   27     } else {
 28 -       return Observable.create(observer => {
 29 -         // At this point make a request to your backend to make a real check!
 30 -         let access = (credentials.password === "senha" && credentials.email === "email");
 31 -         this.currentUser = new User('Felipe Almeida', 'felipe.almeida@thinkam.net');
 32 -         observer.next(access);
 33 -         observer.complete();
 34 -       });
 28 +       return this.http.post("http://thinkam.azurewebsites.net/api/login", credentials)
 29 +         .map(res => res.json())
 30 +         .catch(error => Observable.throw(error.json()));
 35   31     }
 36   32   }
 37   33
 37
```

Mas existe um detalhe importante que teremos que ter nesse Login que será fundamental mais adiante, que é o armazenamento do nosso Token para uso nas requisições do Wakatime que necessitam de autenticação.



Sendo assim, alteramos a API para retornar no atributo data um objeto contendo o nome e o token:

The screenshot shows the Postman interface with a POST request to `thinkam.azurewebsites.net/api/login`. The Body tab is selected, showing a JSON payload:

```
1 {  
2   email: "f.albuquerquealmeida@gmail.com",  
3   password: "BRA$1LMobile"  
4 }  
5  
6  
7  
8  
9  
10
```

The response body is displayed in the Body tab under the JSON tab, showing a successful response with an object containing "ok" and "data" fields:

```
1 {  
2   "ok": true,  
3   "data": {  
4     "name": "Felipe A. de Almeida",  
5     "token": "NmQyODU1YjctOGY3OS00NjNLLWIxYzktNDRlZDdlMjgwZjQw"  
6   }  
7 }
```

Veja que já está em base64 que é como vamos utiliza-lo dentro do app, só precisamos agora armazenar essa informação no banco de dados do aplicativo.



Para isso teremos que adicionar novos módulos do aplicativo, um deles será feito por nós, que é um provider de acesso a dados locais.

```
ionic g provider dbProvider
```

E adicionar o seguinte conteúdo:

```
import { Injectable } from '@angular/core';
import { Storage } from '@ionic/storage';
import { SQLite, SQLiteObject } from '@ionic-native/sqlite';

@Injectable()
export class DbProvider {
  constructor(private storage: Storage, private sqlite: SQLite) {
    this.storage.length().then((length: number) => {
      console.log(length);
    });
  }

  get(key: string): any {
    return this.storage.get(key);
  }

  set(key: string, value: any) {
    return this.storage.set(key, value);
  }

  remove(key: string) {
    return this.storage.remove(key);
  }
}
```



```
like(key: string) {
  return new Promise((resolve, reject) => {
    let retorno: any[] = [];
    this.storage.keys().then((keys: string[]) => {
      let count: number = 0;

      keys.forEach((keyData: string) => {
        if (keyData.indexOf(key, 0) !== -1) {
          this.storage.get(keyData).then((res: any[]) => {
            if (Array.isArray(res)) {
              res.forEach((resItem: any) => {
                retorno.push(resItem);
              });
            }

            count++;

            if (count >= keys.length)
              resolve(retorno);
          }) else {
            retorno.push(res);

            count++;
            if (count >= keys.length)
              resolve(retorno);
          }
        });
      });
    }) else {
      count++;

      if (count >= keys.length)
        resolve(retorno);
    });
  });
}
```



```
likeRemove(key: string) {
  return new Promise((resolve, reject) => {
    this.storage.keys().then((keys: string[]) => {
      let count: number = 0;

      keys.forEach((keyData: string) => {
        if (keyData.indexOf(key, 0) !== -1) {
          this.remove(keyData);

          count++;
        }

        if (count >= keys.length)
          resolve(true);
      } else {
        count++;
      }

      if (count >= keys.length)
        resolve(true);
    })
    .then(() => {
      this.storage.set(key, null);
    })
    .catch(error => {
      console.error(`Error removing key ${key}: ${error}`);
    });
  });
}

public count<T>(key: string): Promise<number> {
  return this.storage.get(key).then((res: T[]) => {
    try {
      return res.length || 0;
    } catch (error) {
      console.log(error);
      return 0;
    }
  });
}
```

Porém isso só funciona se fizermos as seguintes modificações no package.config:

```
25   "@ionic-native/status-bar": "3.12.1",
26   "@ionic/storage": "2.0.1",
27   "ionic-angular": "3.6.1",
28   "ionicons": "3.0.0",
29   "rxjs": "5.4.0",
30   "sw-toolbox": "3.6.0",
31   "zone.js": "0.8.12"
32 },
33 "devDependencies": {
34   "@ionic/app-scripts": "2.1.4",
35   "typescript": "2.3.4"
36 },
37 "cordovaPlugins": [
38   "cordova-plugin-whitelist",
39   "cordova-plugin-device",
40   "cordova-plugin-console",
41   "cordova-plugin-splashscreen",
42   "cordova-plugin-statusbar",
43   "ionic-plugin-keyboard"
44 ],
45 "cordovaPlatforms": []
46 
```



Ou seja, adicionar as dependências:

- "@ionic-native/sqlite": "3.12.1",
- "@ionic/app-scripts": "2.1.4",

E o plugin de acesso a esse recurso no dispositivo:

"cordova-sqlite-storage"

Após feito isso, use o comando **npm install** no seu terminal integrado do VSCode ou no terminal/cmd do seu sistema operacional (dentro da pasta do projeto).

Fizemos tudo isso com o propósito de gravar o novo retorno da API (token e nome) no nosso banco de dados do celular. E quando adicionamos novos módulos temos que alterar nosso **app.module.ts**, por isso vamos as seguintes alterações:

```
5 import { IonicApp, IonicErrorHandler, IonicModule } from 'ionic-angular';
6 import { SplashScreen } from '@ionic-native/splash-screen';
7 import { StatusBar } from '@ionic-native/status-bar';
8
9 import { AuthService } from './providers/auth-service';
10
11 import { Email } from '../utils/email';
12
13 import { MyApp } from './app.component';
14
15 import { HomePage } from './pages/home/home';
16 import { LoginPage } from './pages/login/login';
17 import { RegisterPage } from './pages/register/register';
18
19 @NgModule({
20   declarations: [
21     MyApp,
22     HomePage,
23     LoginPage,
24     RegisterPage
25   ],
26   imports: [
27     BrowserModule,
28     IonicModule.forRoot(MyApp),
29
30     HttpClientModule,
31     bootstrap: [IonicApp],
32     entryComponents: [
33       MyApp,
34       HomePage,
35       LoginPage,
36       RegisterPage
37     ],
38     providers: [
39       StatusBar,
40       SplashScreen,
41       {provide: ErrorHandler, useClass: IonicErrorHandler},
42       AuthService,
43       Email
44     ]
45   ]
46 })
47 export class AppModule {
48 }
```

## Imports

```
import { IonicStorageModule } from '@ionic/storage';
import { SQLite } from '@ionic-native/sqlite';
```

```
import { DbProvider } from '../providers/db-provider';
```

```
IonicStorageModule.forRoot({
  name: 'thinkam',
  driverOrder: ['websql']
}),
```

## Providers

```
SQLite,
DbProvider,
```



No componente de Login teremos as seguintes modificações:

```
3 import { AuthService } from '../../../../../providers/auth-service';
4
5
6 import { HomePage } from '../home/home';
7 import { RegisterPage } from '../register/register';
8
9 @Component({
10   selector: 'page-login',
11   templateUrl: 'login.html',
12 })
13 export class LoginPage {
14   loading: Loading;
15   registerCredentials = { email: '', password: '' };
16   constructor(private nav: NavController,
17             private auth: AuthService,
18             private alertCtrl: AlertController,
19             private loadingCtrl: LoadingController) {
20 }
21
22 public createAccount() {
23   this.nav.push(RegisterPage);
24 }
25
26 public login() {
27   this.showLoading();
28   this.auth.login(this.registerCredentials).subscribe(res => {
29     if (res.ok) {
30       this.nav.setRoot(HomePage);
31     } else {
32       this.showError(res.data);
33     }
34   });
35 }
36 }
```

```
3 import { AuthService } from '../../../../../providers/auth-service';
4 +import { DbProvider } from '../../../../../providers/db-provider';
5
6
7 import { HomePage } from '../home/home';
8 import { RegisterPage } from '../register/register';
9
10 @Component({
11   selector: 'page-login',
12   templateUrl: 'login.html',
13 })
14 export class LoginPage {
15   loading: Loading;
16   registerCredentials = { email: '', password: '' };
17   constructor(private nav: NavController,
18             private auth: AuthService,
19             private db: DBProvider,
20             private alertCtrl: AlertController,
21             private loadingCtrl: LoadingController) {
22 }
23
24 public createAccount() {
25   this.nav.push(RegisterPage);
26 }
27
28 public login() {
29   this.showLoading();
30   this.auth.login(this.registerCredentials).subscribe(res => {
31     if (res.ok) {
32       this.db.set("user", res.data);
33       this.nav.setRoot(HomePage);
34     } else {
35       this.showError(res.data);
36     }
37   });
38 }
```

Vamos testar nosso App? Fiz login com o e-mail [felipe.almeida@thinkam.net](mailto:felipe.almeida@thinkam.net) que já havia utilizado em testes nos bastidores e consegui o seguinte resultado:

The screenshot shows the Chrome DevTools Network tab with a single request to 'http://localhost:8100/api/login'. The response status is 200 OK, and the response body is a JSON object: {"user": {"name": "Felipe Almeida", "token": "NmQyODU1YjctOGY3OS00NjNlWiXZktNDRlZddMlgwZQw"}}, indicating a successful login.

Below the Network tab, the Application tab is open, showing the Storage section. It lists a key 'user' with value: {"name": "Felipe Almeida", "token": "NmQyODU1YjctOGY3OS00NjNlWiXZktNDRlZddMlgwZQw"}. The Storage section also includes Local Storage, Session Storage, IndexedDB, Web SQL, and Cookies.

The Console tab at the bottom shows several warning messages related to Cordova and Native code, such as 'viewport-fit' being ignored and Cordova not being available.

Como sempre, deixei registrado lá no [GitHub](#) para vocês!



## 6. Integração com o Wakatime através do provider e exibição dos dados do usuário da API na tela inicial do nosso APP

Já que nós vamos exibir os dados do usuário na tela inicial, vamos começar criando o model desse objeto que será exibido, lá em **src/models** criaremos o arquivo **current-user.ts**:

```
export class CurrentUser {  
    created_at: string;  
    display_name: string;  
    email: string;  
    email_public: boolean;  
    full_name: string;  
    has_premium_features: boolean;  
    human_readable_website: string;  
    id: string;  
    is_email_confirmed: boolean;  
    is_hireable: boolean;  
    languages_used_public: boolean;  
    last_heartbeat: string;  
    last_plugin: string;  
    last_plugin_name: string;  
    last_project: string;  
    location: string;  
    logged_time_public: boolean;  
    modified_at: any;  
    photo: string;  
    photo_public: boolean;  
    plan: string;  
    timezone: string;  
    username: string;  
    website: string;  
  
    constructor(){  
        this.display_name = "";  
        this.full_name = "";  
        this.photo = "";  
        this.website = "";  
        this.human_readable_website = "";  
        this.last_plugin = "";  
        this.location = "";  
        this.username = "";  
        this.created_at = "";  
        this.username = "";  
    }  
}
```

Notem que só estamos inicializando no constructor aqueles atributos que serão exibidos ou futuramente poderão ser utilizados na tela (exemplo: **full\_name**).



No model acima separamos para ser exibida a coluna **created\_at** que será uma data que chegará sem formatação para ser exibida e precisamos formata-la, seguindo uma boa prática que é separar essa função em um arquivo a parte e em todos os lugares que forem necessárias essa utilidade, teremos a chamada simples dessa função sem ter que reescrever o conteúdo dela que estará centralizado em um único arquivo que criamos em **src/utils**, chamado de **date.ts**:

The screenshot shows the VS Code interface. On the left, the Explorer sidebar displays the project structure:

- OPEN EDITORS:
  - TS date.ts src/utils
- THINKAMWAKATIME
  - .sourcemaps
  - hooks
  - node\_modules
  - platforms
  - plugins
  - resources
- src
  - app
  - assets
  - models
  - pages
  - providers
  - theme
  - utils
    - TS date.ts
    - TS email.ts
- index.html

The main editor area shows the code for **date.ts**:

```
1 import { Injectable } from '@angular/core';
2
3 @Injectable()
4 export class Date {
5   format(date: string){
6     let dateArray = date.substring(0, 10).split("-");
7
8     return dateArray[2] + "/" + dateArray[1] + "/" + dateArray[0];
9   }
10 }
```



Existe um ditado de escoteiro que é o seguinte, quando sair de um lugar deixe sempre mais limpo do que quando você chegou, a mesma coisa podemos fazer com o nosso código, eu estava observando nosso **auth-service** para seguir o mesmo padrão no **waka-service** e me deparei com uma possível melhoria, tratei então de aplicá-la porque era algo bem simples, que foi pegar um texto que estava sendo repetido em dois lugares e coloca-lo em uma variável, dessa forma se houver alteração nesse texto, eu só terei que alterar uma vez essa informação, e não sair alterando em tudo, veja como ficou o **antes** e **depois**:

