

# ANDROID MARSHMALLOW

O QUE MUDA PARA O DESENVOLVEDOR

Ubiratan Soares  
Agosto de 2015

# Next Level Apps

do por

Google  
**Developers**



@ubiratanfsoares



+UbiratanSoares



ubiratansoares

[ubiratansoares.github.io](https://ubiratansoares.github.io)



Intel®  
Software  
Innovator



Google Developers  
**Experts**



# RUNTIME PERMISSIONS

# RUNTIME PERMISSIONS

Modelo adotado partir do SDK 23

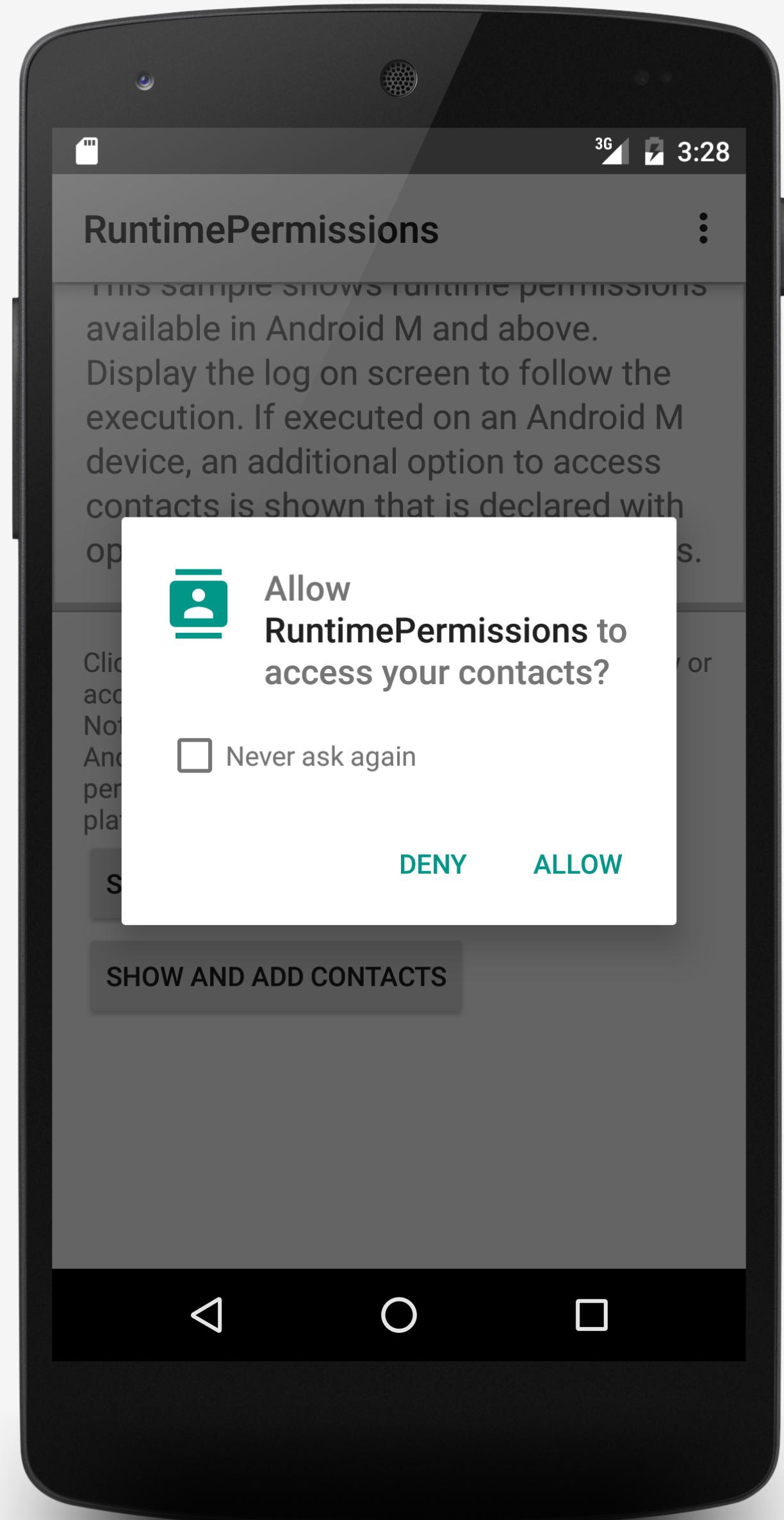
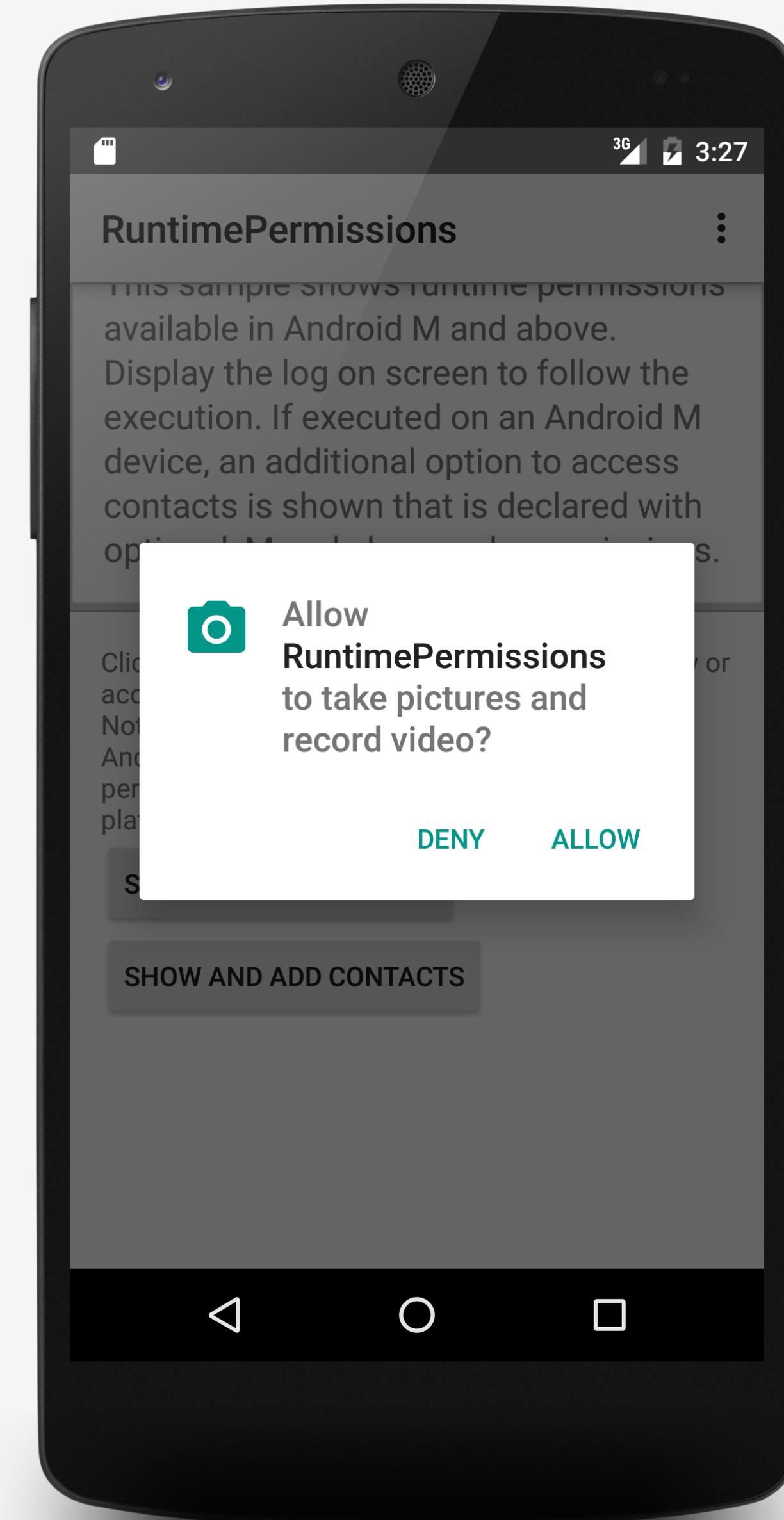
Objetivo principal : maior controle do usuário sobre a aplicação

