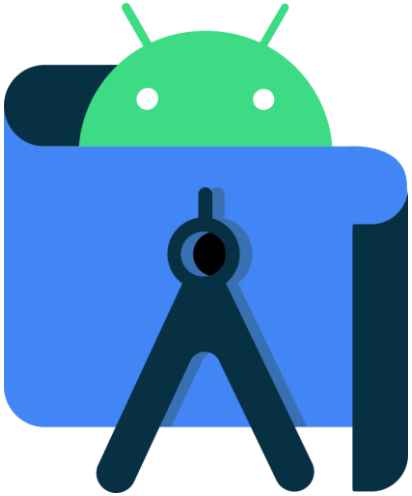
Universidade Federal do Amazonas
Instituto de Computação
IArTES - Especialização em IA para Engenharia de Testes de Software



Notificações

Prof. Diogo Soares

diogosoaresm@ufam.edu.br

Github: github.com/diogosm/android_class_IArTES

Notificações

- Um recurso bastante interessante do Android são as notificações
  - Além de gerar alguma comunicação pro usuário de forma interativa, pode ser usado para agendar notificações como beber água em tais horários, um compromisso ou um aviso do seu app

- Android já vem com recursos para notificação e agendamentos

Como criar?

- 1. Vamos criar um novo projeto
  - Vamos usar a activity principal com apenas um botão

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">

    <Button
        android:id="@+id/notificacao"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Ativa Notificacao"
        app:layout_constraintBottom_toTopOf="@+id/textView"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

</androidx.constraintlayout.widget.ConstraintLayout>
```

Como criar?

- 2. Vamos adicionar uma função para verificar permissões
  - Chamando uma função para verificar se o app tem a permissão de gerar notificações

```
if(ContextCompat.checkSelfPermission(MainActivity.this,
        Manifest.permission.POST_NOTIFICATIONS) != PackageManager.PERMISSION_GRANTED){
    ActivityCompat.requestPermissions(MainActivity.this,
            new String[]{Manifest.permission.POST_NOTIFICATIONS}, 101);
}else{
    Log.i("debug", "Permissao garantida já!");
}
```

  - Na função acima verificamos se a permissão foi setada para o aplicativo, mas além disso precisamos pedir do usuário ao instalar o app

Como criar?

- 2. Vamos adicionar uma função para verificar permissões
  - Isso é feito através do Manifesto (*AndroidManifest.xml* na pasta manifests)
  - Adicione as linhas abaixo no manifesto

```
<uses-permission android:name="android.permission.POST_NOTIFICATIONS" />
```

  - Além disso, adicione um novo parâmetro dentro da tag application para identificar uma subclasse que irá atuar como um receiver broadcast, isto é, um componente que irá permitir que seu app responda a mensagens que são repassadas pelo sistema android ou outras aplicações (Como acontece nos apps que usamos no dia-a-dia)

```
<receiver android:name=".NotificationPub"></receiver>
```

  - Perceba que criaremos a classe NotificationPub futuramente!

Como criar?

- 3. Vamos adicionar uma função de clicar ao botão
  - Quando o botão for clicado, irá chamar uma função que irá agendar a notificação

```
button = (Button) findViewById(R.id.notificacao);

button.setOnClickListener(new View.OnClickListener() {
   @Override
   public void onClick(View view) {
      Toast toast = Toast.makeText(view.getContext(),
            "Criando uma notificacao em 10 segundos..",
            Toast.LENGTH_SHORT);
      toast.show();
      //faz a notificacao
      agendaNotificacao2(10000);
   }
});
```

Como criar?

- 4. Vamos criar a função de agendar a notificação
  - Para isso, vamos criar um intent que leve para a subclasse que irá fazer o build (na prática desenhar) da notificação para o usuário, usando a subclasse definida no manifest
  - Adicione o intent e uma flag de clear top
  - Após isso, crie uma PendingIntent, que é um tipo de intent para ser utilizado por uma entidade externa, como outra aplicação do sistema.
  - Após isso, crie o alerta, usando o alarmManager

```
private void agendaNotificacao2(int delay){
    Intent intent = new Intent(getApplicationContext(),
        NotificationPub.class);
    intent.addFlags(Intent.FLAG_ACTIVITY_CLEAR_TOP);