```
4 import { Observable } from 'rxjs/Observable';
5
6 import 'rxjs/add/operator/map';
7 import 'rxjs/add/observable/throw';
8 import 'rxjs/add/operator/catch';
9
10 import { User } from '../models/user';
11
12 import { Email } from '../utils/email';
13
14 @Injectable()
15 export class AuthService {
16
17   constructor(private emailUtils : Email,
18             private http: Http){
19
20   }
21
22   currentUser: User;
23
24   public login(credentials) {
25     if (credentials.email === null || credentials.password === null) {
26       return Observable.throw("Por favor, informe suas credenciais");
27     } else {
28      return this.http.post("http://thinkam.azurewebsites.net/api/login", credentials)
29        .map(res => res.json())
30        .catch(error => Observable.throw(error.json()));
31    }
32  }
33
34   public register(credentials) {
35     if (credentials.name === null ||
36         credentials.email === null ||
37         credentials.password === null ||
38         credentials.secretAPIKey === null) {
39       return Observable.throw("Por favor informe todas as informações de registro.");
40     } else {
41       if (credentials.name.length < 3){
42         return Observable.throw("Por favor informe um nome com no mínimo 3 caracteres.");
43       }
44       if (!this.emailUtils.validateEmail(credentials.email)){
45         return Observable.throw("Por favor informe um e-mail válido.");
46       }
47       if (credentials.secretAPIKey.length < 8){
48         return Observable.throw("Por favor informe uma secret API key com no mínimo 8 caracteres");
49       }
50
51      return this.http.post("http://thinkam.azurewebsites.net/api/user", credentials)
52        .map(res => res.json())
53        .catch(error => Observable.throw(error.json()));
54    }
55
56   public login(credentials) {
57     if (credentials.email === null || credentials.password === null) {
58       return Observable.throw("Por favor, informe suas credenciais");
59     } else {
60       return this.http.post(this.baseApiUrl + "login", credentials)
61         .map(res => res.json())
62         .catch(error => Observable.throw(error.json()));
63    }
64
65   public register(credentials) {
66     if (credentials.name === null ||
67         credentials.email === null ||
68         credentials.password === null ||
69         credentials.secretAPIKey === null) {
69       return Observable.throw("Por favor informe todas as informações de registro.");
70     } else {
71       if (credentials.name.length < 3){
72         return Observable.throw("Por favor informe um nome com no mínimo 3 caracteres.");
73       }
74       if (!this.emailUtils.validateEmail(credentials.email)){
75         return Observable.throw("Por favor informe um e-mail válido.");
76       }
77       if (credentials.secretAPIKey.length < 8){
78         return Observable.throw("Por favor informe uma secret API key com no mínimo 8 caracteres");
79       }
80
81      return this.http.post(this.baseApiUrl + "user", credentials)
82        .map(res => res.json())
83        .catch(error => Observable.throw(error.json()));
84    }
85
86  }
```

Voltando ao nosso novo serviço, teremos o seguinte comando no nosso terminal:

ionic g provider wakaService



Como adicionamos novos providers que nos ajudarão, teremos que adicionar ele no **app.module.ts**:

```
10 import { AuthService } from './providers/auth-service';
11
12 import { DbProvider } from '../providers/db-provider';
13
14 import { Email } from '../utils/email';
15
16 import { MyApp } from './app.component';
17
18 import { HomePage } from '../pages/home/home';
19 import { LoginPage } from '../pages/login/login';
20 import { RegisterPage } from '../pages/register/register';
21
22 @NgModule({
23   declarations: [
24     MyApp,
25     HomePage,
26     LoginPage,
27     RegisterPage
28   ],
29   imports: [
30     BrowserModule,
31     IonicModule.forRoot(MyApp),
32     IonicStorageModule.forRoot({
33       name: 'thinkam',
34       driverOrder: ['websql']
35     }),
36     HttpModule
37   ],
38   bootstrap: [IonicApp],
39   entryComponents: [
40     MyApp,
41     HomePage,
42     LoginPage,
43     RegisterPage
44   ],
45   providers: [
46     StatusBar,
47     SplashScreen,
48     {provide: ErrorHandler, useClass: IonicErrorHandler},
49     AuthService,
50     SQLite,
51     DbProvider,
52     Email
53   ]
54 }
```

```
10 import { AuthService } from './providers/auth-service';
11 import { AuthService } from './providers/auth-service';
12 + import { WakaService } from '../providers/waka-service';
13 +
14 import { DbProvider } from '../providers/db-provider';
15
16 import { Email } from '../utils/email';
17 + import { Date } from '../utils/date';
18
19 import { MyApp } from './app.component';
20
21 import { HomePage } from '../pages/home/home';
22 import { LoginPage } from '../pages/login/login';
23 import { RegisterPage } from '../pages/register/register';
24
25 @NgModule({
26   declarations: [
27     MyApp,
28     HomePage,
29     LoginPage,
30     RegisterPage
31   ],
32   imports: [
33     BrowserModule,
34     IonicModule.forRoot(MyApp),
35     IonicStorageModule.forRoot({
36       name: 'thinkam',
37       driverOrder: ['websql']
38     }),
39     HttpModule
40   ],
41   bootstrap: [IonicApp],
42   entryComponents: [
43     MyApp,
44     HomePage,
45     LoginPage,
46     RegisterPage
47   ],
48   providers: [
49     StatusBar,
50     SplashScreen,
51     {provide: ErrorHandler, useClass: IonicErrorHandler},
52     AuthService,
53 +     WakaService,
54     SQLite,
55     DbProvider,
56 +     Email,
57 +     Date
58   ]
59 })
```



Em seguida adicione o conteúdo a seguir:

```
import { Http, Headers, RequestOptions } from '@angular/http';

import { Injectable } from '@angular/core';

import { Observable } from 'rxjs/Observable';

import { DbProvider } from './db-provider';

import 'rxjs/add/operator/map';
import 'rxjs/add/observable/throw';
import 'rxjs/add/operator/catch';

@Injectable()
export class WakaService {

  baseApiUrl: string = "https://wakatime.com/api/v1/";

  headers: Headers;
  options: RequestOptions;

  constructor(private http: Http,
    private dbProvider: DbProvider) {

  }

  getCurrentUser(){
    return new Promise((resolve: any, reject: any) => {
      this.dbProvider.get("user").then((user: any) => {
        this.headers = new Headers({
          'Authorization': 'Basic ' + user.token
        });

        this.options = new RequestOptions({ headers: this.headers });

        this.http.get(this.baseApiUrl + "users/current", this.options)
          .map(res => res.json())
          .catch(error => reject(error.json()))
          .subscribe((res: any) => {
            resolve(res.data);
          });
      });
    });
  }
}
```

Agora podemos chamar esta função no componente no momento que ele é construído:

```

1 import { Component } from '@angular/core';
2 -import { NavController } from 'ionic-angular';
3
4 @Component({
5   selector: 'page-home',
6   templateUrl: 'home.html'
7 })
8 export class HomePage {
9
10 -  constructor(public navCtrl: NavController) {
11 -
12   }
13 -
14 }
15

```

```

1 import { Component } from '@angular/core';
2
3 +import { WakaService } from '../../../../../providers/waka-service';
4 +
5 +import { Date } from '../../../../../utils/date';
6 +
7 +import { CurrentUser } from '../../../../../models/current-user';
8 +
9 @Component({
10   selector: 'page-home',
11   templateUrl: 'home.html'
12 })
13 export class HomePage {
14 +  currentUser: CurrentUser = new CurrentUser();
15
16 +  constructor(private wakaService: WakaService,
17 +               private dateUtils: Date) {
18 +    this.wakaService.getCurrentUser().then((currentUser: any) => {
19 +      console.log(currentUser);
20 +      currentUser.created_at = this.dateUtils.format(currentUser.created_at);
21 +      this.currentUser = currentUser;
22 +    })
23   }
24 }
25

```

Por fim teremos nosso HTML:

```

1 <ion-header>
2   <ion-navbar>
3     <ion-title>
4       Think A.M. - Wakatime
5     </ion-title>
6   </ion-navbar>
7 </ion-header>
8
9 <ion-content padding>
10 - Curso de Mobile Básico
11   <p>
12     | Se você está perdido, veja os <a href="http://io
13   </p>
14
15
16
17 +<ion-item>
18 +  <ion-avatar item-start>
19 +    | 
20 +  </ion-avatar>
21 +  <h2>{{currentUser.display_name}}</h2>
22 +  <p>{{currentUser.created_at}}</p>
23 +</ion-item>
24 +<ion-col width=67>
25 +  | 
26 +</ion-col>
27 +
28 +<ion-card-content>
29 +  | <a [href]="currentUser.website">{{currentUser.human_readable_website}}</a>
30 +</ion-card-content>
31 +
32 +<ion-row>
33 +  <ion-col>
34 +    <button ion-button icon-left clear small>
35 +      | <ion-icon name="map"></ion-icon>
36 +      | <div>{{currentUser.location}}</div>
37 +    </button>
38 +  </ion-col>
39 +  <ion-col>
40 +    <button ion-button icon-left clear small>
41 +      | <ion-icon name="contact"></ion-icon>
42 +      | <div>{{currentUser.username}}</div>
43 +    </button>
44 +  </ion-col>
45 +  <ion-col>
46 +    <button ion-button icon-left clear small>
47 +      | <ion-icon name="code"></ion-icon>
48 +      | <div>{{currentUser.last_plugin_name}}</div>
49 +    </button>
50 +  </ion-col>
51 +</ion-row>
52 +
53 +</ion-card>
54 </ion-content>

```

Vamos testar no nosso emulador de android e não no navegador chrome, porque para as requisições do Wakatime não conseguiremos liberação para nossa URL, portanto vamos aos passos para que isso seja possível:

- Vamos baixar o [Gradle Build Tool](#) e seguir os passos de instalação conforme ensino no vídeo
- Iniciaremos nosso emulador pelo Visual Studio (como sugerimos no início do curso) ou simplesmente na pasta da SDK do Android (caso você tenha baixado apenas a SDK), ou ainda você pode olhar este [GUIA](#) do cordova e fazer utilizando o Android Studio que também funciona.
- Adicione a plataforma do android no projeto, utilize o comando:  
`ionic platform add android`
- Garanta que as dependências em seu **package.config** estejam assim depois de feito o passo anterior:

```

  20 |     "@angular/http": "4.1.3",
  21 |     "@angular/platform-browser": "4.1.3",
  22 |     "@angular/platform-browser-dynamic": "4.1.3",
  23 |     "@ionic-native/core": "3.12.1",
  24 |     "@ionic-native/splash-screen": "3.12.1",
  25 -    "@ionic-native/status-bar": "3.12.1",
  26 |     "@ionic-native/sqlite": "3.12.1",
  27 |     "@ionic/app-scripts": "2.1.4",
  28 |     "@ionic/storage": "2.0.1",
  29 |     "ionic-angular": "3.6.1",
  30 |     "ionicons": "3.0.0",
  31 |     "rxjs": "5.4.0",
  32 |     "sw-toolbox": "3.6.0",
  33 -    "zone.js": "0.8.12"
  34 |
  35 },
  36 "devDependencies": {
  37     "@ionic/app-scripts": "2.1.4",
  38     "typescript": "2.3.4"
  39 },
  40 "cordovaPlugins": [
  41     "cordova-plugin-whitelist",
  42     "cordova-plugin-device",
  43     "cordova-plugin-console",
  44     "cordova-plugin-splashscreen",
  45     "cordova-plugin-statusbar",
  46     "ionic-plugin-keyboard",
  47     "cordova-sqlite-storage"
  48 ],
  49 - "cordovaPlatforms": [],
  50 - "description": "Wakatime - Think A.M.: An Ionic pr
  51 -
  52
  53
  54
  55 +
  56 +
  57 +
  58 +
  59 +
  60 +
  61 +
  62 +
  63 +
  64 +
  65 +
  66 +
  67 +
  68 +
  69 }

  20 |     "@angular/http": "4.1.3",
  21 |     "@angular/platform-browser": "4.1.3",
  22 |     "@angular/platform-browser-dynamic": "4.1.3",
  23 |     "@ionic-native/core": "3.12.1",
  24 |     "@ionic-native/splash-screen": "3.12.1",
  25 |     "@ionic-native/sqlite": "3.12.1",
  26 +    "@ionic-native/status-bar": "3.12.1",
  27 |     "@ionic/app-scripts": "2.1.4",
  28 |     "@ionic/storage": "2.0.1",
  29 +    "cordova-android": "^6.2.3",
  30 |     "ionic-angular": "3.6.1",
  31 |     "ionicons": "3.0.0",
  32 |     "rxjs": "5.4.0",
  33 |     "sw-toolbox": "3.6.0",
  34 +    "zone.js": "0.8.12",
  35 +
  36 +
  37 +
  38 +
  39 +
  40 },
  41 "devDependencies": {
  42     "@ionic/app-scripts": "2.1.4",
  43     "typescript": "2.3.4"
  44 },
  45 "cordovaPlugins": [
  46     "cordova-plugin-whitelist",
  47     "cordova-plugin-device",
  48     "cordova-plugin-console",
  49     "cordova-plugin-splashscreen",
  50     "cordova-plugin-statusbar",
  51     "ionic-plugin-keyboard",
  52     "cordova-sqlite-storage"
  53 ],
  54 "cordovaPlatforms": [],
  55 + "description": "Wakatime - Think A.M.: An Ionic project",
  56 + "cordova": {
  57 +     "plugins": {
  58 +         "cordova-plugin-device": {},
  59 +         "cordova-plugin-splashscreen": {},
  60 +         "cordova-plugin-statusbar": {},
  61 +         "cordova-plugin-whitelist": {},
  62 +         "ionic-plugin-keyboard": {}
  63 +     },
  64 +     "platforms": [
  65 +         "android"
  66 +     ]
  67 + }
  68 +
  69 }
```



- Algo importante que foi mostrado em vídeo e quero reforçar é a adição da linha nas preferências do nosso **config.xml**