Efeito colateral : menor fricção para atualização da aplicação

Diversas permissões “contempladas” ao longo de grupos distintos

M permite que permissão seja **concedida** ou **retirada** on-the-fly pelo usuário

Fluxos e checagens adicionais para o desenvolvedor



GROUP	PERMISSIONS
<b>CALENDAR</b>	android.permission.READ_CALENDAR android.permission.WRITE_CALENDAR
<b>CAMERA</b>	android.permission.CAMERA
<b>LOCATION</b>	android.permission.FINE_LOCATION android.permission.COARSE_LOCATION
<b>CONTACTS</b>	android.permission.READ_CONTACTS android.permission.WRITE_CONTACTS android.permission.READ_PROFILE android.permission.WRITE_PROFILE
<b>STORAGE</b>	android.permission.READ_EXTERNAL_STORAGE android.permission.WRITE_EXTERNAL_STORAGE

GROUP	PERMISSIONS
<b>MICROPHONE</b>	android.permission.RECORD_AUDIO
<b>PHONE</b>	android.permission.READ_PHONE_STATE android.permission.CALL_PHONE android.permission.READ_CALL_LOG android.permission.WRITE_CALL_LOG android.permission.ADD_VOICEMAIL android.permission.USE_SIP android.permission.PROCESS_OUTGOING_CALLS
<b>SENSORS</b>	android.permission.BODY_SENSORS android.permission.USE_FINGERPRINT
<b>SMS</b>	android.permission.SEND_SMS android.permission.RECEIVE_SMS android.permission.READ_SMS android.permission.RECEIVE_WAP_PUSH android.permission.RECEIVE_MMS android.permission.READ_CELL_BROADCASTS

## PARTE I : checar se a permissão já está concedida

```
private static final int REQUEST_CODE_PERMISSION_CONTACT = 0xCAFE;

private void checkForPermissionAndInsertContact() {

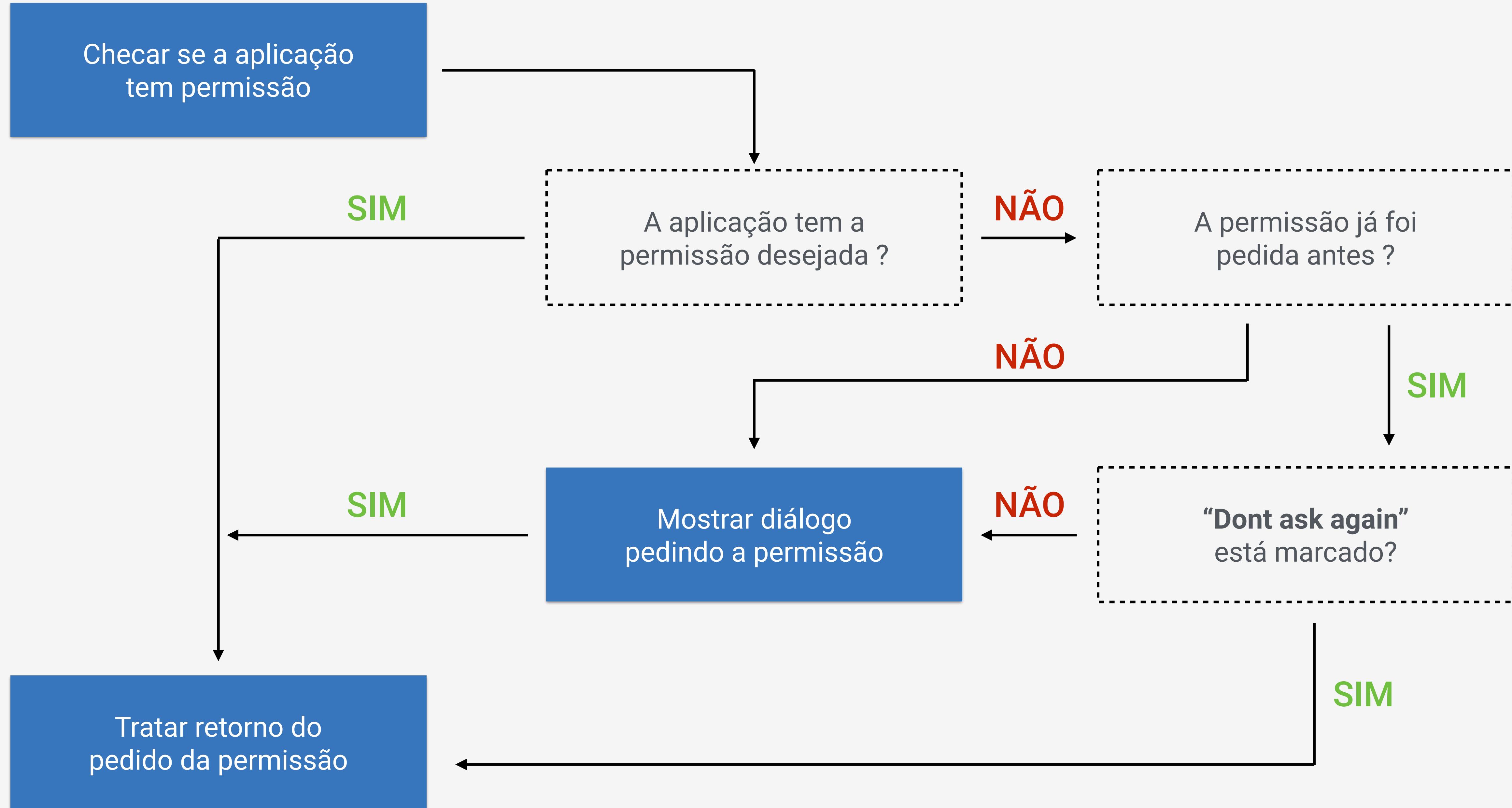
    int hasWriteContactsPermission =
        checkSelfPermission(Manifest.permission.WRITE_CONTACTS);

    if (hasWriteContactsPermission != PackageManager.PERMISSION_GRANTED) {
        requestPermissions(
            new String[]{Manifest.permission.WRITE_CONTACTS,
            REQUEST_CODE_PERMISSION_CONTACT});
        return;
    }

    // Already granted
    insertContact();
}
```

