



Flutter
Talks 2019

Flutter e IoT

Estudo sobre o uso de Internet
das Coisas na plataforma Flutter

Sobre o Palestrante



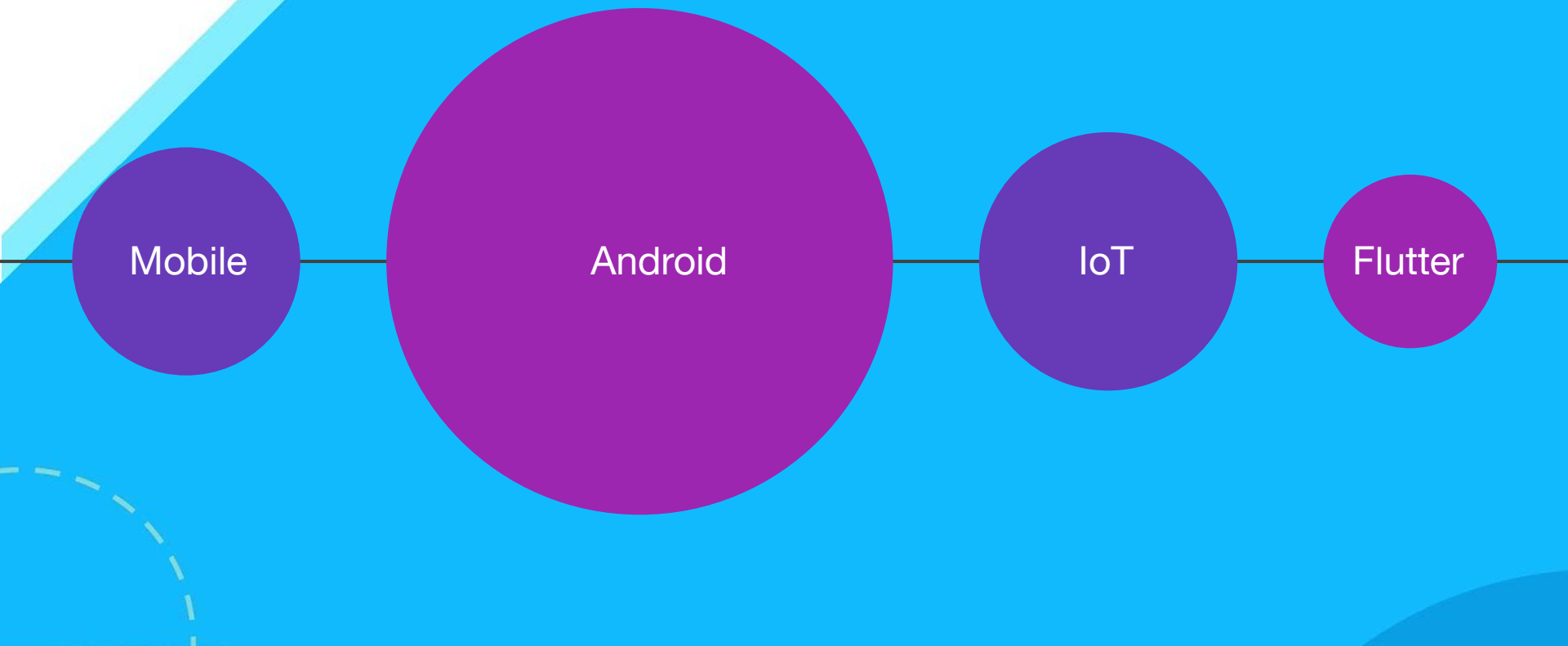
- Ricardo da Silva Ogliari
- Graduação Ciência da Computação
- Pós Web: Estratégias de Inovação e Tecnologia
- MBA: Jogos e Aplicativos Móveis
- Bradoo Technologies / Mobilearn
- Co-autor do livro Android: do Básico ao Avançado. Autor do livro Internet das Coisas para Desenvolvedores
- Things Hacker Team
- Vice-Campeão de Chula. Vice-Campeão de KickBox K-1. PF Fight.



