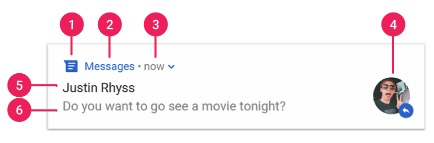
安卓推送通知

1. 概览

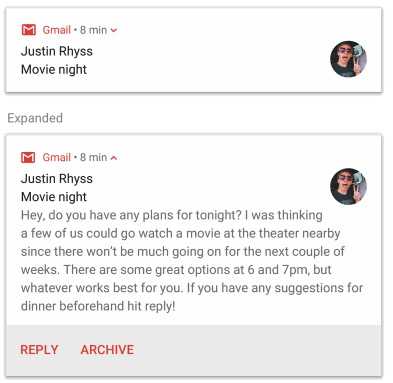
**1.下面展示了通知中的一些基本元素：**



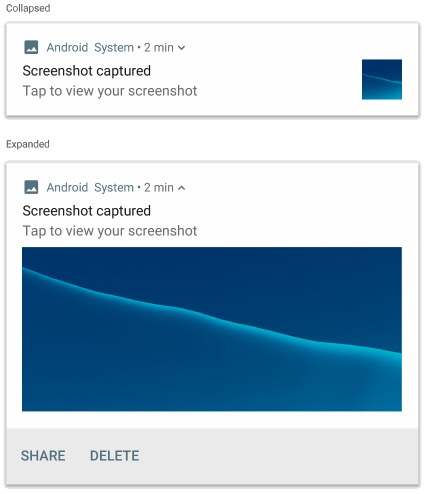
1. 1.小图标：此为必要图标，通过 NotificationCompat.Builder.[setSmallIcon()](https://developer.android.google.cn/reference/androidx/core/app/NotificationCompat.Builder" \l "setSmallIcon(int)) 设置。
2. 2.应用名称：由系统提供。
3. 3.推送时间：由系统提供，但可以通过 [setWhen(time)](https://developer.android.google.cn/reference/androidx/core/app/NotificationCompat.Builder" \l "setWhen(long)) 进行设置，time是以微秒为单位的时间戳（长整型）。也可以使用 [setShowWhen(false)](https://developer.android.google.cn/reference/androidx/core/app/NotificationCompat.Builder" \l "setShowWhen(boolean)) 将其隐藏。
4. 4.大图标：为可选图标（通常仅用于联系人照片，通过 [setLargeIcon()](https://developer.android.google.cn/reference/androidx/core/app/NotificationCompat.Builder" \l "setLargeIcon(android.graphics.Bitmap)) 设置。
5. 5.标题：为可选内容，通过 [setContentTitle()](https://developer.android.google.cn/reference/androidx/core/app/NotificationCompat.Builder" \l "setContentTitle(java.lang.CharSequence)) 设置。
6. 6.文本：为可选内容，通过 [setContentText()](https://developer.android.google.cn/reference/androidx/core/app/NotificationCompat.Builder" \l "setContentText(java.lang.CharSequence)) 设置。
7. **展开式通知**：

为了满足仅显示一行的要求，通知的文本内容会被截断。如果希望显示的通知内容更长，则可以通过调用setStyle()方法使通知展开成不同的风格。例如：

1. [NotificationCompat.BigTextStyle](https://developer.android.google.cn/reference/androidx/core/app/NotificationCompat.BigTextStyle)：长文本风格可以添加一大段文本。



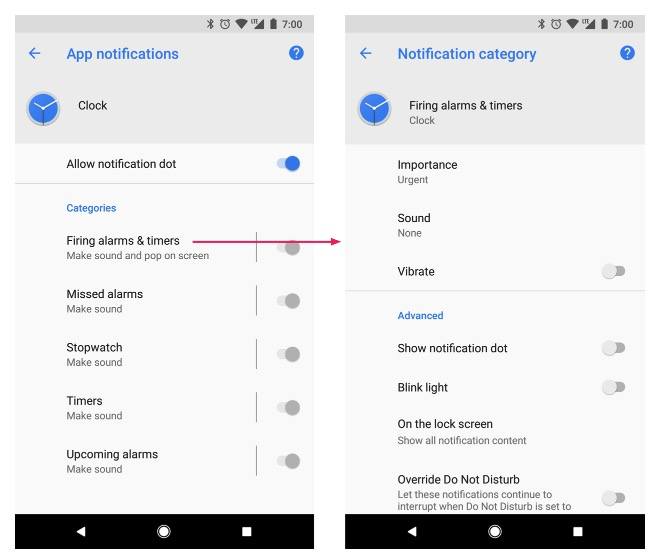
2 [NotificationCompat.BigPictureStyle](https://developer.android.google.cn/reference/androidx/core/app/NotificationCompat.BigPictureStyle)： 大图片风格可以添加一张展示的大图。