## PARTE II: tratar retorno após opção do usuário

```
@Override public void onRequestPermissionsResult( int requestCode, String[] permissions, int[] grantResults) { if (requestCode == REQUEST_CODE_PERMISSION_CONTACT) { if (grantResults[0] == PackageManager.PERMISSION_GRANTED) { // Permission Granted insertContact(); } else { // Permission Denied Toast.makeText(this, "Why user, why !!!", LENGTH_SHORT).show(); } return; } super.onRequestPermissionsResult(requestCode, permissions, grantResults); }
```



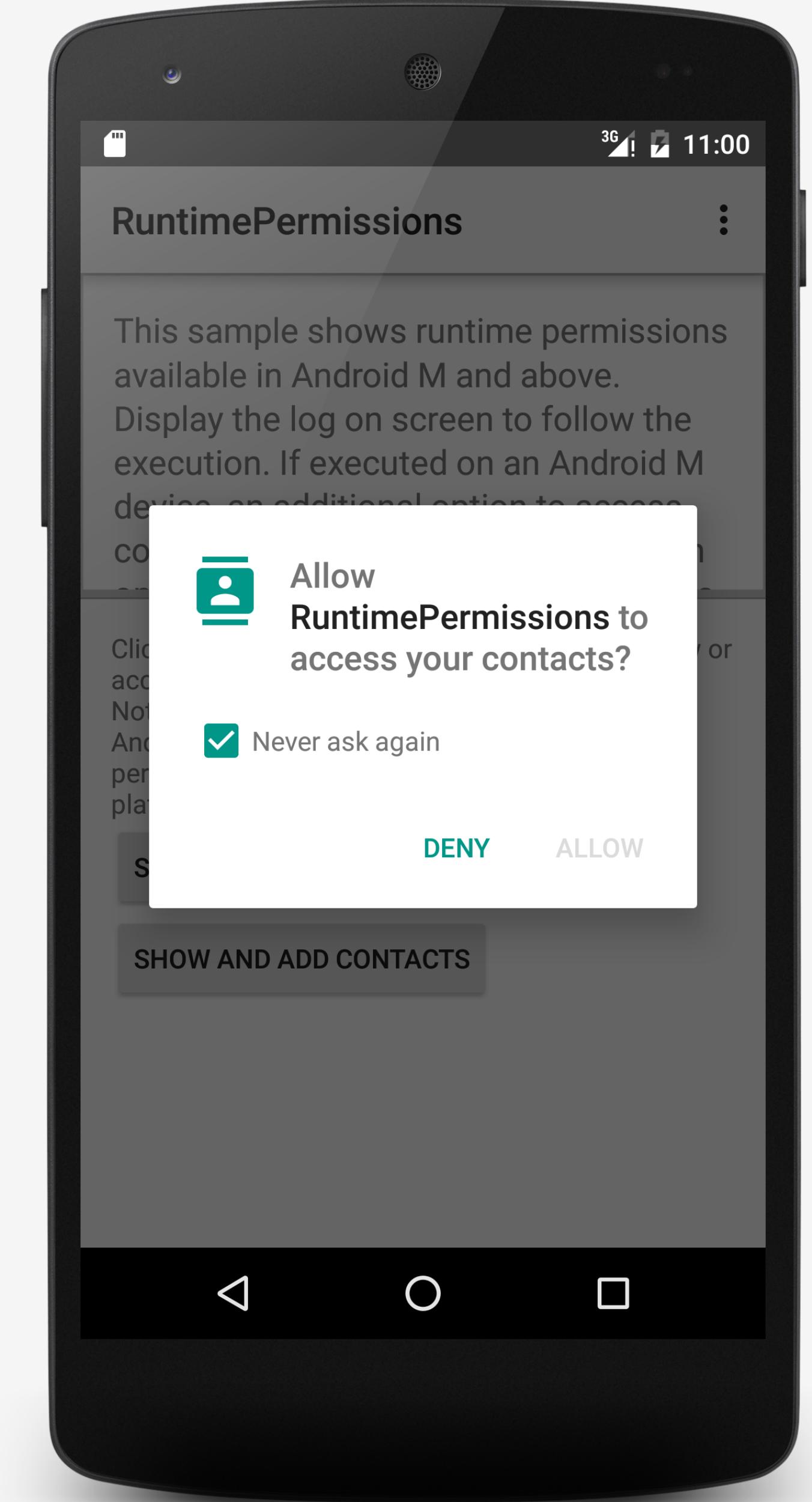
# USANDO RATIONALE

Em algum momento, o usuário pode escolher não visualizar o *prompt* da permissão novamente

Você tem a opção de verificar essa condição e exibir uma UI de interesse para o usuário, justificando sua necessidade dessa permissão (*rationale*)