Cronograma

- Introdução
- Publisher Subscriber
- FCM
- MQTT
- PubNub
- Twilio

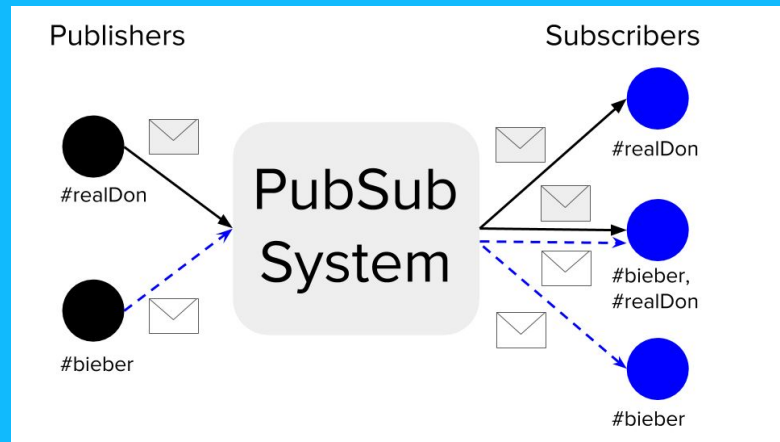
Introdução



Publisher-Subscriber

Publicadores que enviam mensagem para um ponto central, um broker, um event bus, ou algo do gênero. Este sistema central encaminha aos interessados, aos assinantes daquele broker.

PERFEITO pra IoT.