```
<preference name="loadUrlTimeoutValue" value="60000" />
```

```
diff --git a/config.xml b/config.xml
--- a/config.xml
+++ b/config.xml
@@ -1,24 +1,26 @@
- <?xml version="1.0" encoding="UTF-8" standalone="yes"?
+ <?xml version="1.0" encoding="utf-8"?>
  <widget id="net.thinkam.wakatimethinkam" version="0.1.0">
    <name>Wakatime - Think A.M.</name>
    <description>Um aplicativo incrível usando Ionic/Cordova</description>
    <author email=" contato@thinkam.net " href="http://thinkam.net/">Think A.M. Team</author>
    <content src="index.html" />
    <access origin="*"/>
-   <allow-navigation href="http://ionic.local/*"/>
+   <allow-navigation href="http://ionic.local/*" />
-   <allow-intent href="http://*/*"/>
+   <allow-intent href="https://*/*"/>
-   <allow-intent href="tel:*"/>
-   <allow-intent href="sms:*"/>
-   <allow-intent href="mailto:*"/>
-   <allow-intent href="geo:*"/>
+   <allow-intent href="geo:*" />
-   <preference name="webviewbounce" value="false"/>
+   <preference name="webviewbounce" value="false" />
-   <preference name="UIWebViewBounce" value="false"/>
+   <preference name="UIWebViewBounce" value="false" />
-   <preference name="DisallowOverscroll" value="true"/>
+   <preference name="DisallowOverscroll" value="true" />
-   <preference name="android-minSdkVersion" value="16"/>
+   <preference name="android-minSdkVersion" value="16" />
-   <preference name="BackupWebStorage" values="none"/>
+   <preference name="BackupWebStorage" values="none" />
-   <preference name="SplashMaintainAspectRatio" value="true"/>
+   <preference name="SplashMaintainAspectRatio" value="true" />
-   <preference name="FadeSplashScreenDuration" value="300"/>
+   <preference name="FadeSplashScreenDuration" value="300" />
-   <preference name="SplashShowOnlyFirstTime" value="true"/>
+   <preference name="SplashShowOnlyFirstTime" value="false" />
-   <preference name="SplashScreen" values="screen"/>
+   <preference name="SplashScreen" values="screen" />
-   <preference name="SplashScreenDelay" value="3000"/>
+   <preference name="SplashScreenDelay" value="3000" />
-   <platform name="android">
+   <platform name="android" />
-     <allow-intent href="market:*"/>
+     <allow-intent href="market:" />
-     <icon density="ldpi" src="resources/android/icon/>
+     <icon density="ldpi" src="resources/android/icon/>
-     <icon density="mdpi" src="resources/android/icon/>
+     <icon density="mdpi" src="resources/android/icon/>
-     <icon density="hdpi" src="resources/android/icon/>
+     <icon density="hdpi" src="resources/android/icon/>
-     <icon density="xhdpi" src="resources/android/icon/>
+     <icon density="xhdpi" src="resources/android/icon/>
-     <icon density="xxhdpi" src="resources/android/icon/>
+     <icon density="xxhdpi" src="resources/android/icon/>
-     <icon density="xxxhdpi" src="resources/android/icon/>
+     <icon density="xxxhdpi" src="resources/android/icon/>
-     <splash density="land-ldpi" src="resources/android/splash/>
+     <splash density="land-ldpi" src="resources/android/splash/>
-     <splash density="land-mdpi" src="resources/android/splash/>
+     <splash density="land-mdpi" src="resources/android/splash/>
-     <splash density="land-hdpi" src="resources/android/splash/>
+     <splash density="land-hdpi" src="resources/android/splash/>
-     <splash density="land-xhdpi" src="resources/android/splash/>
+     <splash density="land-xhdpi" src="resources/android/splash/>
-     <splash density="land-xxhdpi" src="resources/android/splash/>
+     <splash density="land-xxhdpi" src="resources/android/splash/>
-     <splash density="port-ldpi" src="resources/android/splash/>
+     <splash density="port-ldpi" src="resources/android/splash/>
-     <splash density="port-mdpi" src="resources/android/splash/>
+     <splash density="port-mdpi" src="resources/android/splash/>
-     <splash density="port-hdpi" src="resources/android/splash/>
+     <splash density="port-hdpi" src="resources/android/splash/>
```

Após feito isso é só executar o comando que fazemos a instalação no emulador do android **ionic run android -target “emulator-5554”** (5554 pode alterar de acordo com o ambiente, você verá esse número quando iniciar o emulador)

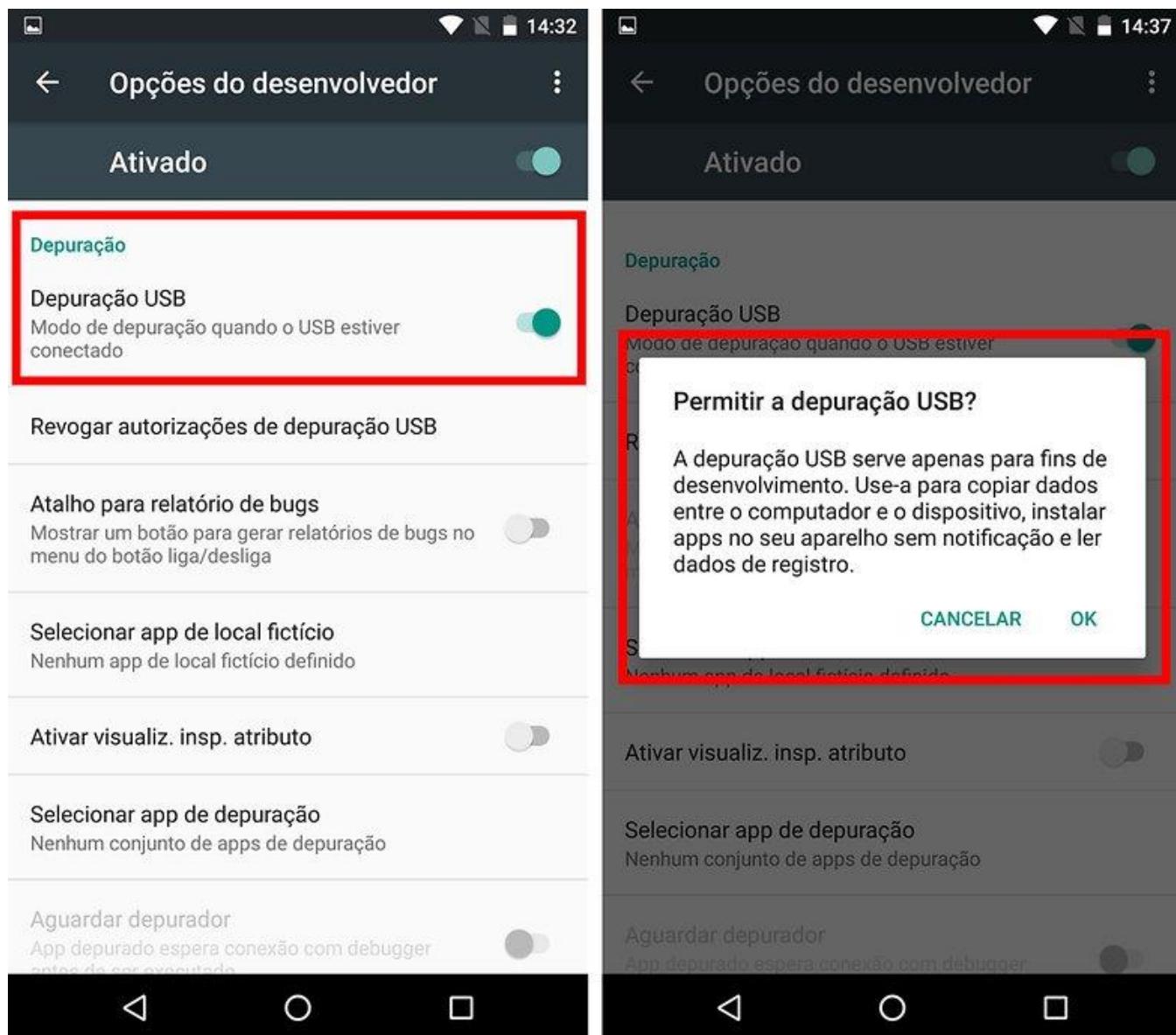
Alguns trechos de código não foram inseridos aqui, isso é para reforçar o uso do GitHub que tem esse [commit](#) registrados e todas as mudanças que foram feitas nesse vídeo.

## 7. Exportar para o seu dispositivo em modo debug (testes sem passar pela playstore) e também gerar a release e a keystore para subir o aplicativo na PlayStore como Alfa

Vamos primeiro aprender como gerar o aplicativo em nosso celular android que estará conectado ao computador que possui o projeto que fizemos até aqui.

Quando conectamos ele, primeiro devemos habilitar o modo desenvolvedor no celular android, conforme este [guia](#).

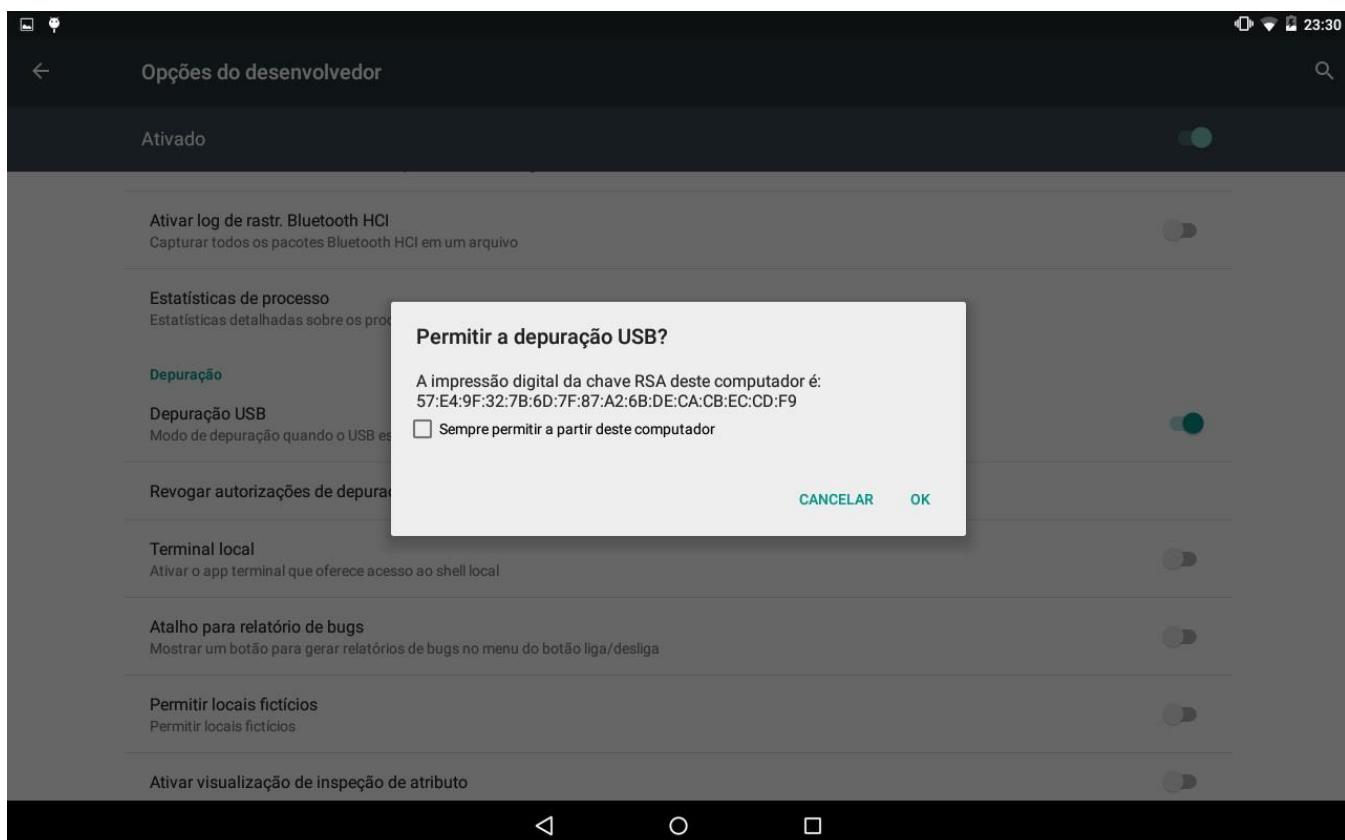
Com o modo de desenvolvedor habilitado, vá na opção programador que foi habilitada e ative a opção Depuração USB, ok?



Fez isso? Então vamos em frente...



Agora ficou simples, quando você conectar o seu celular no USB do computador que está rodando o projeto, aparecerá uma caixa de mensagem no celular perguntando se você permite que esta conexão tenha a depuração USB ativa, inclusive você pode colocar para sempre confiar nesse seu computador para não ficar perguntando todas as vezes.



Agora seja no seu windows, mac ou linux, [se todas as variáveis de ambiente estão devidamente cadastradas](#), execute o comando **adb devices**:

```
platform-tools — -bash — 75x10
MacBook-Pro-de-Think-AM:platform-tools felipe$ adb devices
List of devices attached
0422820189      device
```

The terminal window shows the command 'adb devices' being run. The output lists a single device with the identifier '0422820189' and the status 'device'. The entire line '0422820189 device' is highlighted with a red box.

**0422820189** é a identificação do android do meu celular para o meu computador.



No caso de gerar uma versão debug para este aparelho, a única coisa que muda no comando que estávamos utilizando é o nosso alvo, ou seja, o **-target** que passa de **emulator-XXXX** para o **código que está aparecendo no seu terminal/cmd, que no meu caso é 0422820189**. Portanto ficará da seguinte forma:

```
ionic run android -target "0422820189"
```

Ou para você...

