


Android

Serviço de Notificação, Toasts e Alarmes




Softblue
cursos online

Tópicos Abordados



- Serviço de Notificação
 - O que é
 - Criando notificações
 - Cancelando notificações
 - Notificações estendidas
- Toasts
 - Criando toasts para notificar usuários
- Alarmes
 - Quando usar
 - Tipos de alarmes
 - Como agendá-los

Serviço de Notificação



- Quando um componente executando em segundo plano deseja notificar o usuário sobre algo, é preferível utilizar o serviço de notificação
- Serviços como o recebimento de mensagens de texto utilizam este recurso

Exemplo de Notificação de SMS

Criando uma Notificação

- Obtendo o **NotificationManager**

```
NotificationManager nm =
(NotificationManager) getSystemService(NOTIFICATION_SERVICE);
```
- Criar o objeto **Notification.Builder**

```
Notification.Builder builder = new
Notification.Builder(this, channelId);
builder.setContentTitle("Título da Mensagem");
builder.setContentText("Texto da Mensagem");
builder.setSmallIcon(R.drawable.ic_action_mail);
```
- Criar um objeto **PendingIntent**

```
Intent i = new Intent(this, TriggerActivity.class);
PendingIntent pi = PendingIntent.getActivity(this, 0, i, 0);
builder.setContentIntent(pi);
```

Criando uma Notificação

- Criar o objeto **Notification**

```
Notification notification = builder.build();
```
- Enviar a notificação

```
nm.notify(R.string.notification_id, notification);
```

Notificação em Versões Anteriores



- A classe *Notification.Builder* surgiu no Android 3.0 (API Level 11)
- Para suportar versões anteriores do Android, é preciso usar a classe **NotificationCompat.Builder** da API de compatibilidade

```
NotificationCompat.Builder builder = new  
NotificationCompat.Builder(this);
```

Cancelando Notificações



- Existem duas formas de cancelar notificações
 - Chamando *setAutoCancel()* no *Builder*

```
builder.setAutoCancel(true);
```

- Cancelando via código

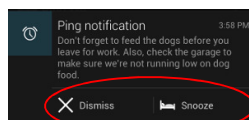
```
NotificationManager nm =  
(NotificationManager) getSystemService(NOTIFICATION_SERVICE);  
nm.cancel(R.string.notification_id);
```

- O método *cancelAll()* também pode ser usado
 - Cancela todas as notificações disparadas

Notificações Expandidas



- Notificações podem ser maiores e conter botões de ação



Notificações Expandidas



- Exemplo de como criar uma notificação expandida

```
Intent intent1 = new Intent(this, MyActivity1.class);
Intent intent2 = new Intent(this, MyActivity2.class);
PendingIntent pi1 = PendingIntent.getActivity(this, 0, intent1, 0);
PendingIntent pi2 = PendingIntent.getActivity(this, 0, intent2, 0);

Notification.Builder builder = new Notification.Builder(this, channelId)
    .setSmallIcon(android.R.drawable.sym_action_chat);
    .addAction(R.drawable.img1, "Ação 1", pi1);
    .addAction(R.drawable.img2, "Ação 2", pi2);
    .setStyle(new Notification.BigTextStyle()
        .bigText("Texto")
        .setBigContentTitle("Titulo"));

Notification n = builder.build();
```

Notificações com Toast



- Um toast permite mostrar uma mensagem curta e rápida para o usuário
- Enquanto o toast é mostrado, o usuário pode continuar a usar o dispositivo normalmente



Exibição do toast

Criando Toasts



```
Toast t = Toast.makeText(this, R.string.txt, Toast.LENGTH_SHORT);
t.show();
```

Context

Texto

Duração

```
t.setGravity(Gravity.TOP, 0, 200);
```

Define a posição do toast

```
t.setView(view);
```

Define uma view para o toast

Alarmes



- Alarmes podem ser usados para disparar intents em um horário pré-definido e de acordo com intervalos de tempo
- São independentes de aplicações
- Ficam ativos mesmo que o dispositivo esteja em estado de espera
 - Se o dispositivo for reiniciado ou desligado, o alarme é desativado

Tipos de Alarmes



- Quando um alarme é criado, é preciso definir um tipo para ele
- Existem 4 tipos disponíveis

Tipo	Descrição
RTC_WAKEUP	Acorda o dispositivo e dispara a intent na hora definida
RTC	Mesmo que RTC_WAKEUP, mas não acorda o dispositivo
ELAPSED_REALTIME_WAKEUP	Acorda o dispositivo e dispara a intent com base no tempo decorrido desde que o dispositivo foi ligado
ELAPSED_REALTIME	Mesmo que RTC_REALTIME_WAKEUP, mas não acorda o dispositivo

A Classe *AlarmManager*



- Serviço que gerencia os alarmes

```
AlarmManager am = (AlarmManager) getSystemService(ALARM_SERVICE);
```

- Possui métodos para configurar o agendamento dos alarmes

```
set(int, long, PendingIntent)  
setExact(int, long, PendingIntent)  
setRepeating(int, long, long, PendingIntent)  
setWindow(int, long, long, PendingIntent)  
cancel(PendingIntent)
```

Exemplo de Agendamento



- Exemplo de alarme que vai disparar uma vez às 13h do dia 01/01/2050

```
AlarmManager am = (AlarmManager) getSystemService(ALARM_SERVICE);  
Intent i = new Intent(getApplicationContext(), MyReceiver.class);  
PendingIntent pi = PendingIntent.getBroadcast(this, 0, i, 0);  
  
Calendar c = Calendar.getInstance();  
c.set(2050, 0, 1, 13, 0, 0);  
  
am.set(AlarmManager.RTC_WAKEUP, c.getTimeInMillis(), pi);
```

Alarmes Repetidos



- Os alarmes podem ser programados para repetirem de acordo com um intervalo
- O método **setRepeating()** define um intervalo de repetição
- A partir da API Level 19, todos os alarmes repetidos são inexatos

Considerações sobre alarmes



- Prefira os alarmes inexatos sempre que possível
 - Permite otimizações internas feitas pelo Android para adequar o horário de disparo
 - Evita que o dispositivo tenha que acordar mais vezes do que o necessário
 - Economiza a bateria
- Alarmes são voltados para situações onde o código deve executar em determinado horário
 - Para outras operações envolvendo tempo, handlers são mais recomendados