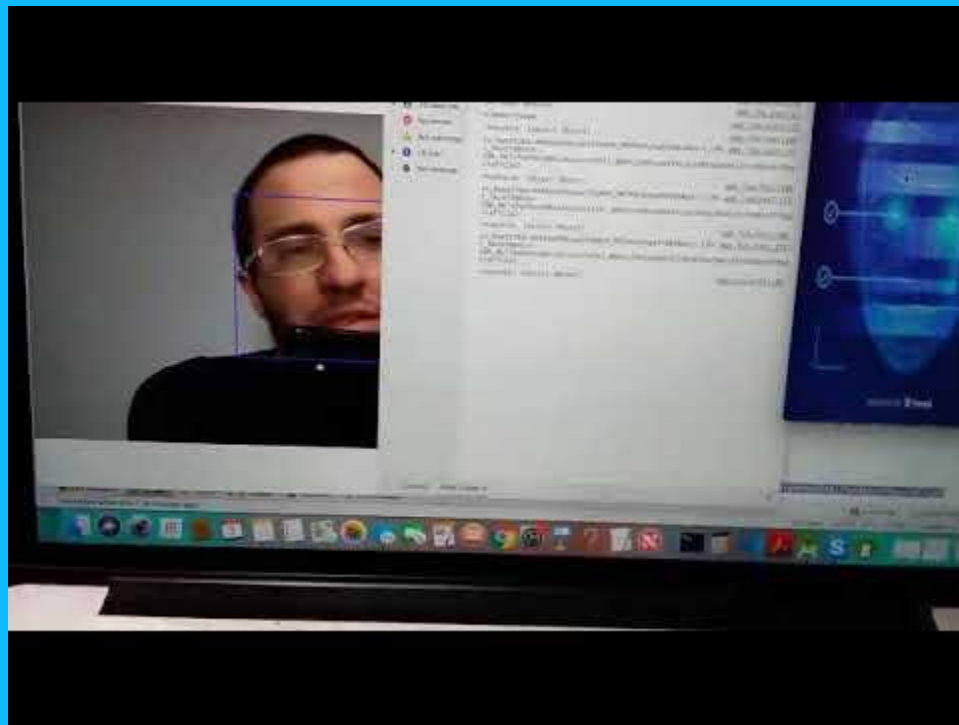




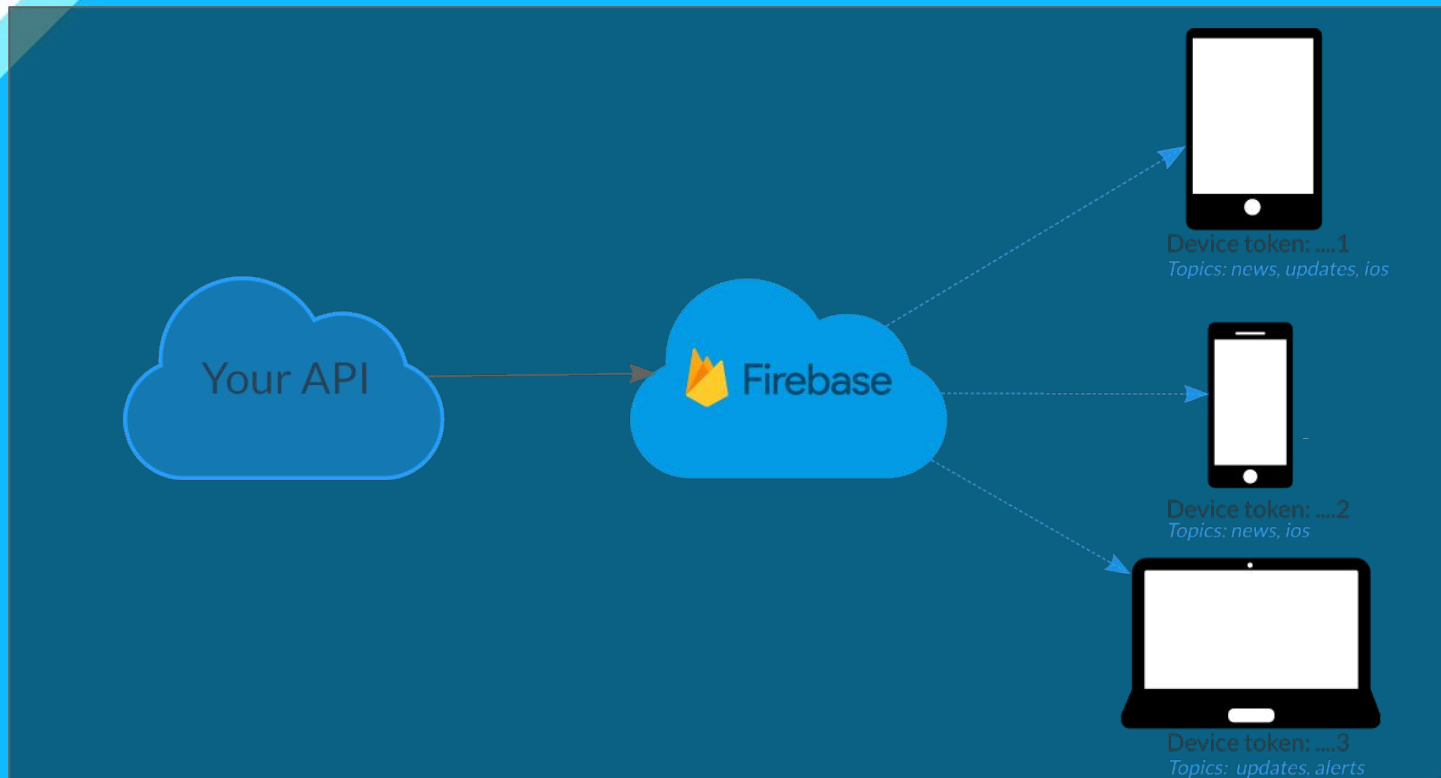
Flutter
Talks 2019

Flutter Cloud Messaging

Solução



Solução



Detalhes - 1

```
<< pubspec.yaml >>
```

```
dependencies:
```

```
  firebase_messaging: ^5.1.2
```

```
assets:
```

```
  - assets/livia.jpg
```

Detalhes - 2

```
<< [project]/android/build.gradle >>
```

```
...
```

```
dependencies {
```

```
  classpath 'com.android.tools.build:gradle:3.2.1'
```

```
  classpath 'com.google.gms:google-services:4.3.0'
```

```
}
```

```
...
```

Detalhes - 3

```
<< [project]/android/app/build.gradle >>
```

```
...
```

```
// ADD THIS AT THE BOTTOM  
apply plugin: 'com.google.gms.google-services'
```

Detalhes - 4

<< *AndroidManifest.xml* >>

...

```
<intent-filter>
  <action android:name="android.intent.action.MAIN"/>
  <category android:name="android.intent.category.LAUNCHER"/>
  <action android:name="FLUTTER_NOTIFICATION_CLICK" />
  <category android:name="android.intent.category.DEFAULT" />
</intent-filter>
```