```
ionic run android -target "xxxxxxxxxxxxxxxxxxxx"
```

Agora vamos para o processo de geração da release:

## Passo 1 – Gerar o www a partir do typescript:

Para isso podemos executar o comando **ionic serve** ou o **ionic build android**, porém no segundo pode ocorrer de aparecer a seguinte mensagem de erro:

```
Error: No platforms added to this project. Please use `cordova platform add <platform>`.
```

Neste caso execute **cordova platform add android** como sugere a mensagem acima.

Lembrando que o [Gradle](#) deve estar instalado conforme demonstrado anteriormente.

Segue um guia rápido para quem está no Windows e pode encontrar dificuldades:

Gradle | Releases

Seguro | https://gradle.org/releases/

Command-line completion scripts for bash and zsh can be downloaded from the [gradle-completion project page](#).

### Getting Started Resources

The Gradle team offers free [introductory training](#) and [advanced training](#) courses bi-monthly.

There are many [getting started guides](#) are available to help you get building quickly. Source distributions also include many [working samples](#) for which guides are not yet written.

v4.2

Sep 20, 2017

- Download: [binary-only](#) [sha256] or [complete](#) [sha256]
- [User Manual](#)
- [API Javadoc](#)
- [DSL Reference](#)
- [Release Notes](#)

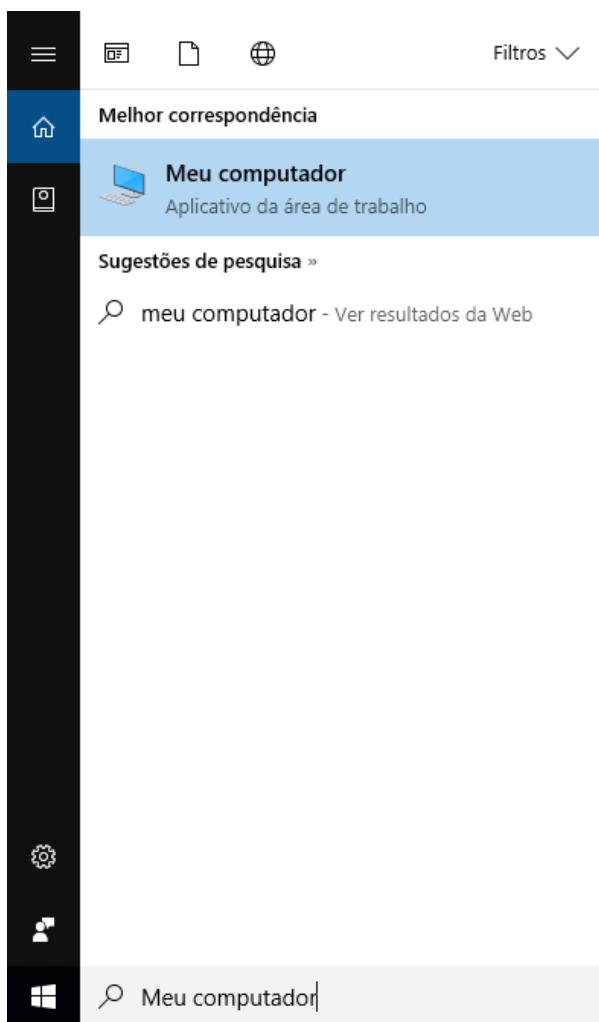
Esta foi a versão que acabei de baixar no meu computador Windows, quando gravei realizando o procedimento no mac não demonstrei aonde cliquei para baixar, mas é bem simples, é apenas seguir as instruções do gradle.org, lembre sempre: “A coisa



mais difícil na vida é aprender a ler, o resto está escrito". Isso serve para qualquer informação que ficar faltando nessa documentação, ou seja, tudo pode ser pesquisado, provavelmente alguém já escreveu algo que bate com a solução do seu problema e ao ler a solução é somente colocar em prática para solucionar o problema, caso não resolva repita o processo até resolver, te garanto que até hoje todas as coisas que não sabia e realizei esse processo, eu consegui resolver em algum momento.

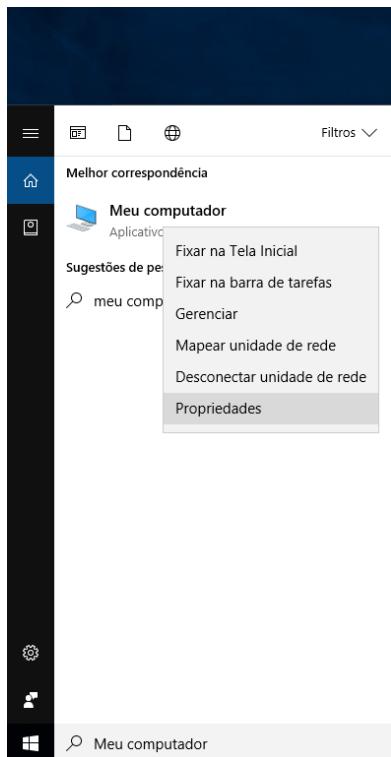
Mas mesmo assim vou colocar aqui todos os imprevistos que eu acredito que vocês possam ter, no Windows mesmo, tem alguns passos que eu posso detalhar para vocês aqui referente aos pré-requisitos para realizar build, no caso de adicionar a variável de ambiente do Gradle, siga os seguintes passos:

1. Digite na barra de pesquisa **Meu Computador**:

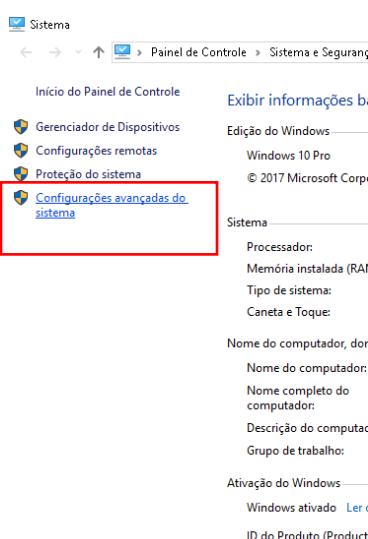




2. Clique com o botão direito do mouse em cima do ícone e escolha a opção **Propriedades**:



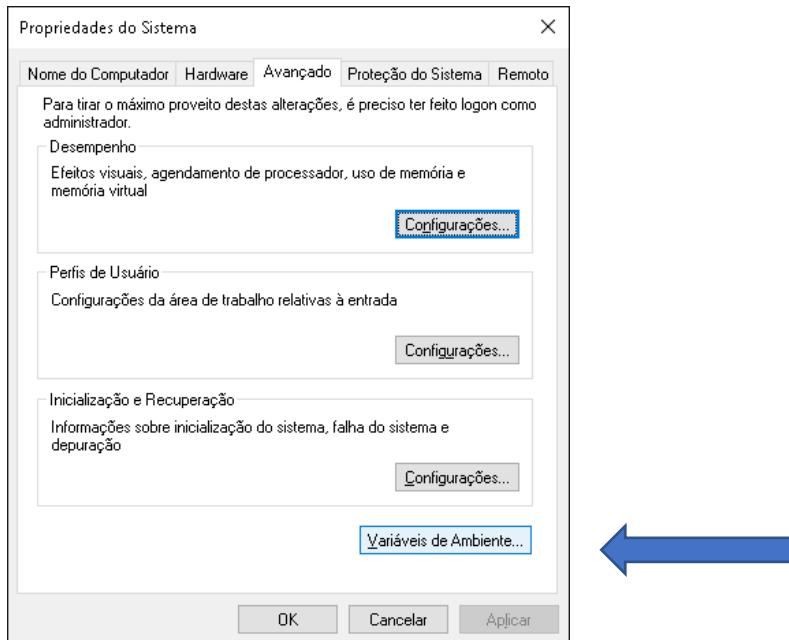
3. Abrirá uma janela chamada Sistema, nela escolha a opção **Configurações avançadas do sistema**:



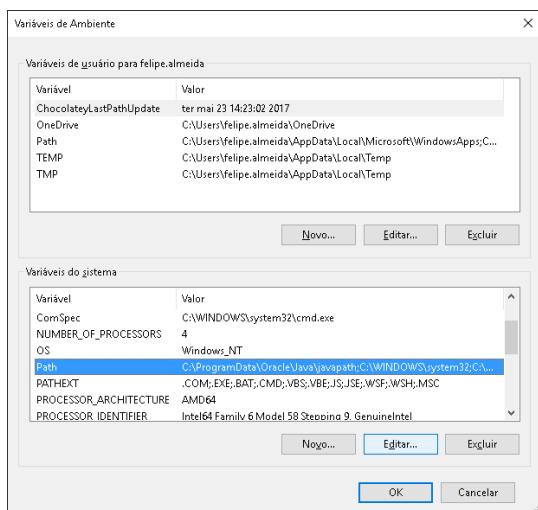
Consulte também  
Segurança e Manutenção



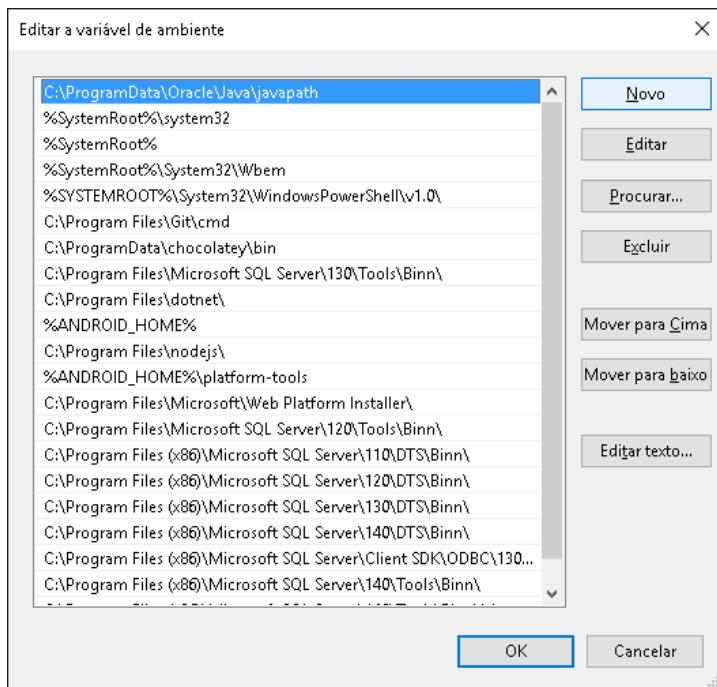
4. Abrirá uma janela das Propriedades do Sistema na aba Avançado, nessa janela você vai clicar no botão **Variáveis de Ambiente**:



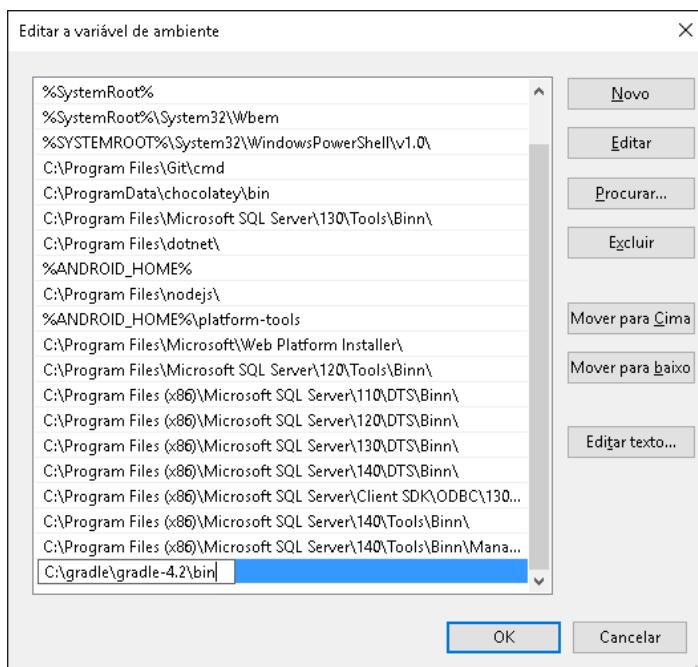
5. Em Variáveis de Ambiente na seção de Variáveis do Sistema, selecione a linha com a Variável chamada Path e clique no botão Editar...



6. Na janela “Editar a variável de ambiente” clique no botão Novo:



7. Informe o caminho que estará incluso na lista de caminhos dessa variável de ambiente, neste caso estou adicionando o caminho para o Gradle (c:\gradle\gradle-4.2\bin) que é o que estou testando em meu Windows:



Após feito isso clique em OK nesta janela e na janela de Variáveis de Ambiente que se encontra aberta.



## 8. Realize o teste digitando no prompt de comando **gradle -v**

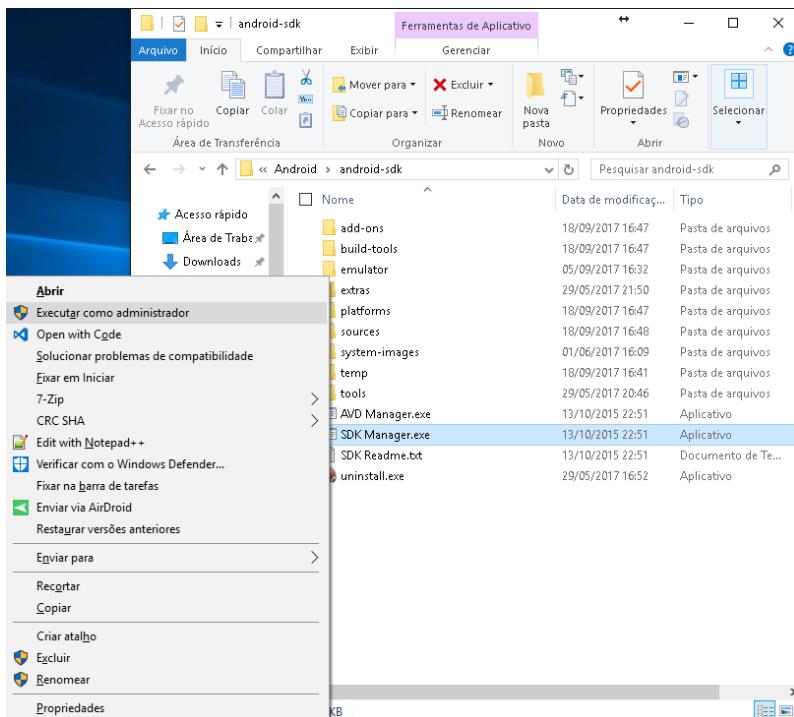
```
C:\Selecionar Prompt de Comando
C:\ThinkAcademy\github\Wakatime\ThinkAMWakatime>gradle -v
-----
Gradle 4.2
-----
Build time: 2017-09-20 14:48:23 UTC
Revision: 5ba503cc17748671c83ce35d7da1cffd6e24dfbd

