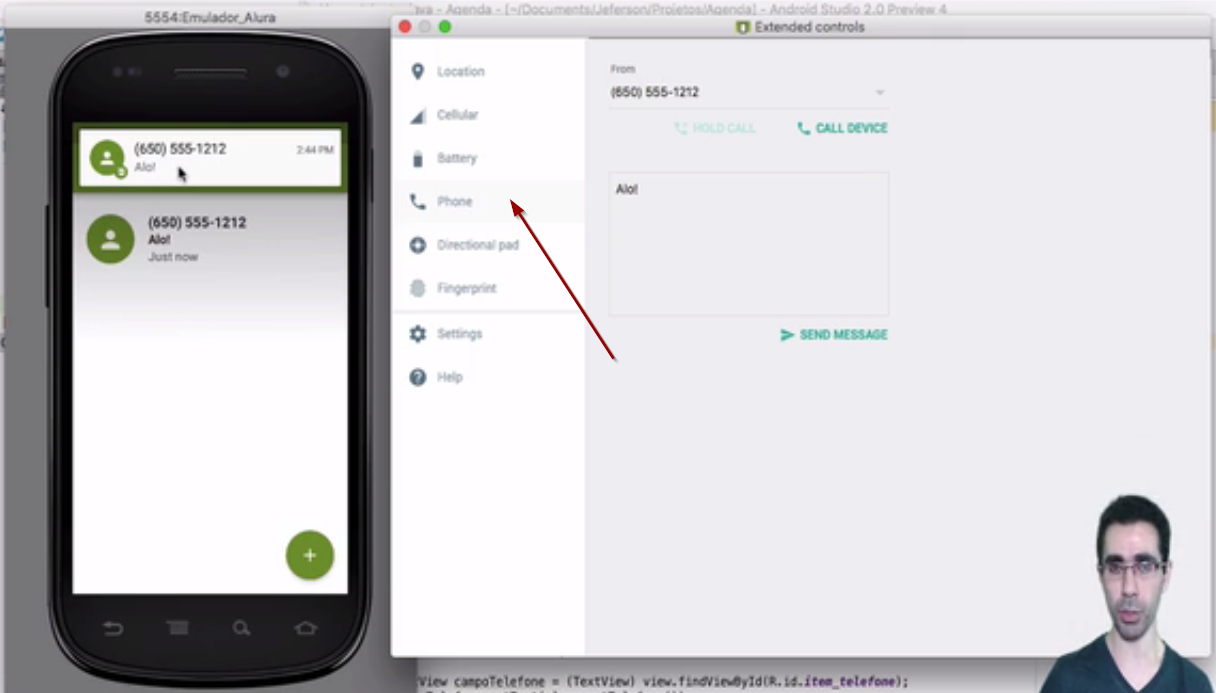
Tratando eventos de sistema

O que fizemos por último foi melhorar o visual da nossa agenda e colocamos as fotos dos alunos na lista também.

Agora, queremos adicionar um diferencial para as mensagens de SMS que chegassem dos contatos da nossa agenda de aluno. Antes de adicionarmos estas melhorias, vamos ver o que acontece quando um Android recebe um SMS.

As mensagens sempre chegarão em um aplicativo específico para SMS. No emulador, conseguimos simular o recebimento de um SMS nos "Extended controls" ("controles extendidos"), clicando nas funções de Telefone.



Vamos simular o envio de um SMS. Ao recebermos a mensagem, visualizamos uma notificação na tela, além de um sinal sonoro próprio do Android. Também receberemos a notificação, mesmo que outro aplicativo esteja funcionando. E se abrirmos o aplicativo de SMS, veremos que todas as mensagens recebidas foram guardadas.

Quando o Android recebe um SMS, ele irá querer verificar a quais aplicativos interessa esta mensagem. Assim que ele identifica, é feito uma cópia do SMS para os aplicativos que irão tratá-lo. Isto significa que o Android "baterá na porta" de todos os apps - esta operação nós chamamos de *broadcast*. Porém, os aplicativos terão que manifestar esse interesse, é o que a nossa agenda precisará fazer também.