Detalhes - 5

<< *arquivo dart* >>

```
_firebaseMessaging.getToken().then(...);  
_firebaseMessaging.configure(  
  onMessage: (Map<String, dynamic> message) async {  
    for (var keys in message.keys) {  
      if (keys == "data") {  
        String valueData = message[keys].toString();  
        if (valueData.indexOf("ricardo") > 0) ...  
      }  
    }  
  },  
  onResume: (Map<String, dynamic> message) async {},  
  onLaunch: (Map<String, dynamic> message) async {},  
);
```

Detalhes - 6

	App in Foreground	App in Background	App Terminated
Notification on Android	<code>onMessage</code>	Notification is delivered to system tray. When the user clicks on it to open app <code>onResume</code> fires if <code>click_action: FLUTTER_NOTIFICATION_CLICK</code> is set (see below).	Notification is delivered to system tray. When the user clicks on it to open app <code>onLaunch</code> fires if <code>click_action: FLUTTER_NOTIFICATION_CLICK</code> is set (see below).
Notification on iOS	<code>onMessage</code>	Notification is delivered to system tray. When the user clicks on it to open app <code>onResume</code> fires.	Notification is delivered to system tray. When the user clicks on it to open app <code>onLaunch</code> fires.
Data Message on Android	<code>onMessage</code>	<code>onMessage</code> while app stays in the background.	<i>not supported by plugin, message is lost</i>
Data Message on iOS	<code>onMessage</code>	Message is stored by FCM and delivered to app via <code>onMessage</code> when the app is brought back to foreground.	Message is stored by FCM and delivered to app via <code>onMessage</code> when the app is brought back to foreground.

Save me life

Android Studio - flutter_app_fcm

Web:

https://flutter-fcm-84fd3.firebaseio.com/web_fcm.htm



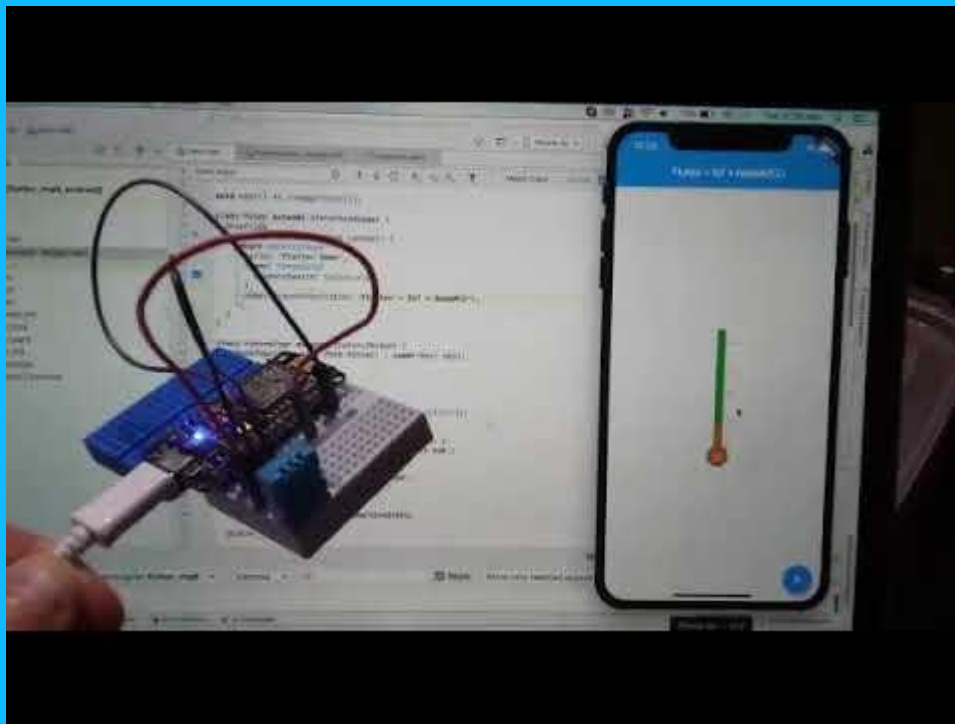
Flutter
Talks 2019

MQTT “Puro”

MQTT

O MQTT foi inventado e desenvolvido inicialmente pela IBM no **final dos anos 90**. Sua aplicação original era vincular sensores em pipelines de petróleo a satélites. Como seu nome sugere, ele é um protocolo de mensagem com suporte para a comunicação assíncrona entre as partes. Um protocolo de sistema de mensagens assíncrono **desacopla o emissor e o receptor** da mensagem tanto no espaço quanto no tempo e, portanto, é escalável em ambientes de rede que não são confiáveis. Apesar de seu nome, ele não tem nada a ver com filas de mensagens, na verdade, **ele usa um modelo de publicação e assinatura**. No final de 2014, ele se tornou oficialmente um padrão aberto OASIS, com suporte nas linguagens de programação populares, usando diversas implementações de software livre.