Groovy: 2.4.11
Ant: Apache Ant(TM) version 1.9.6 compiled on June 29 2015
JVM: 1.8.0_131 (Oracle Corporation 25.131-b11)
OS: Windows 10 10.0 amd64

C:\ThinkAcademy\github\Wakatime\ThinkAMWakatime>
```

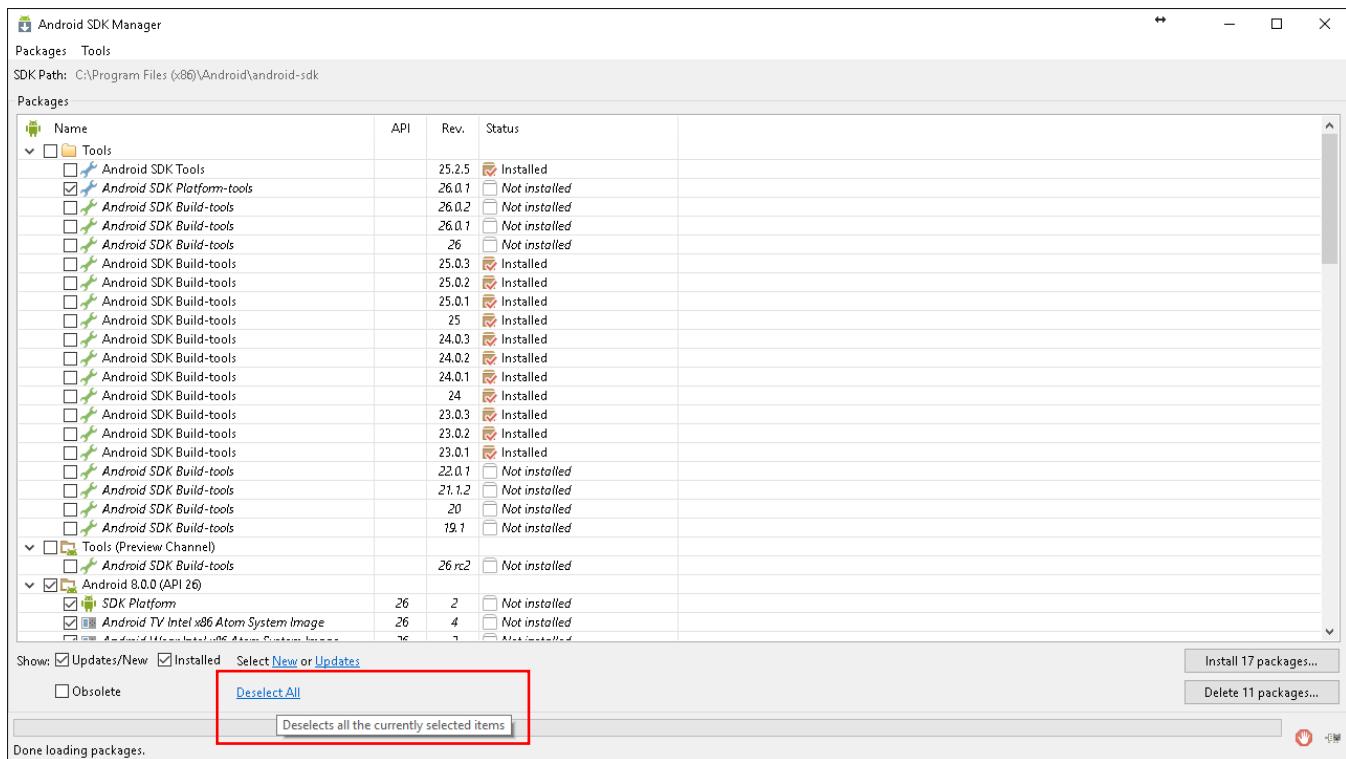
Outro detalhe importante que pode ocorrer no build, é você não ter aceito e instalado no Android SDK Manager a package “Android SDK Platform-tools”, neste caso siga os seguintes passos:

1. Vá até a pasta da SDK do Android em seu computador e inicie o SDK Manager como Administrador:



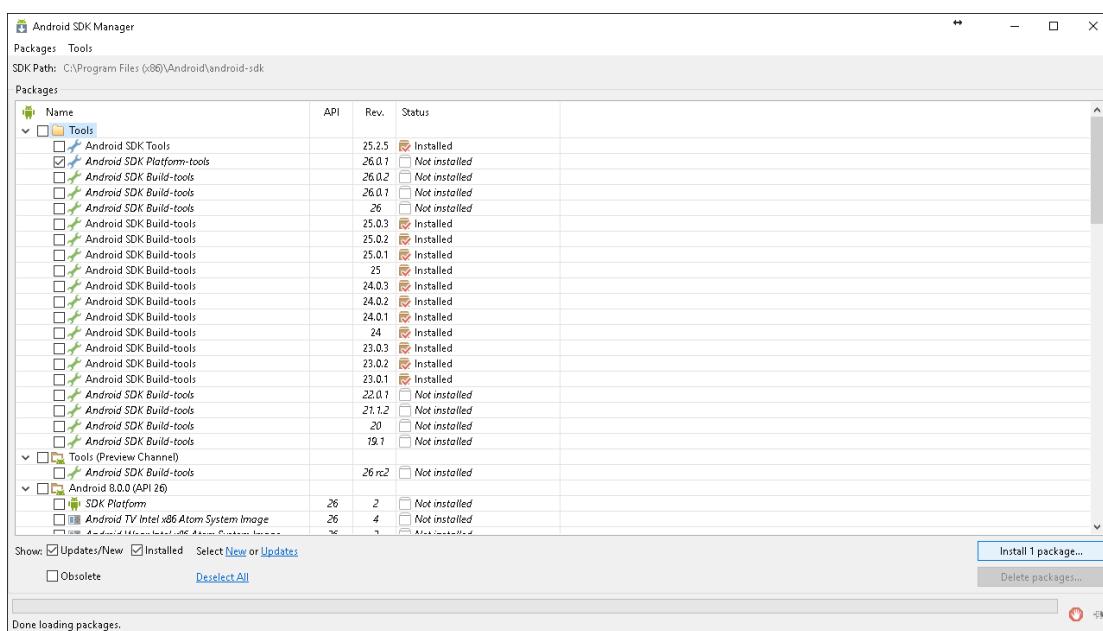


## 2. Escolha embaixo da janela a opção desmarcar todos (Deselect All)



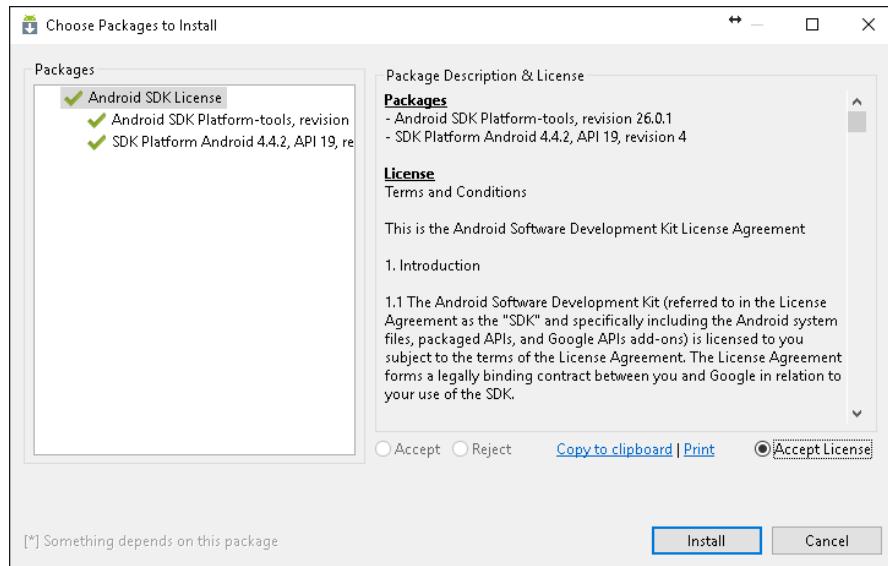
Note que a segunda opção “Android SDK Platform-tools” encontra-se com o status de não instalada.

## 3. Selecione apenas a opção que queremos instalar “Android SDK Platform-tools” e clique em Install 1 package...





#### 4. Surgira a janela a seguir que você vai aceitar e instalar de fato



Agora sim! Após todos os passos acima estarem concluídos, poderemos fazer o comando `ionic build android` sem problemas no Windows, no linux não realizei o teste, mas é bem semelhante com a forma que fiz no Mac, exceto que no Linux não temos o Visual Studio que sugeri, mas coloquei um [GUIA](#) massa que vai lhe dar os passos da SDK no Linux, além do Windows e Mac que está lá também de uma outra forma que não demonstrei no vídeo.

```
BUILD SUCCESSFUL

Total time: 1 mins 20.43 secs
Built the following apk(s):
  C:/ThinkAcademy/github/Wakatime/ThinkAMWakatime/platforms/android/build/outputs/apk/android-debug.apk
```

Lembrando que se ainda assim ocorrer um outro erro que não está listado aqui você pode executar também o comando `ionic serve` que também atualiza o nosso www.

#### Passo 2 – Gerar release para android (release necessária para criar o apk da playstore)

```
cordova build --release android
```

```
BUILD SUCCESSFUL

Total time: 23.526 secs
Built the following apk(s):
  C:/ThinkAcademy/github/Wakatime/ThinkAMWakatime/platforms/android/build/outputs/apk/android-release-unsigned.apk
```



## Passo 3 – Entrar na bin do JDK do Java:

**Exemplo do meu mac:** /Library/Java/JavaVirtualMachines/1.8.0.jdk/Contents/Home

**Exemplo do meu windows:** C:\Program Files (x86)\Java\jdk1.8.0\_131\bin

```
cd "%JAVA_HOME%\bin" (Windows)
cd "$JAVA_HOME/bin" (Mac/Linux)
```

- Substitua a variável pelo caminho exemplo: cd C:\Program Files (x86)\Java\jdk1.8.0\_131\bin

## Passo 4 – Gerar keystore (Só gerar quando for subir a primeira vez na Playstore, depois você usará essa mesma chave para subir uma atualização)

Execute o seguinte comando como administrador:

```
keytool -genkey -v -keystore think.keystore -alias testapp-key -keyalg RSA -keysize 2048 -validity 10000
```