二、通知渠道

从 Android 8.0（API 级别 26）开始，所有通知都必须分配到一个通知渠道，否则通知将不会显示。通过将通知分配到渠道，用户可以停用特定通知渠道的通知（而不是停用*所有*通知），同时用户还可以修改每个渠道的视觉和听觉选项，所有这些操作都在 Android 系统设置中完成（用户也可以长按通知以更改关联渠道的行为）（如下图所示）。

在运行 Android 7.1（API 级别 25）及更低版本的设备上，用户仅可以按应用来管理通知（实际上，在 Android 7.1 或更低版本中每个应用仅有一个渠道）。



(左图是渠道列表， 右图是某特定渠道的设置界面）渠道演示

创建通知渠道后，便无法通过程序更改通知渠道的设置，此时用户拥有完全控制权。但是仍然可以通过程序更改渠道的名称和描述。

用途：我们可以为不同类型的通知各创建一个通知渠道，让不同类型的通知有不同的响应。

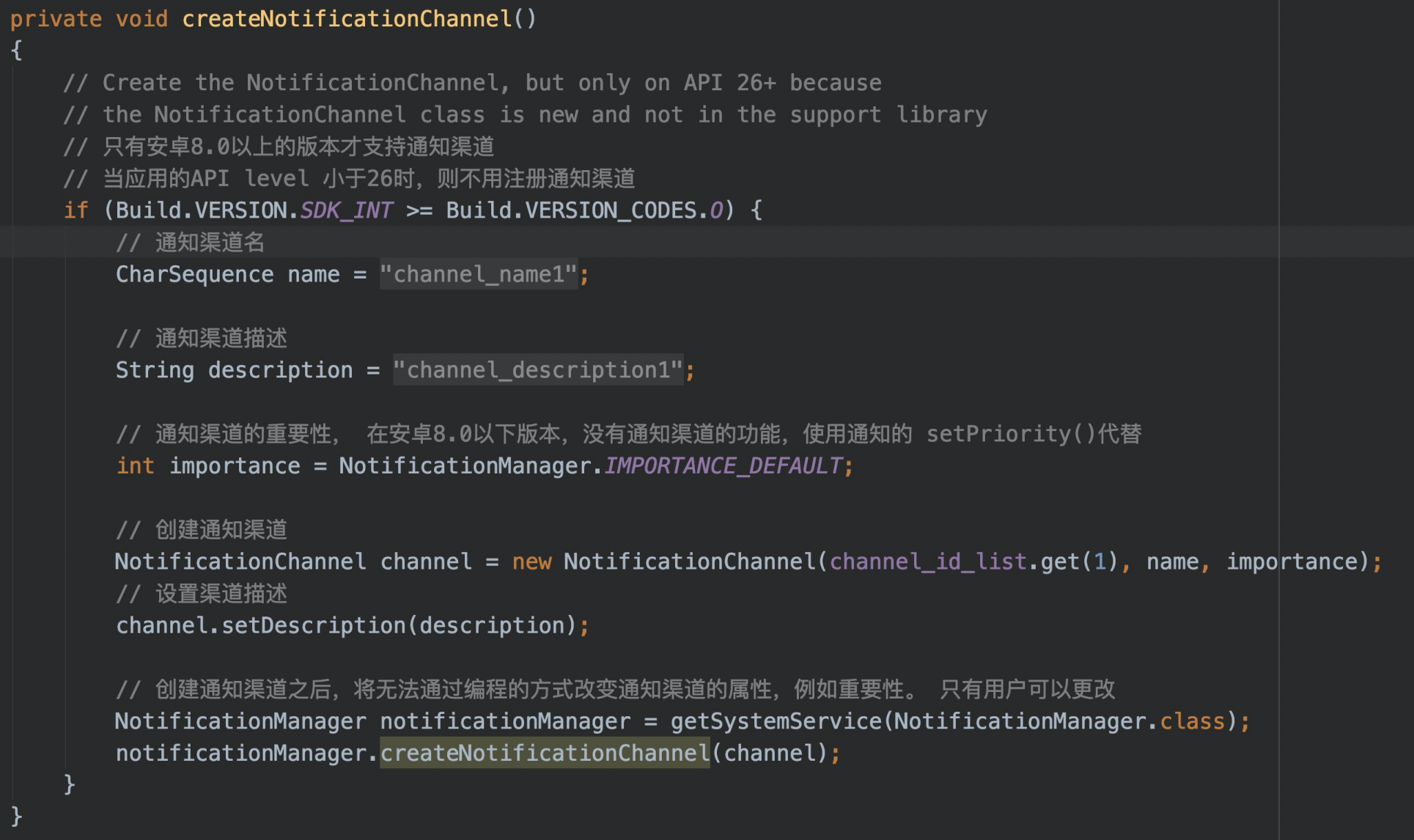
1. **创建通知渠道**

要创建通知渠道，请按以下步骤操作：

1.构建一个具有唯一渠道 ID、渠道名和重要性级别(importance)的 [NotificationChannel](https://developer.android.google.cn/reference/android/app/NotificationChannel) 对象，importance参数是[NotificationManager](https://developer.android.google.cn/reference/android/app/NotificationManager) 类中的其中一个常量，该参数决定任何属于该渠道的通知的具体表现行为。

2.（可选）使用 [setDescription()](https://developer.android.google.cn/reference/android/app/NotificationChannel" \l "setDescription(java.lang.String)) 指定用户在系统设置中看到的说明。

3.使用 NotificationManager.[createNotificationChannel()](https://developer.android.google.cn/reference/android/app/NotificationManager" \l "createNotificationChannel(android.app.NotificationChannel))方法，进行渠道的注册。



由于必须先创建通知渠道，然后才能在 Android 8.0 及更高版本的设备上发布任何通知，因此应在应用启动时立即执行这段代码。反复调用这段代码是安全的，因为创建现有通知渠道不会执行任何操作。

如果希望进一步自定义渠道的默认通知行为，可以调用 [NotificationChannel](https://developer.android.google.cn/reference/android/app/NotificationChannel) 的的 [enableLights()](https://developer.android.google.cn/reference/android/app/NotificationChannel" \l "enableLights(boolean))、[setLightColor()](https://developer.android.google.cn/reference/android/app/NotificationChannel" \l "setLightColor(int)) 和 [setVibrationPattern()](https://developer.android.google.cn/reference/android/app/NotificationChannel" \l "setVibrationPattern(long[]))【震动模式】 等方法。同样的，创建渠道后，将无法通过程序更改这些设置。