Solução



Solução



Hosted message broker for the
Internet of Things

Perfectly configured and optimized message queues for IoT, ready in seconds.



Detalhes - 1

```
<< pubspec.yaml >>
```

```
dependencies:
```

```
  mqtt_client: ^5.5.3
```

```
assets:
```


```
  - assets/livia.jpg
```

Detalhes - 2


<https://github.com/NPKompleet/ThermometerWidget>



Detalhes - 3

 **CloudMQTT**

Flutter IoT ▾

 rogliariping@gmail.com ▾

DETAILS

SETTINGS

CERTIFICATES

USERS & ACL

BRIDGES

AMAZON KINESIS STREAM

WEBSOCKET UI


STATISTICS

CONNECTIONS


LOG

Details

Instance info

Server	postman.cloudmqtt.com	
User	fcgymrix	Restart
Password	0qViGj... 	Refresh
Port	13370	
SSL Port	23370	
Websockets Port (TLS only)	33370	
Connection limit	5	

Active Plan



Cute Cat

[Upgrade Instance](#)

Detalhes - 4

```
String broker                =  
'postman.cloudmqtt.com';  
int port                     = 13370;  
String username              =  
'fcg.._seu_user_no_brokerrix';  
String passwd                = '0qVi...seu_pass';  
String clientIdentifier     = 'android';  
  
mqtt.MqttClient client;  
mqtt.MqttConnectionState connectionState;  
  
StreamSubscription subscription;
```

Detalhes - 5

```
void _connect() async {  
  client = mqtt.MqttClient(broker, '');  
  client.port = port;  
  
  try {  
    await client.connect(username, passwd);  
  } catch (e) { }  
  
  subscription = client.updates.listen(_onMessage);  
  _subscribeToTopic("temp");  
}
```


Detalhes - 6

```
void _onMessage(List<mqtt.MqttReceivedMessage> event) {  
    final mqtt.MqttPublishMessage recMess =  
        event[0].payload as mqtt.MqttPublishMessage;  
    final String msg =  
        mqtt.MqttPublishPayload.bytesToStringAsString(recMess.payload.  
            message);  
  
    setState(() {  
        _temp = double.parse(msg);  
    });  
}
```