```
C:\Program Files (x86)\Java\jdk1.8.0_131\bin>keytool -genkey -v -keystore think.keystore -alias testapp-key -keyalg RSA -keysize 2048 -validity 10000
Informe a senha da área de armazenamento de chaves:
Informe novamente a nova senha:
Qual é o seu nome e o seu sobrenome?
[Unknown]: Felipe Albuquerque de Almeida
Qual é o nome da sua unidade organizacional?
[Unknown]: Developer
Qual é o nome da sua empresa?
[Unknown]: Think A.M.
Qual é o nome da sua Cidade ou Localidade?
[Unknown]: São Paulo
Qual é o nome do seu Estado ou Município?
[Unknown]: SP
Quais são as duas letras do código do país desta unidade?
[Unknown]: BR
CN=Felipe Albuquerque de Almeida, OU=Developer, O=Think A.M., L=São Paulo, ST=SP, C=BR Está correto?
[não]: sim

Gerando o par de chaves RSA de 2.048 bit e o certificado autoassinado (SHA256withRSA) com uma validade de 10.000 dias
para: CN=Felipe Albuquerque de Almeida, OU=Developer, O=Think A.M., L=São Paulo, ST=SP, C=BR
Informar a senha da chave de <testapp-key>
(RETURNE se for igual à senha da área de armazenamento de chaves):
Informe novamente a nova senha:
[Armazenando think.keystore]
```

## Passo 5 – Execute a ferramenta jarsigner que está inclusa na JDK

### Windows

```
jarsigner -verbose -sigalg SHA1withRSA -digestalg SHA1 -keystore think.keystore
"%APP_DIR%\platforms\android\build\outputs\apk\android-release-unsigned.apk"
testapp-key
```

### MAC/Linux

```
sudo jarsigner -verbose -sigalg SHA1withRSA -digestalg SHA1 -keystore think.keystore
$APP_DIR/platforms/android/build/outputs/apk/android-release-unsigned.apk testapp-key
```



Troque a variável APP\_DIR pelo diretório do seu app, exemplo:

```
jarsigner -verbose -sigalg SHA1withRSA -digestalg SHA1 -keystore think.keystore  
"C:\ThinkAcademy\github\Wakatime\ThinkAMWakatime\platforms\android\build\outputs\  
apk\android-release-unsigned.apk" testapp-key
```

```
signing: res/drawable-port-mdpi-v4/screen.png  
signing: res/drawable-port-xhdpi-v4/screen.png  
signing: res/drawable-port-xxhdpi-v4/screen.png  
signing: res/drawable-port-xxxhdpi-v4/screen.png  
signing: res/mipmap-hdpi-v4/icon.png  
signing: res/mipmap-ldpi-v4/icon.png  
signing: res/mipmap-mdpi-v4/icon.png  
signing: res/mipmap-xhdpi-v4/icon.png  
signing: res/mipmap-xxhdpi-v4/icon.png  
signing: res/mipmap-xxxhdpi-v4/icon.png  
signing: res/xml/config.xml  
signing: resources.arsc  
jar signed.
```



## Passo 6 – Entrar na pasta de build do Android:

### Windows

```
cd "%ANDROID_HOME%\build-tools\x.x.x"
```

### MAC/Linux

```
cd $ANDROID_HOME/build-tools/x.x.x
```

Troque a variável ANDROID\_HOME pelo diretório do seu app, exemplo:

```
cd C:\Program Files (x86)\Android\android-sdk\build-tools\25.0.3
```

## Passo 7 – Gerar o pacote final:

### Windows

```
.\zipalign -v 4 "%APP_DIR%\platforms\android\build\outputs\apk\android-  
release-unsigned.apk" "%APP_DIR%\platforms\android\build\outputs\apk\think-  
x.x.x.apk"
```

### Mac/Linux

```
sudo ./zipalign -v 4 $APP_DIR/platforms/android/build/outputs/apk/android-  
release-unsigned.apk $APP_DIR/platforms/android/build/outputs/apk/think-  
x.x.x.apk
```



Troque a variável APP\_DIR pelo diretório do seu app, exemplo:

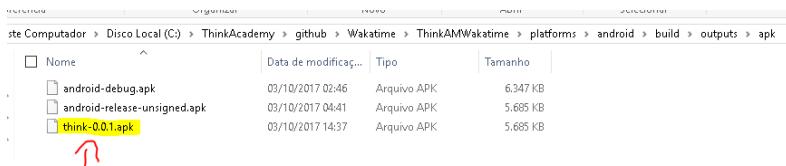
```
.\zipalign -v 4  
"C:\ThinkAcademy\github\Wakatime\ThinkAMWakatime\platforms\android\build\outputs\apk\android-release-unsigned.apk"  
"C:\ThinkAcademy\github\Wakatime\ThinkAMWakatime\platforms\android\build\outputs\apk\think-0.0.1.apk"
```

Notem que adicionei no nome do arquivo o mesmo número de versão que está em config.xml:

```
config.xml x  
1  <?xml version='1.0' encoding='utf-8'?>  
2  <widget id="net.thinkam.wakatimethinkam" version="0.0.1"  
3  |   <name>Wakatime – Think A.M.</name>  
4  |   <description>Um aplicativo incrível usando Ionic/Cordova</description>  
5  |   <author email="contato@thinkam.net" href="http://thinkam.net"></author>  
6  |   <content src="index.html" />
```

Esta nomenclatura do arquivo é uma preferência minha para uma melhor organização.