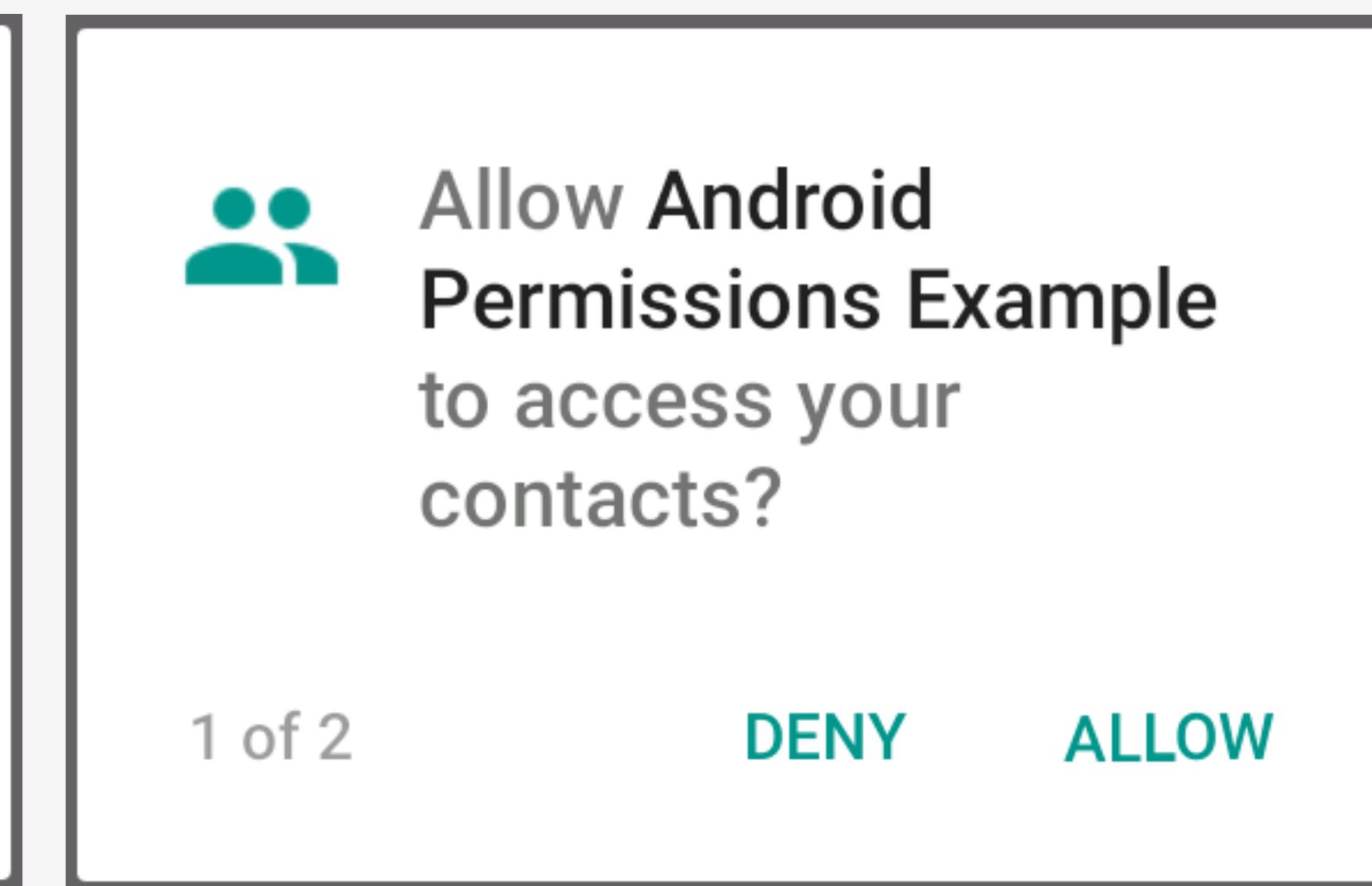
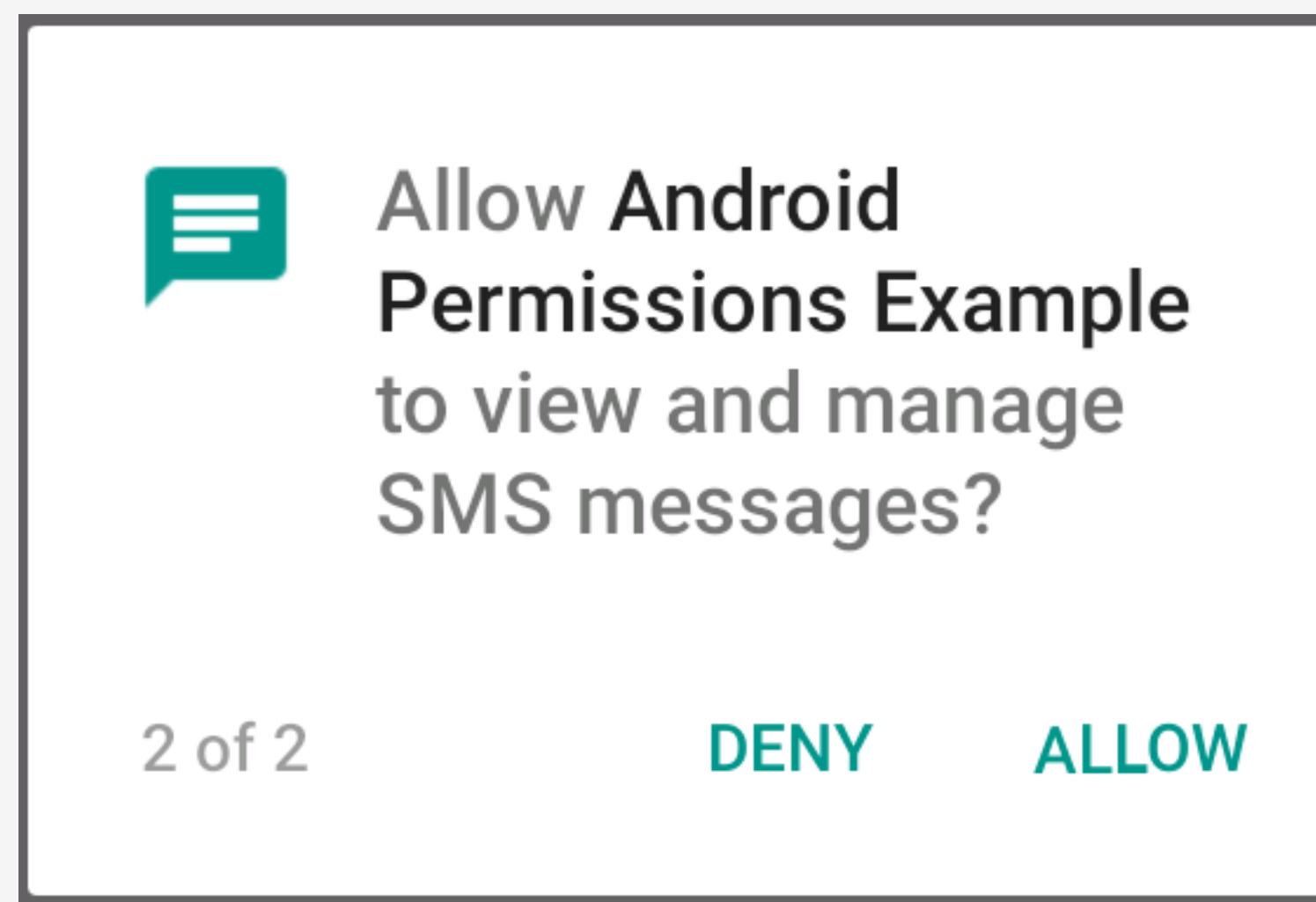
**shouldShowRequestPermissionRationale()** é o método utilitário disponível

Permissão será concedida via **Settings**



# PEDINDO MÚLTIPAS PERMISSÕES

- Configurar pelo array de permissões necessárias
- Para cada permissão, tratar o uso do *rationale* quando for o caso
- Potenciais problemas de UX



# COMPATIBILIDADE

Métodos como `checkSelfPermission()`, `requestPermission()`, `onRequestPermissionsResult()` e outros pertencem à classe `Activity` e apenas estão disponíveis para API23+



**Estratégia #01** : checar valor `Build.VERSION.SDK_INT` para aplicar comportamento (infinitos ifs ....)

**Estratégia #02** : utilizar métodos espelhados da classe `ActivityCompat`, implementando `ActivityCompat.OnRequestPermissionsResultCallback` (disponível em support-v4)

NÃO ROLA UMA  
SOLUÇÃO DA MODA ?