Vamos criar uma classe que tratará o recebimento deste SMS e incluir, por enquanto, um Toast. Vamos criar um novo pacote chamado receiver e dentro dele, vamos adicionar a classe SMSReceiver e vamos estender uma classe do Android que já existe BroadcastReceiver. Quando chegar o evento, o Android irá chamar o método onReceive.

public class SMSReceiver extends BroadcastReceiver {  
 @Override  
 public void onReceive(Context context, Intent intent) {  
 Toast.makeText(context, "Chegou um SMS!", Toast.LENGTH\_SHORT).show();  
 }  
}

Nós criamos uma classe, que o Android não sabe ainda que existe.

No Toast, vamos uma usar o context e a nossa aplicação irá exibir a mensagem Chegou um SMS!. Usamos o LENGTH\_SHORT, porque a duração será curto, e invocamos o método show, para o nosso Toast não ficar guardado.

Em seguida, iremos no arquivo manifest e dentro dela, encontraremos a tag application.

<application  
 android:allowBackup="true"  
 android:icon="@mipmap/ic\_launcher"  
 android:label="Agenda"  
 android:theme="@style/AppTheme" >  
 <activity  
 android:name".ListaAlunosActivity"  
 android:label="Agenda" >  
 <intent-filter>  
 <action android:name="android.intent.action.MAIN" />  
  
 <category android:name="android.intent.category.LAUNCHER" />  
 </intent=filter>  
 </activity>  
 <activity  
 android:name".FormularioActivity"  
 android:label="Formulário" >  
  
 </activity>  
 </application>

Agora, queremos cadastrar no manifest, um *BroadcastReceiver*. Para isto, iremos declarar o nosso receiver.

</activity>  
  
 <receiver android:name=".receiver.SMSReceiver">  
  
 </receiver>  
 </application>

Porém, apenas o nome da classe não é o suficiente. Poderíamos ter dado qualquer nome para SMSReceiver. Teremos que declarar no manifest, qual evento está relacionado com o receiver. Iremos inserir um intent-filter, que irá dizer qual evento irá disparar a execução do receiver.

No action, iremos usar uma constante que ele não completa automaticamente:

<receiver android:name=".receiver.SMSReceiver">  
 <intent-filter>  
 <action android:name="android.provider.Telephony.SMS\_RECEIVED"/>  
 </intent-filter>  
 </receiver>

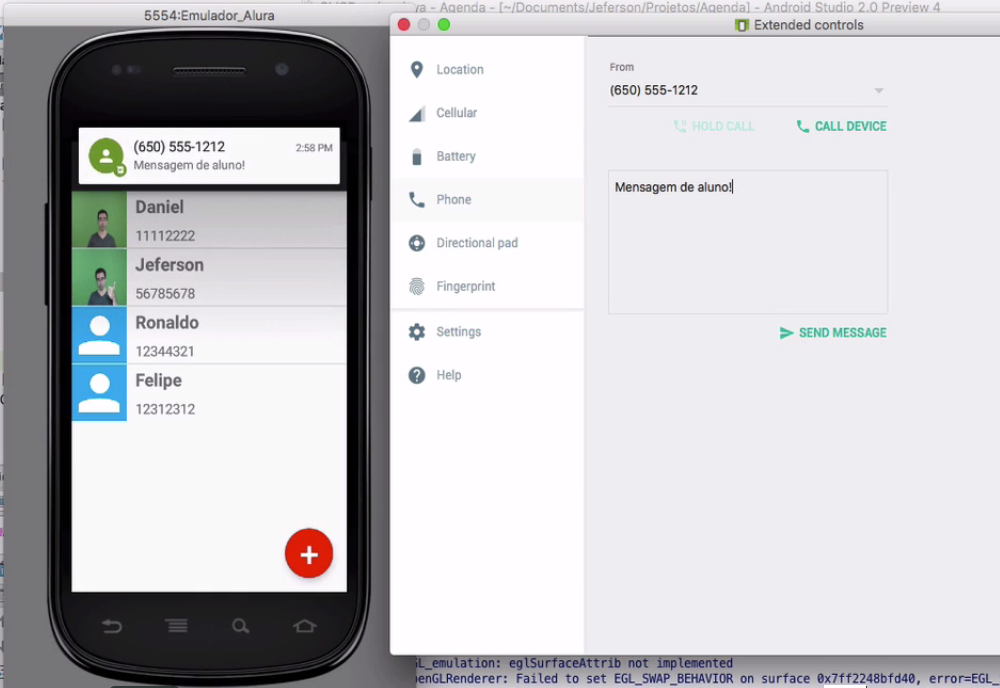
Por questões de segurança, o Android pede que toda aplicação que roda com SMS trabalhe com uma permissão especifica. Iremos trabalhar com a tag uses-permission, posicionada na parte de cima do manifest.