```
5583268 res/drawable-port-mdpi-v4/screen.png (OK)  
5592292 res/drawable-port-xhdpi-v4/screen.png (OK)  
5619972 res/drawable-port-xxhdpi-v4/screen.png (OK)  
5680704 res/drawable-port-xxxhdpi-v4/screen.png (OK)  
5763816 res/mipmap-hdpi-v4/icon.png (OK)  
5768576 res/mipmap-ldpi-v4/icon.png (OK)  
5769940 res/mipmap-mdpi-v4/icon.png (OK)  
5771892 res/mipmap-xhdpi-v4/icon.png (OK)  
5779148 res/mipmap-xxhdpi-v4/icon.png (OK)  
5791236 res/mipmap-xxxhdpi-v4/icon.png (OK)  
5808158 res/xml/config.xml (OK - compressed)  
5810480 resources.arsc (OK)  
Verification succesful
```



PRONTO!! TEMOS A NOSSA RELEASE NO JEITO PARA SUBIR NA PLAY STORE,  
AGORA TEMOS QUE FAZER O REGISTRO EM UMA CONTA DE  
DESENVOLVEDOR, POR ISSO SIGA OS PASSOS DESTE [LINK](#).



São 4 etapas muito simples:

#### Etapa 1: inscrever-se para uma conta de desenvolvedor do Google Play

1. Com sua Conta do Google, [inscreva-se para uma conta de desenvolvedor](#).
2. Depois de criar a conta de desenvolvedor, você poderá usar o Play Console para [publicar e gerenciar seus apps](#).

#### Etapa 2: aceitar o Contrato de distribuição do desenvolvedor

Durante o processo de inscrição, será preciso revisar e aceitar o [Contrato de distribuição do desenvolvedor do Google Play](#).

#### Etapa 3: pagar a taxa de inscrição

Há uma taxa de inscrição única de USD 25 que pode ser paga com um dos seguintes cartões de crédito ou débito:

- MasterCard
- Visa
- American Express
- Discover (somente nos EUA)
- Visa Electron (somente fora dos EUA)

**Observação:** os tipos de cartão aceitos variam de acordo com o local.

#### Etapa 4: inserir os detalhes da sua conta

Digite os detalhes de sua conta. Seu "Nome do desenvolvedor" será exibido aos clientes no Google Play.

É possível adicionar mais [informações da conta](#) após a criação da sua conta.

Após seguir essas 4 etapas acesse o [link para acessar o Google Play Console](#).

Faça login com a conta de desenvolvedor.

E escolha a opção “**CRIAR APP**”:



Esta quase acabando pessoal, e estou me dedicando Full Time a esse curso, por isso quero ver vocês me seguindo no Twitter:

Página Inicial    Moments    Notificações    Mensagens    Buscar no T

Tweets 6.989    Seguindo 1.156    Seguidores 467    Curtidas 245    Listas 1    Moments 0

**Felipe A. de Almeida**  
@felipealbuquerque  
CEO Think A.M.  
São Paulo - Brasil  
[thinkam.net](http://thinkam.net)

**Tweets**    **Tweets e respostas**    **Mídia**

Felipe A. de Almeida @felipealbuquerque · 6 min  
Nos últimos 15 dias já gravei mais de 40 vídeos, 58 páginas de documentação e 4865 palavras escritas (e aumentando).

Seguiu lá? Então beleza...

Quando você clicar em **CRIAR APP** vai aparecer assim ó:

Criar app

Idioma padrão \*

Português (Brasil) – pt-BR

Título \*

Think A.M.

10/50

CANCELAR    CRIAR

Coloquei o título como Think A.M. e cliquei em CRIAR.



Quando fiz isso apareceu um formulário de Detalhes do app (Nós vamos preencher somente os campos obrigatórios dessa vez):

Tirando o título são 9 campos para serem preenchidos, então força ai! Porque vale a pena...

O formulário de detalhes do app na Google Play Console. A seção 'Detalhes do produto' é visível. Os campos 'Breve descrição' (Português (Brasil) - pt-BR) e 'Descrição completa' (Português (Brasil) - pt-BR) estão circulados em vermelho.

O formulário de detalhes do app na Google Play Console com descrições preenchidas. As seções 'Breve descrição' e 'Descrição completa' estão circuladas em vermelho.

A seção 'Capturas de tela' no formulário de detalhes do app. O campo 'Capturas de tela' (Padrão – Português (Brasil) – pt-BR) está circulado em vermelho.

Eram pelo menos duas, mas fiz 5 capturas:



E-mail

Senha

LOGIN

CRIE UMA CONTA NA THINK A.M.

Nome

E-mail

Senha

Secret API Key Wakatime

ENVIAR

Bem vindo Felipe A. de Almeida ao Curso de Mobile Básico

Se você está perdido, veja os [docs](#) que vão te guiar.

Felipe A. de Almeida  
07/09/2016

www.thinkam.net

MARTINÓPOLIS, SP - BRAZIL

FELIPEALBUQUERQ VS CODE



**Ícone de alta resolução \***  
Padrão – Português (Brasil) – pt-BR  
512 x 512  
PNG de 32 bits (com alfa)

**Gráfico de recursos \***  
Padrão – Português (Brasil) – pt-BR  
1024 x 500 a  
JPG ou PNG de 24 bits (sem alfa)

**Gráfico promocional**  
Padrão – Português (Brasil) – pt-BR  
180 x 120 a  
JPG ou PNG de 24 bits (sem alfa)

**+**  
Adicionar ícone de alta resolução  
Solte uma imagem aqui

**+**  
Adicionar recurso gráfico  
Solte uma imagem aqui

**+**  
Adicionar gráfico promocional  
Solte uma imagem aqui

**Banner de TV**  
Padrão – Português (Brasil) – pt-BR  
1.280 (largura) x 720 (altura)  
JPG ou PNG de 24 bits (sem alfa)

**Imagen estereoscópica de 360º do Daydream**  
Padrão – Português (Brasil) – pt-BR  
4.096 (largura) x 4.096 (altura)  
JPG ou PNG de 24 bits (sem alfa)

**+**  
Adicionar banner de TV  
Solte uma imagem aqui

**+**  
Adicionar uma imagem estereoscópica de 360º  
Solte uma imagem

Para agilizar, no ícone fiz uma pesquisa no google que atende aos requisitos de ícone com exatamente 512X512. Fiz o mesmo com o gráfico de recursos indo em ferramentas e em tamanho colocando o tamanho exato que precisamos 1024X500, é claro que você pode criar a sua imagem com as características solicitadas, só estou querendo agilizar o processo por ser um aplicativo de teste.

**Ícone de alta resolução \***  
Padrão – Português (Brasil) – pt-BR  
512 x 512  
PNG de 32 bits (com alfa)

**Gráfico de recursos \***  
Padrão – Português (Brasil) – pt-BR  
1024 x 500 a  
JPG ou PNG de 24 bits (sem alfa)

**Gráfico promocional**  
Padrão – Português (Brasil) – pt-BR  
180 x 120 a  
JPG ou PNG de 24 bits (sem alfa)

**+**  
Adicionar gráfico promocional  
Solte uma imagem aqui

**Banner de TV**  
Padrão – Português (Brasil) – pt-BR  
1.280 (largura) x 720 (altura)  
JPG ou PNG de 24 bits (sem alfa)

**Imagen estereoscópica de 360º do Daydream**  
Padrão – Português (Brasil) – pt-BR  
4.096 (largura) x 4.096 (altura)  
JPG ou PNG de 24 bits (sem alfa)



Detalhes do app

Categorização

Tipos de app \*  

Categoria \*  

Classificação de conteúdo \*  

Você precisa preencher um questionário de classificação e aplicar uma [classificação de conteúdo](#).

Detalhes do contato

Site <http://www.thinkam.net>

Preenchemos o campo “Tipos de app” como **Apps** e “Categoria” como **Bibliotecas e demos**

Categorização

Tipos de app \* Apps

Categoria \* Bibliotecas e demos ↓

Classificação de conteúdo \* Você precisa preencher um questionário de classificação e aplicar uma [classificação de conteúdo](#).

Vamos ter que clicar neste link acima para preencher a classificação de conteúdo, mas vocês devem salvar as alterações na página antes disso, lá em cima no topo em **SALVAR RASCUNHO**:

Detalhes do app

Think A.M. Rascunho SALVAR RASCUNHO

PORTUGUÊS (BRASIL) - PT-BR   Gerenciar traduções

Os campos marcados com \* devem ser preenchidos antes de publicar.

Detalhes do produto

Título \* Português (Brasil) – pt-BR Think A.M. 10/50

Breve descrição \* Português (Brasil) – pt-BR Aplicativo criado no curso de Programação Mobile Básico



E quando for responder a classificação de conteúdo verá a seguinte mensagem:

The screenshot shows the 'Content rating' section of the Google Play developer console. It includes instructions about the IARC classification system, responsibilities of the developer, and how the rating will be used. A prominent red box highlights the message: 'É preciso fazer o upload de um APK antes de responder ao questionário de classificação de conteúdo.'

Portanto preencha o restante...

The screenshot shows the 'Privacy Policy' section where developers can enter their app's privacy policy URL. A red box highlights the URL input field.

Dependendo das permissões que o app pede vai ser necessária uma política de privacidade, mas nesse nosso exemplo está tranquilo.

E agora depois de SALVAR O RASCUNHO vamos no menu versões de apps:

The screenshot shows the 'App Versions' menu. A red box highlights the 'Content Rating' option under the 'Classificação de conteúdo' heading.

Notem que teremos que percorrer todos os menus que estão com o ícone de atenção, mas não se afobe porque é tranquilo!



Em versões de apps vamos na opção Gerenciar Alfa:

The screenshot shows the 'Versões de apps' (Versioning) screen in the Google Play Developer Console. At the top, there's a search bar and some navigation icons. Below it, there are two main sections: 'Beta' and 'Alfa'. The 'Beta' section has a button labeled 'GERENCIAR BETA'. The 'Alfa' section has a button labeled 'GERENCIAR ALFA', which is highlighted with a red circle. Both sections contain instructions for adding APKs to their respective tracks.

Essa é a nossa primeira subida para a Play Store, então eu recomendo que seja em Alfa, para não haver nenhuma crítica pública ou uma avaliação negativa ao seu aplicativo sem antes testarmos e melhorarmos.

The screenshot shows the 'Alfa' version management screen. It includes a section for managing testers ('Gerenciar testadores') with a note that an APK needs to be uploaded before configuring testers. Below this, there's a message about an unlaunched version and a blue 'EDITAR VERSÃO' button. A blue arrow points downwards towards this button.

Após clicar em Editar Versão vá em Continuar

The screenshot shows the 'Lançamento para Alfa' (Release to Alpha) screen. It displays a progress bar with step 1 (Preparar versão) and step 2 (Analizar e lançar). A note about Google Play App Signing is present, mentioning that clicking 'Continuar' inscribes the app into the service. A red arrow points to the 'CONTINUAR' button. Other options like 'REUTILIZAR CHAVE DE ASSINATURA' and 'ADICIONAR UM APK DA BIBLIOTECA' are also visible.



## Aceite o contrato

sem ativar a participação no GPAS.

**4. Otimizações opcionais do app**

4.1. O Google pode oferecer a Você otimizações de app diferentes das otimizações automáticas referidas na Seção 2. Você pode optar por aplicar essas otimizações aos apps que estiverem inscritos no GPAS.

4.2. Você não é obrigado a aceitar as otimizações opcionais do app.

4.3. Se optar por aplicar uma otimização opcional de app, Você poderá desativar essa otimização opcional a qualquer momento.

**5. Alterações no Contrato**

5.1. O Google pode fazer alterações nestes termos a qualquer momento, desde que envie a Você um aviso com antecedência descrevendo o que foi mudado. O Google também divulgará uma notificação no Google Play Console com a descrição dessas modificações. Elas entram em vigor e são consideradas aceitas por Você: (a) imediatamente para aqueles que ativaram o GPAS após o envio da notificação ou (b) para usuários que já haviam ativado o GPAS na data especificada no aviso. Caso não concorde com as modificações dos Termos, Você precisará obrigatoriamente sair do GPAS, sujeito à Seção 3, que será Sua medida judicial única e exclusiva. Você concorda que, ao não abandonar o GPAS, aceita os termos alterados.

© Google · Privacidade e Termos · AjudaAlterar o idioma ou a região: Português (Brasil) – Brasil

[SAIBA MAIS](#) [CANCELAR](#) [ACEITAR](#)

Clique em Procurar Arquivos para selecionar a APK que geramos como release

Versões de apps

Lançamento para Alfa

Preparar versão Analisar e lançar

Google Play App Signing

Ativado

APKs a serem adicionados

Estes APKs serão veiculados na Google Play Store após o lançamento desta versão.

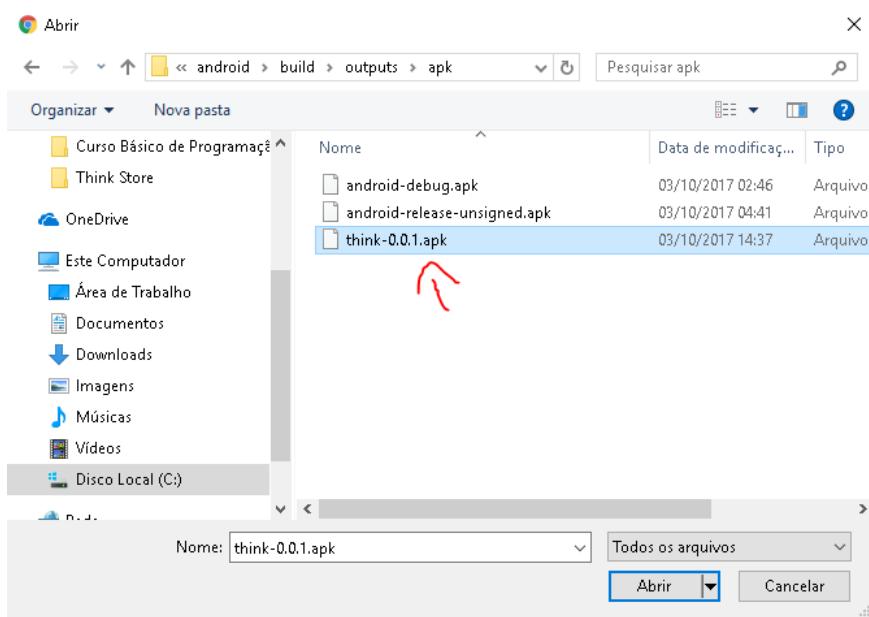
ADICIONAR UM APK DA BIBLIOTECA

PROCURAR ARQUIVOS

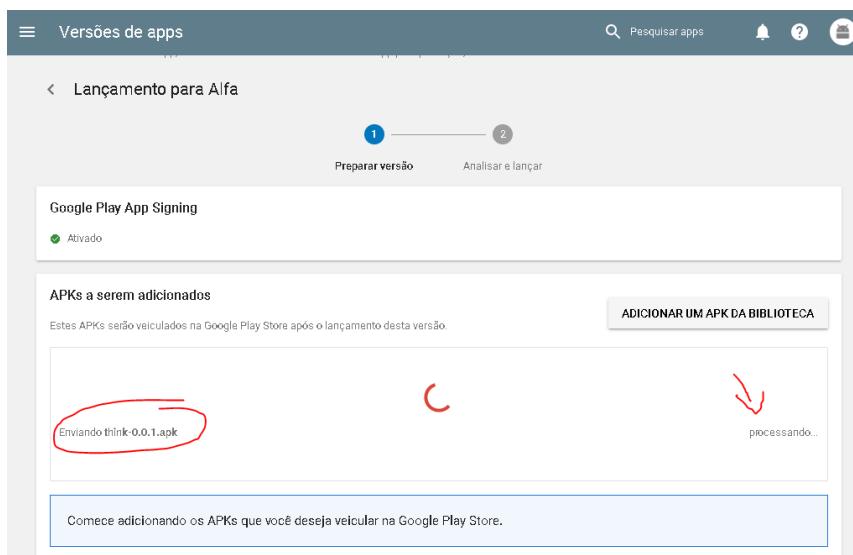
Comece adicionando os APKs que você deseja veicular na Google Play Store.



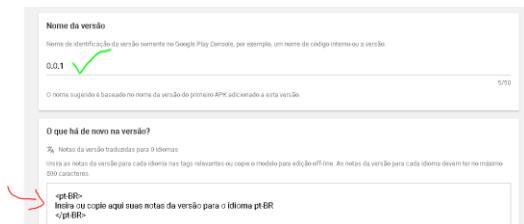
Selecione o arquivo (exemplo think-0.0.1.apk)



O console da Google Play irá processar para saber se está tudo ok com a nossa APK



Estará OK porque seguimos todos os passos anteriores que mostrei aqui na geração da APK



Você poderá colocar uma descrição do APP dizendo o que existe de novo nessa versão



Coloquei a seguinte descrição

O que há de novo na versão?

Notas da versão traduzidas para 0 idiomas

Insera as notas da versão para cada idioma nas tags relevantes ou copie o modelo para edição off-line. As notas da versão para cada idioma devem ter no máximo 500 caracteres.

```
<pt-BR>
Aplicação de exemplo do curso de mobile básico
</pt-BR>
```

DESCARTAR SALVAR REVISAR

E cliquei em “Salvar” e depois “Revisar”

As alterações foram salvas.

0.0.1

O nome sugerido é baseado no nome da versão do primeiro APK adicionado à esta versão.

O que há de novo na versão?

Notas da versão traduzidas para 0 idiomas

Insera as notas da versão para cada idioma nas tags relevantes ou copie o modelo para edição off-line. As notas da versão para cada idioma devem ter no máximo 500 caracteres.