    Log.i("debug", "agendando notificacao");
    PendingIntent pendingIntent2 =
        PendingIntent.getBroadcast(getApplicationContext(),
            0,
            intent,
            PendingIntent.FLAG_IMMUTABLE);

    AlarmManager alarmManager = (AlarmManager)
                getSystemService(Context.ALARM_SERVICE);
    alarmManager.setAndAllowWhileIdle(
        AlarmManager.RTC_WAKEUP,
        SystemClock.elapsedRealtime() + 1000,
        pendingIntent2
    );
}
```

Como criar?

- 4. Vamos criar a função de agendar a notificação
  - Percebam que utilizamos a função setAndAllowIdle() para criar uma notificação em plano de fundo mesmo que o device esteja idle (Lembre-se: existem várias formas de usar o gerenciador de alarmes
  - Além disso, pedimos para ele executar 1 segundo depois
  - Isto é, depois de 1 segundo executa o pendingIntent2 (o Android, não sua app :p)
- Referência: developer.android.com/reference/android/app/AlarmManager

```
private void agendaNotificacao2(int delay){
    Intent intent = new Intent(getApplicationContext(),
        NotificationPub.class);
    intent.addFlags(Intent.FLAG_ACTIVITY_CLEAR_TOP);

    Log.i("debug", "agendando notificacao");
    PendingIntent pendingIntent2 =
        PendingIntent.getBroadcast(getApplicationContext(),
            0,
            intent,
            PendingIntent.FLAG_IMMUTABLE);

    AlarmManager alarmManager = (AlarmManager)
            getSystemService(Context.ALARM_SERVICE);
    alarmManager.setAndAllowWhileIdle(
        AlarmManager.RTC_WAKEUP,
        SystemClock.elapsedRealtime() + 1000,
        pendingIntent2
    );
}
```

Como criar?

- 5. Crie a subclasse NotificationPub.java
  - Atenção! Esta classe não precisa, mas pode ter layout. No nosso exemplo, usaremos sem layout. Lembre-se que, quando clicado, é esperado que sua app faça algo, como abrir algum intent, por exemplo
  - Ela vai ser uma extensão da BroadcastRecei ver (afinal, você floodou uma notificação para o sistema :p)

Como criar?

- 5. Crie a subclasse NotificationPub.java
  - Pontos chaves nesse momento:
    - *A função chamada será a onReceive(), e é ela que iremos implementar!*
    - *Crie um notificationChannel. É uma espécie de template para as notificações, no qual canais podem ter configurações próprias como suporte a vibração, lanterna, etc*
    - *Crie o notificationManager, que é o cara que é responsável por mostrar a notificação construído no passo abaixo*

```
NotificationManager nfm = (NotificationManager)
    context.getSystemService(
        Context.NOTIFICATION_SERVICE
    );
```

    - *Crie o builder,. O builder é um subcomponente da classe notificationCompat que seta um contexto a um canal e é responsável pela atribuição de propriedades da notificação como texto, cores e ícones*

Subclasse NotificationPub

- Código-fonte da activity:

```
public class NotificationPub extends BroadcastReceiver {
   @Override
   public void onReceive(Context context, Intent intent) {
      Log.i("debug","Trying rcv notif");
      NotificationChannel channel = null;
      if (android.os.Build.VERSION.SDK_INT >= android.os.Build.VERSION_CODES.O) {
         channel = new NotificationChannel(
               "br.edu.ufam.testenotification",
               "testeNotification",
               NotificationManager.IMPORTANCE_DEFAULT
         );
      }
      NotificationManager nfm = (NotificationManager)
            context.getSystemService(
                  Context.NOTIFICATION_SERVICE
            );
      if(nfm != null){
         if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.O) {
            nfm.createNotificationChannel(channel);
            Log.i("debug","criando canal de notif");
         }
      }
      Log.i("debug","notificando...");
      String channelID = "CHANNEL_ID_NOTIFICATION";
      NotificationCompat.Builder builder = new NotificationCompa
            context, channelID
      );

      builder.setSmallIcon(R.drawable.baseline_info_24)
            .setContentTitle("Titulo")
            .setContentText("Notificacao tops")
            .setAutoCancel(true)
            .setPriority(NotificationCompat.PRIORITY_DEFAULT);
      nfm.notify(1, builder.build());
   }
}
```

Resultado

- Notificação ao abrir
- Notificação após 1s
  - Lembre-se que com o AlarmManager você pode agendar!