Save me Life

Arduino - `node_mcu_mqtt`

Android Studio - `flutter_mqtt`

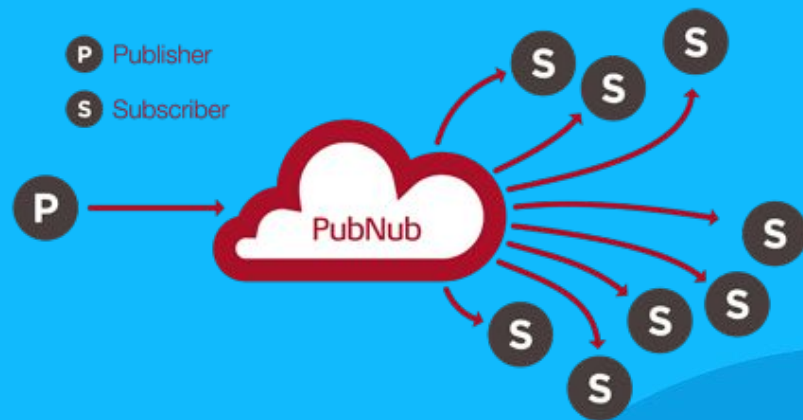


Flutter
Talks 2019

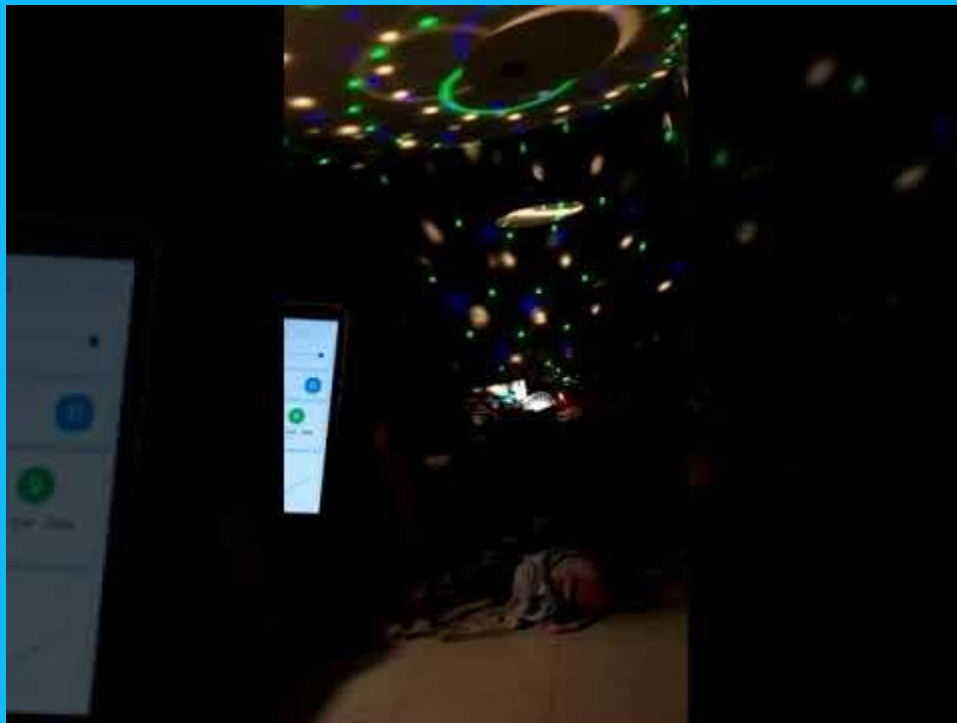
PubNub

PubNub

Plataforma para Publisher-Subscriber com suporte a um grande número de linguagens e plataformas.



Solução



Detalhes - 1

<< *pubspec.yaml* >>

dependencies:

`flutter_android: ^0.4.0`

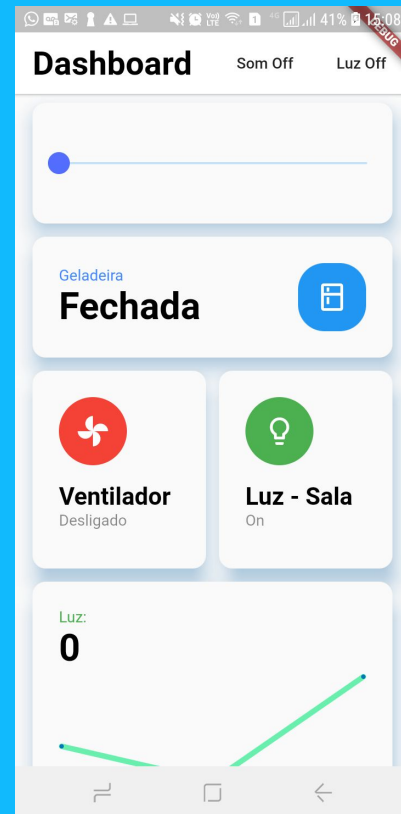
`http: ^0.12.0+2`

`flutter_staggered_grid_view: ^0.3.0`

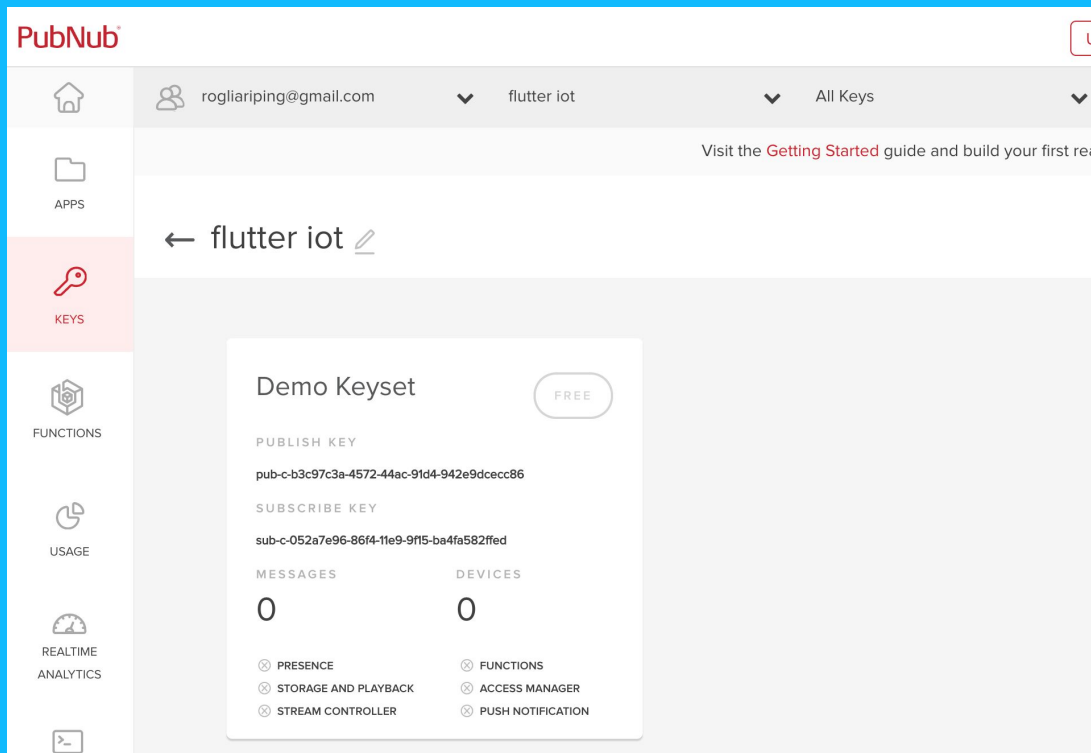
`flutter_sparkline: 0.1.0`

`assets_audio_player: 1.0.1`

`volume: 0.1.0`



Detalhes - 2



The screenshot shows the PubNub web interface for a Flutter IOT project. The left sidebar contains navigation links: Home, APPS, KEYS (highlighted), FUNCTIONS, USAGE, REALTIME ANALYTICS, and a terminal icon. The top header shows the user 'rogliariping@gmail.com' and the project 'flutter iot'. A banner at the top right encourages visiting the 'Getting Started' guide. The main content area displays the 'flutter iot' keyset details, including a 'Demo Keyset' label with a 'FREE' badge. Below this, it lists the 'PUBLISH KEY' (pub-c-b3c97c3a-4572-44ac-91d4-942e9dcecc86) and the 'SUBSCRIBE KEY' (sub-c-052a7e96-86f4-11e9-9f15-ba4fa582ffed). It also shows 'MESSAGES' and 'DEVICES' counts, both at 0. At the bottom, there are two columns of features: PRESENCE, STORAGE AND PLAYBACK, and STREAM CONTROLLER on the left; and FUNCTIONS, ACCESS MANAGER, and PUSH NOTIFICATION on the right.

PubNub

rogliariping@gmail.com flutter iot All Keys

Visit the [Getting Started](#) guide and build your first real-time app.