```
<pt-BR>
Aplicação de exemplo do curso de mobile básico
</pt-BR>
```

Foi inserida a tradução de 1 idioma: pt-BR

DESCARTAR SALVO REVISAR

Notem que ainda temos algumas etapas pendentes, e se tentarmos iniciar o lançamento para Alfa vamos ter uma notificação dizendo para concluir os outros passos:

Google Play Console

Versões de apps

Confirmar lançamento para versão Alfa: 0.0.1

Preparar versão Análise e lancer

Resumo de resenhas

Este versão está pronta para ser lançada.

APKs nesta versão

Código da versão	Enviados	Instalações em dispositivos ativos
1 APK adicionado	1	2 minutos atrás Nenhuma

O que há de novo na versão?

Notas - Português (Brasil) - pt-BR  
Aplicação de exemplo do curso de mobile básico

Traduzido em 1 idioma

INICIAR O LANÇAMENTO PARA ALFA



Um deles é a Classificação de conteúdo, clique em CONTINUAR:

O sistema de classificação de conteúdo para apps e jogos do Google Play foi projetado para oferecer classificações reconhecidas e localmente relevantes a usuários de todo o mundo. Esse sistema inclui classificações oficiais da Coalizão Internacional de Classificação Etária (IARC, na sigla em inglês) e de seus órgãos participantes (veja os Termos de uso deles).  
Responsabilidades do desenvolvedor:

- Preencher o questionário de classificação do conteúdo para cada nova app enviada ao Play Console, para todas as apps ativas no Google Play e para todas as atualizações de apps nas quais houve uma alteração no conteúdo ou na versão que afeta a resposta ao questionário.
- Fornecer respostas precisas para o questionário de classificação do conteúdo. A desinformação do conteúdo da seu app resultará na remoção ou na suspensão da app.

Sua classificação será usada para:

- informar os consumidores sobre a classificação etária da seu app;
- bloquear ou filtrar seu conteúdo em determinadas territórios ou para usuários específicos onde isso seja exigido legalmente;
- avaliar a qualificação da seu app para programas especiais voltados a desenvolvedores.

O questionário de classificação do conteúdo e as novas diretrizes relacionadas são exigências para sua participação na Google Play Store. [Saiba mais](#)

**CONTINUAR**

IARC

Coloquei meu e-mail pessoal aqui, sigam como quiserem aí...

Bem-vindo ao questionário de classificação de conteúdo

O sistema de classificação de conteúdo para apps e jogos do Google Play foi projetado para oferecer classificações reconhecidas e localmente relevantes a usuários de todo o mundo. Esse sistema inclui classificações oficiais da Coalizão Internacional de Classificação Etária (IARC, na sigla em inglês) e de seus órgãos participantes (veja os [Termos de uso deles](#)). Comece inserindo o endereço de e-mail que você deseja que a IARC use para comunicações relacionadas à classificação.

Endereço de e-mail \*

f.albuquerquedealmeida@gmail.com

Confirmar endereço de e-mail \*

f.albuquerquedealmeida@gmail.com

Selezione a categoria do app

Em seguida selecionaremos a categoria do app, eu escolhi essa, porém façam de acordo com o que vocês querem inserir no seu app.

Conclua o questionário para que possamos calcular a classificação da seu app.

**REDES SOCIAIS, FÓRUNS, BLOGS E COMPARTILHAMENTO DE CONTEÚDO GERADO PELO USUÁRIO (UGC)**  
A aplicação é uma aplicação de rede social. [Editar categoria](#)

**TIPO DE APLICAÇÃO**

O foco principal da aplicação é conectar pessoas para fins de namoro, empenhos ou relacionamentos sociais? \* [Saiba mais](#)

Sim  Não

O aplicativo permite o compartilhamento público de nudez? \*

Sim  Não

O aplicativo permite o compartilhamento público de violência gráfica do mundo real fora do contexto notícias? \*

Sim  Não

O aplicativo compartilha a localização física atual do usuário com outras pessoas? \* [Saiba mais](#)

Sim  Não

A aplicação permite aos utilizadores comprar bens digitais? \* [Saiba mais](#)

Sim  Não

**CALCULAR CLASSIFICAÇÃO** **SALVAR QUESTIONÁRIO**

IARC



Classificação de conteúdo

Pesquisar apps

Think A.M. Rascunho SALVAR RASCUNHO

Conclua o questionário para que possamos calcular a classificação do seu app.

REDES SOCIAIS, FÓRUMS, BLOGS E COMPARTILHAMENTO DE CONTEÚDO GERADO PELO USUÁRIO (UGC)

A aplicação é uma aplicação de rede social. Editar categoria

TIPO DE APLICAÇÃO

EDITAR ✓

CALCULAR CLASSIFICAÇÃO SALVAR QUESTIONÁRIO IARC

Salve o questionário e calcule a classificação:

Suas respostas ao questionário foram salvas.

Think A.M. Rascunho SALVAR RASCUNHO

Conclua o questionário para que possamos calcular a classificação do seu app.

REDES SOCIAIS, FÓRUMS, BLOGS E COMPARTILHAMENTO DE CONTEÚDO GERADO PELO USUÁRIO (UGC)

A aplicação é uma aplicação de rede social. Editar categoria

TIPO DE APLICAÇÃO

EDITAR ✓

CALCULAR CLASSIFICAÇÃO SALVO IARC

Suas respostas ao questionário foram salvas.

Google Play Rússia (12) Orientação Parental Recomendada 12+ anos

Google Play Coréia do Sul (12) Orientação Parental Recomendada 12+ anos

Elementos Interativos Usuários Interagem

A classificação calculada mostrada acima pode não ser a mesma exibida aos usuários na Google Play Store.

- O Google pode rejeitar a atualização ou o envio do seu app devido a uma interpretação errada do conteúdo do app.
- O Google pode usar suas respostas para o questionário a fim de gerar classificações a determinados territórios conforme exigido pela lei local.
- As autoridades de classificação da Coalizão Internacional de Classificação Etária (IARC, na sigla em inglês) podem alterar a classificação do seu app após a análise.
- O Google e a IARC compartilham seus dados de contato, as respostas para o questionário, as classificações, as solicitações de suporte ao desenvolvedor e os detalhes do app com as autoridades de classificação participantes.
- Consulte a Central de Ajuda para mais informações sobre processos e diretrizes de classificação.

APLICAR CLASSIFICAÇÃO VOLTA IARC

Classificação de conteúdo

Pesquisar apps

Think A.M. Pronto para publicar SALVAR RASCUNHO Pronto para publicar

CLASSIFICAÇÃO APLICADA Enviado em: segundos atrás Visualizar detalhes Saiba mais

12 T ! 12 12 12 12

Questionários

Data	Detificado da IARC	Email	
segundos atrás	fabuequejudealmend... Editar	VISUALIZAR DETALHES	

Saiba mais sobre o questionário de classificação de conteúdo no Google Play [aqui](#). Envie um novo questionário de classificação de conteúdo para todas as atualizações de apps nas quais houve alteração no conteúdo do app ou em recursos que afetarem suas respostas ao questionário.

COMEÇAR NOVO QUESTIONÁRIO



## Feito isso, vamos para Preços e distribuição

Preços e distribuição

Pesquisar apps

OCULTAR PAÍSES

Disponível em 1 país (somente Alfa ou Beta).

Países *	Não disponível	Disponível
Bielorrússia	<input checked="" type="radio"/>	<input type="radio"/>
Bolívia	<input checked="" type="radio"/>	<input type="radio"/>
Bósnia e Herzegovina	<input checked="" type="radio"/>	<input type="radio"/>
Botswana	<input checked="" type="radio"/>	<input type="radio"/>
Brasil	<input type="radio"/>	<input checked="" type="radio"/>
Bulgária	<input checked="" type="radio"/>	<input type="radio"/>
Burquina Faso	<input checked="" type="radio"/>	<input type="radio"/>
Cambodja	<input checked="" type="radio"/>	<input type="radio"/>

Mostrar opções

Brasil →

↓

Preços e distribuição

Pesquisar apps

Criado para o público infantil \*

Seu app foi criado para o público infantil com menos de 13 anos conforme definido pela COPPA?

Sim  
 Não

Caso seu app seja direcionado ao público infantil, você precisará ativar o programa Feito para família abaixo.

Contém anúncios \*

Seu app tem anúncios? Além disso, consulte nossa política de anúncios para evitar violações comuns. Caso a resposta seja "Sim", os usuários verão o rótulo anúncios no seu app na Play Store. Saiba mais

Sim, ele tem anúncios  
 Não, ele não tem anúncios

Preços e distribuição

Pesquisar apps

Desativação de marketing

Não divulgar meu app, exceto no Google Play e em qualquer propriedade do Google de propriedade do Google on-line para celular. Entendo que qualquer alteração nesta preferência poderá levar 60 dias para surtir efeito.

Diretrizes de conteúdo \*

Esta app está em conformidade com as diretrizes de conteúdo do Android.

Confira estas dicas sobre como criar descrições de apps em conformidade com as políticas para evitar alguns dos motivos comuns para a suspensão de apps. Caso seu app ou a página de detalhes estejam qualificados para aviso antecipado à equipe de análise de apps do Google Play, entre em contato conosco antes de fazer a publicação.

Leis de exportação dos EUA \*

Eu reconheço que meu app de software pode estar sujeito às leis de exportação dos Estados Unidos, independentemente da minha localização ou nacionalidade. Afirmo ter cumprido a todas essas leis, incluindo eventuais requisitos de software com funções de criptografia. Certifico que meu app está autorizado a ser exportado dos Estados Unidos ao abrigo dessas leis. Saiba mais



Preços e distribuição

Think A.M. Rascunho SALVAR RASCUNHO

Este app é PAGOS GRATUITO

Disponibilidade do app

Paises \*

Agora se acessarmos o painel com todos os apps teremos o seguinte:

Google Play Console

Todos os apps

Painel

Estatísticas

Android vitais

Ferramentas de desenvolvimento

Filtrar CRIAR APP

Nome do app	Instalações ativas / totais	Aval. média / Total	Última atualização	Status
Think A.M. net.thinkam.wakatimethinkam	-	★ -	-	Pronto para publicar



Clique no link Pronto para publicar:

The screenshot shows the 'Detalhes do app' (App Details) section of the Think A.M. app. At the top, there's a navigation bar with a search icon, a bell icon, a question mark icon, and a user profile icon. Below the navigation, the app name 'Think A.M.' is displayed with a green status indicator 'Pronto para publicar'. A green arrow points to this button. To the right of the button is a blue button labeled 'PORTUGUÊS (BRASIL) – PT-BR'. Further down, there are sections for 'Título' (Title), 'Breve descrição' (Brief Description), and 'Descrição completa' (Full Description). Each section has a placeholder text and a character count limit.

Ele te dará uma mensagem como esta:

This screenshot shows a confirmation message. It starts with 'Seu app está pronto para ser publicado' (Your app is ready to be published). Below it, a note says 'Agora você pode publicar seu app começando o lançamento de uma versão na página "Gerenciar versões"' (Now you can publish your app by starting the launch of a version on the "Manage versions" page). At the bottom, there are two buttons: 'DISPENSAR' (Dispense) and 'GERENCIAR VERSÕES' (Manage Versions), with the latter being highlighted by a large green circle.

Edite a versão:

This screenshot shows the 'Versões de apps' (App Versions) page. At the top, there's a navigation bar with a search icon, a bell icon, a question mark icon, and a user profile icon. Below the navigation, the section title 'Alfa' is shown with a green status indicator 'GERENCIAR ALFA'. A green circle highlights the 'EDITAR VERSÃO' (Edit Version) button. Below this, there's a section titled 'Serviço de tradução' (Translation Service) with a globe icon and some descriptive text. At the bottom of this section is a blue button labeled 'INICIAR OU CONFERIR PROGRESSO' (Start or Check Progress).



## Clique em Revisar

O que há de novo na versão?

Notas da versão traduzidas para 1 idioma

Insira as notas da versão para cada idioma nas tags relevantes ou copie o modelo para edição off-line. As notas da versão para cada idioma devem ter no máximo 500 caracteres.

<pt-BR>  
Aplicação de exemplo do curso de mobile básico  
</pt-BR>

Foi inserida a tradução de 1 idioma: pt-BR

DESCARTAR SALVO REVISAR

© 2017 Google · Aplicativo para dispositivos móveis · Ajuda · Termos do site · Privacidade · Contrato de distribuição do desenvolvedor

## E depois é só iniciar o lançamento para Alfa

Resumo de resenhas

Esta versão está pronta para ser lançada.

APKs nesta versão

Código da versão	Enviados	Instalações em dispositivos ativos
1 APK adicionado	1 40 minutos atrás	Nenhum dado

O que há de novo na versão?

Padrão – Português (Brasil) – pt-BR  
Aplicação de exemplo do curso de mobile básico

Tradução em 1 idioma

ANTERIOR DESCARTAR INICIAR O LANÇAMENTO PARA ALFA

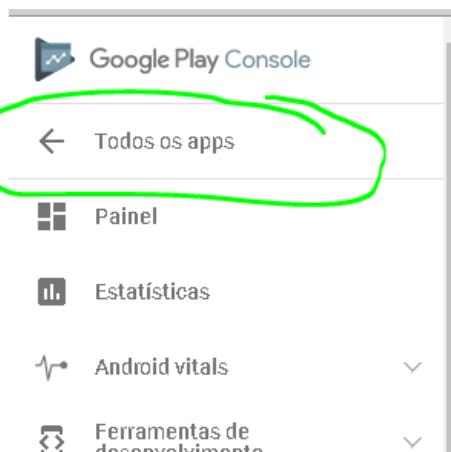
## Confirme

Seu app será disponibilizado somente para testadores Alfa. Deseja continuar?

CANCELAR CONFIRMAR



E em todos os apps



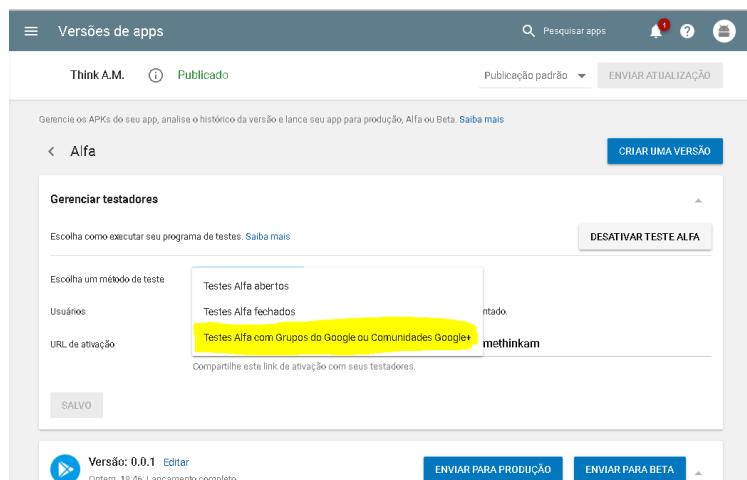
Teremos a seguinte informação, ao clicar no ícone **verde**:



Ele nos direciona ao “Painel” que mostra que está “**Processando atualização**”



Existe um detalhe importante agora no gerenciamento do Alfa quando já estiver publicado, que é definir como será o método de teste do aplicativo:



Vou escolher este método porque será mais fácil de fazer com que vocês tenham acesso ao app depois.



Para isso, vocês terão que solicitar a participação na comunidade a seguir que servirá para vocês terem contato e quem sabe encontrar alguém próximo de vocês que já fez o curso e quer entrar em um projeto em conjunto.

No print acima eu havia acabado de criar a comunidade com o usuário da Think A.M., em seguida convidei o meu usuário pessoal (Felipe Albuquerque de Almeida) para participar, e já vinculei a comunidade com o app, salvei a configuração abaixo:

E postei conforme imagem abaixo com o usuário Felipe, o link de ativação para ser tester do app da Think A.M. feito durante o curso, notem que ao clicar no link e confirmar a participação, chega uma mensagem no e-mail com as informações:



Seguro | https://play.google.com/store/apps/details?id=net.thinkam.wakatimethinkam

Google Play Pesquisar

Apps Categorias Página inicial Mais pesquisados Lançamentos

Meus apps Comprar

Jogos Família Escolha dos editores

Conta Resgatar Comprar vale-presente Minha lista de desejos Minha atividade Play Guia para pais

Think A.M.

Think A.M. - Bibliotecas e demos

Este app é compatível com todos os seus dispositivos.

Instalado

Este app conta com Login e Registro na Think A.M. e uma integração de exemplo ao Wakatime que inicialmente só demonstra a exibição dos dados de cadastro do usuário atual.

RESENHAS Escrever uma resenha

NOVIDADES Aplicação de exemplo do curso de mobile básico

INFORMAÇÕES ADICIONAIS

Atualizado	Requer Android	Classificação do conteúdo
3 de outubro de 2017	4.1 ou superior	Classificação 12 anos Línguagem Imprópria Saiba mais
Elementos interativos	Permissões	Reportar
Usuários Interagem	Ver detalhes	Sinalizar como impróprio
Oferido por	Desenvolvedor	
Think A.M.	Acesse o site E-mail felipe.almeida@thinkam.net	