<?xml version="1.0" encoding="utf-8"?>  
<manifest xmlns:android="http://schenas.android.com/apk/res/android"  
 package:"br.com.alura,agenda" >  
  
 <uses-permission android:name="android.permission.CALL\_PHONE"/>

Vamos adicionar uma permissão para receber SMS.

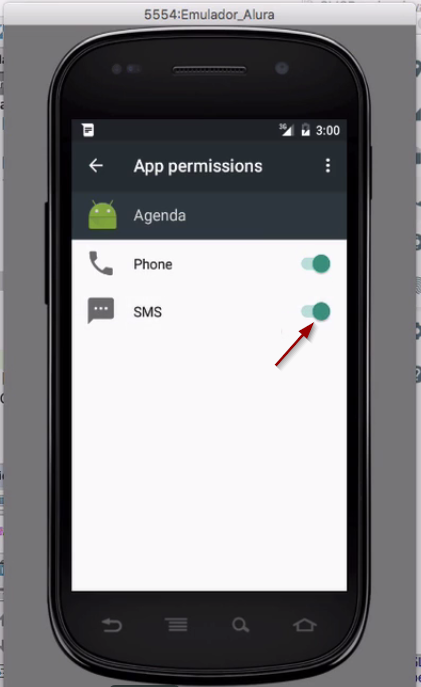
<uses-permission android:name="android.permission.RECEIVE\_SMS"/>

Ao rodarmos o aplicativo no emulador, receberemos uma mensagem do sistema e o sinal sonoro, mas o Toast ainda não será visualizado.

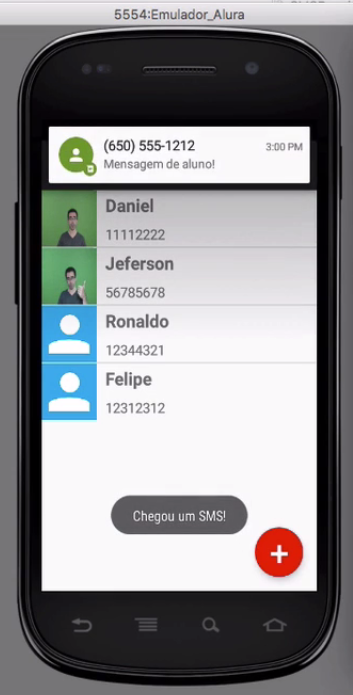


Isto aconteceu, porque ainda não tratamos um detalhe no código. Falamos anteriormente, que a partir do Android 6.0, precisamos pedir permissão para o usuário. Mas seria estranho ter que pedi-la sempre que chegasse uma mensagem. O ideal seria que ela fosse pedida apenas uma única vez.

Para facilitar isso, daremos esta autorização nas configurações do aplicativo no celular, mas poderiamos fazer o que foi feito nas outras permissões, usando o OnRequestPermissionResult().



Após fazermos isto, agora o Toast será visualizado.



Se estivéssemos em outro aplicativo ativo, também receberíamos a mensagem o aviso "Chegou um SMS!".

Mais adiante, veremos como adicionar um aviso sonoro diferente e como verificar se o SMS é de um contato cadastrado na nossa agenda.