← flutter iot

Demo Keyset FREE

PUBLISH KEY

pub-c-b3c97c3a-4572-44ac-91d4-942e9dcecc86

SUBSCRIBE KEY

sub-c-052a7e96-86f4-11e9-9f15-ba4fa582ffed

MESSAGES 0 DEVICES 0

☒ PRESENCE ☒ FUNCTIONS

☒ STORAGE AND PLAYBACK ☒ ACCESS MANAGER

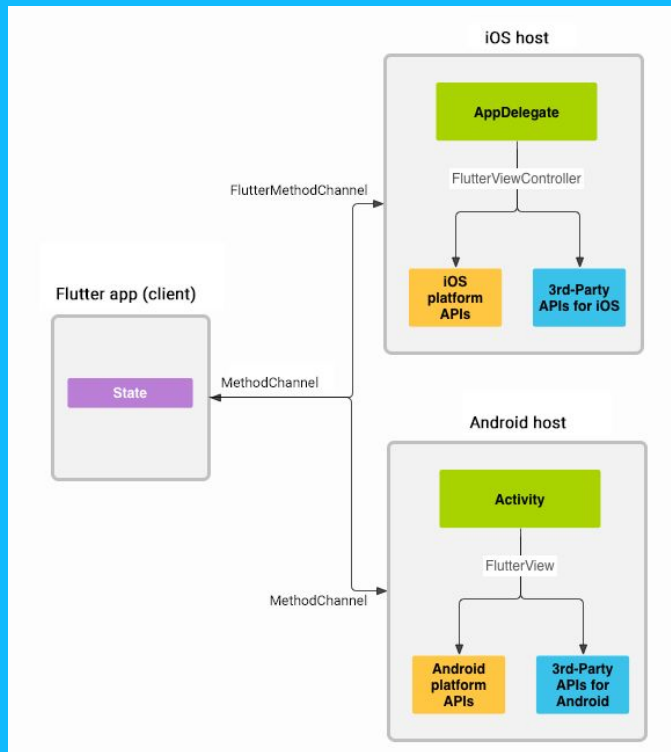
☒ STREAM CONTROLLER ☒ PUSH NOTIFICATION

Detalhes - 3

<< arquivo dart >>

```
Future<String> sendData() async {  
  http.Response response = await http.get(  
    Uri.encodeFull("https://ps.pndsn.com/publish/pub.../s  
ub.../0/flutter_iot_lamp/0/{\"action\":$statusEletro}  
?uuid=db9..."),  
  );  
  
  print(response.body);  
}
```


Platform Channel



Detalhes - 4

<< **MainActivity.kt** >>

```
val pnConfiguration = PNConfiguration()
pnConfiguration.setSubscribeKey("sub...")
pnConfiguration.setPublishKey("pub...")
val pubnub = PubNub(pnConfiguration)
pubnub.addListener(object : SubscribeCallback() {
    override fun message(pubnub: PubNub, message: PNMessageResult)
    {
        MethodChannel().invokeMethod();
    })

pubnub.subscribe().channels(<channels>).execute()
```

Save my life

- Android Studio - flutter_sensors_control_eletro
- Visual Studio Code:
/Users/ricardoogliari/Documents/Estudos/FlutterComIoT



Flutter
Talks 2019

Twilio

Twilio

Plataforma para envio de sms, email, comunicação com whatsapp app, dentre outras coisas.



Solução



Detalhes - 1

<< *Lado do IoT* >>

Arduino Mega

Sensor PushButton

Johnny Five

Firmata

Firmata 2 DART?

Temboo com Choreo Twilio

Detalhes - 1.1

<< *Firmata e DART* >>

<https://pub.dev/packages/firmata>

Funciona só para DART 1

Resolving dependencies...

The current Dart SDK version is 2.3.1.

Because teste_firmata depends on firmata $\geq 0.1.0$ which requires SDK version $< 2.0.0$, **version solving failed.**

Detalhes - 2

<< *MainActivity.kt* >>

```
class MainActivity: FlutterActivity(){
    val CHANNEL = "flutter.native/sms_receiver"

    override fun onCreate(savedInstanceState: Bundle?) {
        registerReceiver(receiver,
IntentFilter("android.provider.Telephony.SMS_RECEIVED"))
    }

    class IncomingSmsBroadcastReceiver : BroadcastReceiver() {
        var listener: SmsListener? = null

        override fun onReceive(context: Context?, intent: Intent?) {
            var smsBody = smsMessage?.messageBody?.toString()
            var originalAddress = smsMessage?.originatingAddress?.toString()
        }
    }
}
```

Detalhes - 2

```
class MainActivity: FlutterActivity(), SmsListener {  
    override fun onCreate(savedInstanceState: Bundle?) {  
        receiver.listener = this  
    }  
  
    override fun receive(msg: String, addr: String){  
        MethodChannel(flutterView, "flutter.native/sms_receiver").invokeMethod(  
            "messageReceived", "{\"address\": \"$addr\", \"body\": \"$msg\"}"  
        )  
    }  
  
class IncomingSmsBroadcastReceiver : BroadcastReceiver() {  
    override fun onReceive(context: Context?, intent: Intent?) {  
        listener?.receive(smsBody!!, originalAddress!!)  
    }  
}  
}  
  
interface SmsListener {  
    fun receive(message: String, address: String)  
}
```

Save me life

Android Studio: flutter_receiver_twilio

/Users/ricardoogliari/Documents/Palestras/FlutterIoT/send.js



Flutter
Talks 2019

Perguntas e Respostas



Flutter
Talks 2019

Obrigado!

`rogliariping@gmail.com`