Else ...

```
@RuntimePermissions
public class MainActivity extends AppCompatActivity {

    @NeedsPermission(Manifest.permission.CAMERA) void showCamera() {
        // Only the show camera logic
    }

    @ShowsRationale(Manifest.permission.CAMERA) void showRationaleForCamera() {
        // Only the rationale logic
    }

    @Override public void onRequestPermissionsResult(
        int requestCode, String[] permissions, int[] grantResults) {
        // Some easy-to-trigger, boilerplate-hidden gateway
    }
}
```

# THERE IS A LIBRARY FOR THAT !

<http://hotchemi.github.io/PermissionsDispatcher>

Annotation-processing driven

Dispatcher generation e manual hook em **onRequestPermissionsResult()**

Tratamento de múltiplas permissões e múltiplos *rationales*

# THERE IS ANOTHER LIBRARY FOR THAT !

<http://www.mobmead.com/EasyMPermission/>

Project Lombok powered (Annotation Processing + Bytecode Manipulation)

Múltiplas permissões, mas sem suporte a *rationales* (por enquanto ...)

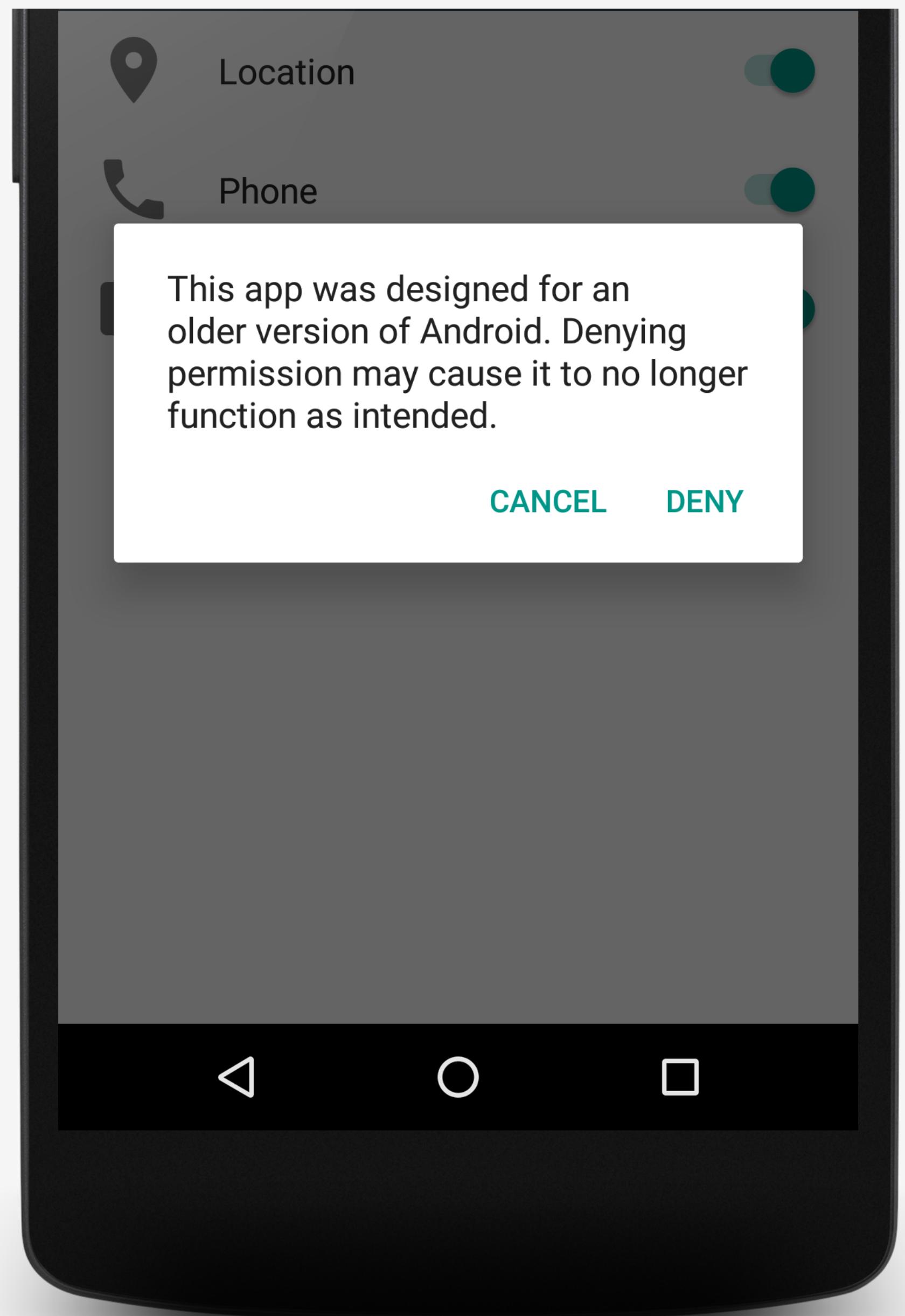
Não é preciso implementar **onRequestPermissionsResult()**

# IMPORTANTE

Legacy Mode se `targetSDK = 22`

Usuário não será interrompido por pedido de permissão !

Porém, usuários são usuários ...



# DICAS DE SOBREVIVÊNCIA

Implemente o suporte à permissões seletivas para **ontem**

Atenção para funcionalidades e telas que precisam de múltiplas permissões de uma única vez

Desenhe fluxos dentro da sua aplicação para o(s) limbo(s) de um ou mais *rationale*(s)

Prefira as bibliotecas para lidar com código burocrático

Sem desculpas e/ou preguiça para ficar no Legacy Mode

**TESTE SUA APLICAÇÃO CONTRA O M** (temos emulador para isso!)

Virtual Device Configuration

## System Image

Select a system image

Release Name	API Level ▾	ABI	Target
MNC	MNC	armeabi-v7a	Android M (Preview)
MNC	MNC	x86_64	Android M (Preview)
Marshmallow	23	x86_64	Android 6.0
Marshmallow	23	x86_64	Google APIs (Google Inc.) – google_ap
Lollipop	22	x86	Google APIs (Google Inc.) – google_ap
Lollipop	21	x86	Android 5.0.1
Lollipop	21	x86_64	Android 5.0.1
KitKat	19	x86	Google APIs (x86 System Image) (Goo
Jelly Bean	17	armeabi-v7a	Google APIs (Google Inc.)
Jelly Bean	16	armeabi-v7a	Google APIs (Google Inc.)
IceCreamSandwich	15	armeabi-v7a	Google APIs (Google Inc.)

Show downloadable system images



**Marshmallow**

API Level  
**23**

Android  
**6.0**

Google Inc.

System Image  
**x86\_64**

**Recommendation**

Install Intel HAXM for better emulation performance.

[Download and install HAXM](#)

Questions on API level?  
See the [API level distribution chart](#)

Cancel Previous **Next** Finish



# APP LINKING

# APP LINKS

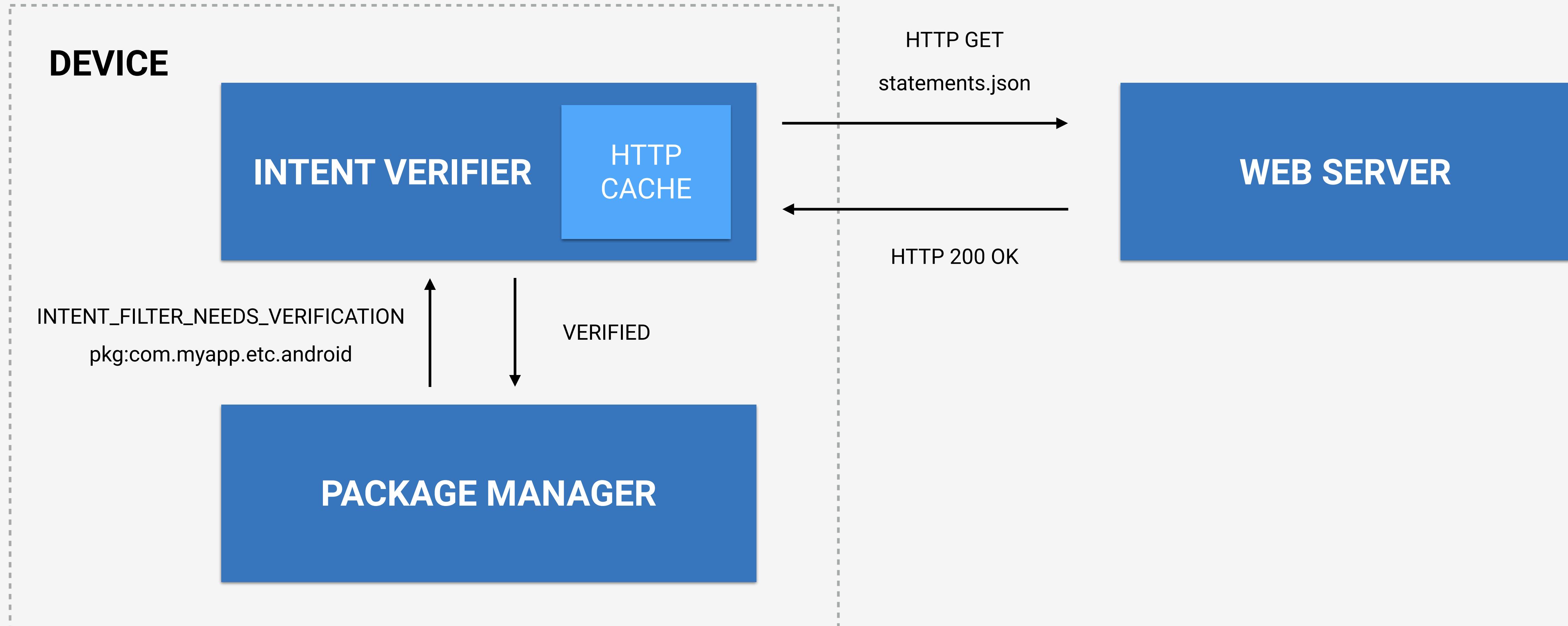
Mecanismo para facilitar integração com *deeplinks*

Menor fricção entre links da Web e experiência nativa com a remoção do  
**ChooserDialog**

*Deeplinks* ainda precisam ser tratados manualmente dentro da aplicação

Representa uma tendência para adotar o esquema HTTP como padrão na  
mecânica de *deeplinking*

# FUNCIONAMENTO



<https://seuapp.com/.well-known/statements.json>

```
[{  
  "relation": ["delegate_permission/common.handle_all_urls"],  
  
  "target": {  
    "namespace": "android_app",  
    "package_name": "com.seupacote.seuapp/etc",  
    "sha256_cert_fingerprints": ["09:87:65:34:12:..."]  
  }  
}]
```



```
> keytool -list -v -keystore release.keystore
```

## AndroidManifest.xml

```
<activity...>

<intent-filter android:autoVerify="true">
    <action android:name="android.intent.action.VIEW" />
    <category android:name="android.intent.category.DEFAULT" />
    <category android:name="android.intent.category.BROWSABLE" />

    <data android:scheme="http" android:host="seuapp.com" />
    <data android:scheme="http" android:host="www.seuapp.com" />
</intent-filter>

</activity>
```



onde **.well-known/statements.json** será validado

# MAIS DETALHES

Verificação falha com *timeout* de 5s contra o web server ou por falha de conectividade

Resultado respeita mecanismo de HTTP Caching (*max-age*, *etags*, etc), apenas para status 200

JSON de validação deve ser acessível de ambos os *hostnames*

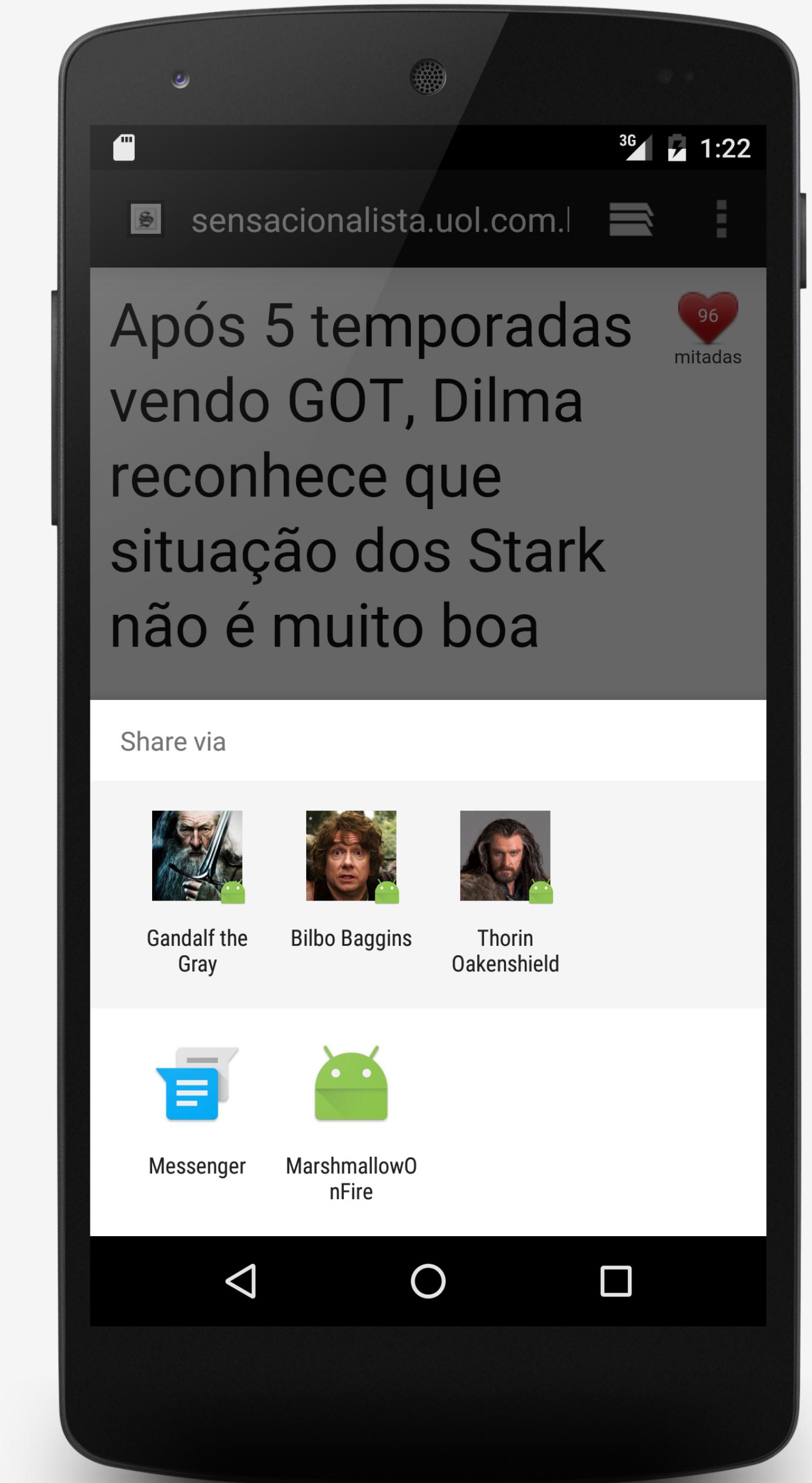
Novas verificações a princípio serão realizadas apenas mediante *updates* ou re-instalações da aplicação



DIRECT  
SHARE

# DIRECT SHARE

- Mecanismo para melhor experiência de compartilhamento
- Aplicações podem oferecer um canal direto - com informações agregadas - para o qual o conteúdo de interesse será compartilhado



## AndroidManifest.xml

```
<activity android:name=".ShareActivity">

    <intent-filter>
        <action android:name="android.intent.action.SEND"/>
        <category android:name="android.intent.category.DEFAULT"/>
        <data android:mimeType="text/plain"/>
    </intent-filter>

    <meta-data
        android:name="android.service.chooser.chooser_target_service"
        android:value=".DirectSharingService"/>
</activity>

<service
    android:name=".DirectSharingService"
    android:label="@string/app_name"
    android:permission="android.permission.BIND_CHOOSER_TARGET_SERVICE">

    <intent-filter>
        <action android:name="android.service.chooser.ChooserTargetService"/>
    </intent-filter>
</service>
```

## DirectShareService.java

```
public class DirectSharingService extends ChooserTargetService {  
  
    private List<FakeContact> contacts = getProvidedContacts();  
  
    @Override public List<ChooserTarget> onGetChooserTargets(  
        ComponentName targetActivityName, IntentFilter matchedFilter) {  
  
        final List<ChooserTarget> targets = new ArrayList<>();  
        for (FakeContact contact : contacts) {  
  
            final String title = contact.getName();  
            final Icon icon = Icon.createWithResource(this, contact.getDrawableResId());  
            final float score = 1.0f;  
            final Bundle extras = new Bundle();  
            extras.putSerializable("contact", contact);  
            ComponentName name = createRelative(this, ShareActivity.class.getSimpleName());  
            targets.add(new ChooserTarget(title, icon, score, name, extras));  
        }  
  
        return targets;  
    }  
}
```



# AUTO-BACKUP FOR APPS

# APP DATA AUTO-BACKUP

A partir do M, Android faz *auto-backup* dos dados internos da aplicação de forma automática (exceto diretório de cache e storage externo)

É possível filtrar conteúdo a partir de esquema de *backup*

Potenciais problemas com restauração de banco de dados e /ou `sharedPreferences`

Potenciais problemas com dados que não possam ser *backupeados* ...

## AndroidManifest.xml

```
<manifest xmlns:android="http://schemas.android.com/apk/res/android"  
    package="com.seupacote.seuapp" >  
  
    <application  
        android:allowBackup="true"  
        android:icon="@mipmap/ic_launcher"  
        android:label="@string/app_name"  
        android:theme="@style/AppTheme"  
  
        android:fullBackupContent="@xml/backup">
```

## res/xml/backup.xml

```
<full-backup-content >
```

```
    <exclude domain="sharedpref" path="gcm"/>
    <exclude domain="sharedpref" path="user_credentials"/>
    <exclude domain="database" path="local_secrets.db"/>
```

```
<!-- Additional domains include "file", "external", "root", and "path". See
     http://developer.android.com/preview/backup/index.html for more details.
```

```
Additionally, content in the cache directory, external storage, and the no_backup directory
(see android.content.Context#getNoBackupFilesDir()) are excluded by default. If you need
to backup data in one of these locations, use the <include> directive. -->
```

```
</full-backup-content>
```



**POWER  
SAVING**

# DOZE

Android M trás nova política super-agressiva para economia de energia

Aparelho entra em **Doze Mode** quando inativo:

- Sem carregar
- Sem movimento percebido
- 1+ hora com a tela desligada e sem interrupções de rede (GCM)

Quando em **Doze Mode**, praticamente todo o processamento em segundo plano não irá acontecer

Aparelho abre janelas para processamento periodicamente (*Doze Windows*)

# DEVICE IDLE CONTROLLER

- Doze é gerenciado por um novo serviço de sistema (`DeviceIdleController`)
- FSM que varia acordo com sinais de atividade do aparelho
- Inspeciona um documento de *whitelist*, o qual determina quais aplicações estão habilitadas a eventualmente operar em Doze Mode (Google Play Services e outras de sistema)
- Aplicações que não constam na *whitelist* terão total bloqueio fora da janela de processamento, mesmo acessando serviços de sistema tradicionais

**`NetworkingPoliceManagerService`, `JobSchedulerService`,  
`SyncManager`, `PowerManagerService`, `AlarmManagerService`**

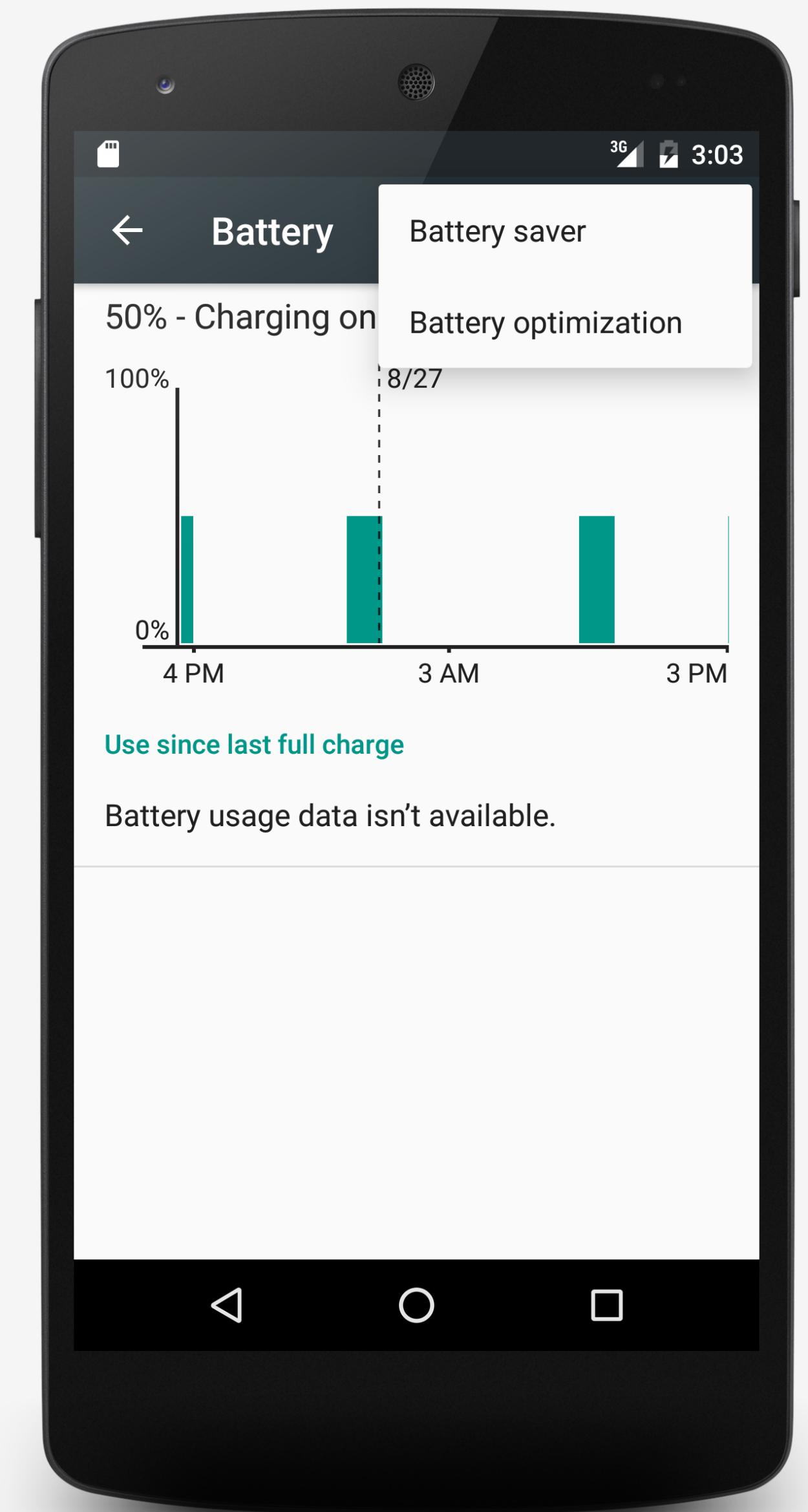
# CONTORNANDO DOZE

Única forma segura de interromper o Doze Mode é um gatilho via GCM

Google pretente prover um qualificador de prioridade para mensagens de *push*, no sentido de identificar as pertinentes para essa condição

Aplicações de sistema podem alterar a *whitelist*

Usuário pode alterar a *whitelist* nas configurações de energia





# SECURITY IMPROVEMENTS

# APACHE HTTP PURGING

Apache HTTP finalmente expurgado da API

Atenção para bibliotecas legadas que utilizam essa API para acesso HTTP diretamente (inspecione seu framework favorito !!!)

Em último caso

```
android {  
    compileSdkVersion 23  
    buildToolsVersion "23.0.0"  
  
    useLibrary 'org.apache.http.legacy'
```

# HARDWARE ID ACCESS

A partir do M, não será possível acessar mac address do aparelho via APIs de Wifi ou Bluetooth (valor *default* em ambos os casos)

Para acessar identificadores de possíveis peers, permissões serão necessárias

Identificadores de Interesse	Permissão Necessária
<code>WifiManager.getScanResults()</code>	<code>android.permission.FINE_LOCATION</code> <code>android.permission.COARSE_LOCATION</code>
<code>BluetoothLeScanner.startScan()</code>	<code>android.permission.FINE_LOCATION</code> <code>android.permission.COARSE_LOCATION</code>



MUITO  
MAIS !!!

# MUITO MAIS !!!

Fingerprint API, Android Pay

Voice Interaction API

Novas APIs para exibição de texto (incluindo em círculos!)

Novas APIs para câmera

Novas APIs de áudio (MIDI e áudio de alta qualidade)

ETC

# REFERÊNCIAS

- Random Musings About M - <http://bit.ly/1KnHIGO>
- Exploring the New Android Permissions Model - <http://bit.ly/1i50vpQ>
- Diving into the Android “M” Doze - <http://bit.ly/1En3hAV>
- Android “M” App Links - <http://bit.ly/1Iip3Qz>
- Official Preview Docs - <https://developer.android.com/preview>
- Github Samples - <https://github.com/googlesamples>

# OBRIGADO

<http://bit.ly/feedback-bira-android-m>