还可以通过调用 [createNotificationChannels()](https://developer.android.google.cn/reference/android/app/NotificationManager" \l "createNotificationChannels(java.util.List&lt;android.app.NotificationChannel&gt;)) 在一次操作中创建多个通知渠道。

1. **设置通知渠道的重要性**

设置通知渠道的重要性方式：

1. 可以在NotificationChannel的构造函数指定通知重要性。
2. 也可以通过setImportance()方法进行设置。

可以使用从 [IMPORTANCE\_NONE(0)](https://developer.android.google.cn/reference/android/app/NotificationManager" \l "IMPORTANCE_NONE) 到 [IMPORTANCE\_HIGH(4)](https://developer.android.google.cn/reference/android/app/NotificationManager" \l "IMPORTANCE_HIGH) 的五个重要性级别。为渠道指定的重要性级别会应用到在其中发布的所有通知消息。

因为Android 7.1（API 级别 25）或更低版本的设备不支持通知渠道，所以会忽略channel相关的设置。而是使用通知的优先级，所以针对每条通知应该调用[NotificationCompat](https://developer.android.google.cn/reference/androidx/core/app/NotificationCompat).Builder.[setPriority()](https://developer.android.google.cn/reference/androidx/core/app/NotificationCompat.Builder" \l "setPriority(int))，设置其优先级。

重要性 (NotificationManager.IMPORTANCE\_\*) 和优先级常量 (NotificationCompat.PRIORITY\_\*) 会映射到用户可见的重要性选项。以下是它们的对应关系：



演示

无论重要性级别如何，所有通知都会在通知栏中显示，并且[在启动器图标上会](https://developer.android.google.cn/guide/topics/ui/notifiers/notifications" \l "app_icon_badge)出现圆点提示。将渠道提交至 [NotificationManager](https://developer.android.google.cn/reference/android/app/NotificationManager) 后，就无法更改重要性级别，只有用户可以随时更改应用的通知渠道的偏好设置。

1. **获取通知渠道设置**

用户可以修改通知渠道的设置，其中包括振动和提醒提示音等行为。可以通过以下方式了解用户对通知渠道所应用的设置：

1.使用 [getNotificationChannel()](https://developer.android.google.cn/reference/android/app/NotificationManager" \l "getNotificationChannel(java.lang.String)) 或 [getNotificationChannels()](https://developer.android.google.cn/reference/android/app/NotificationManager" \l "getNotificationChannels()) 获取 [NotificationChannel](https://developer.android.google.cn/reference/android/app/NotificationChannel) 对象。

2.查询特定的渠道设置，使用渠道对象调用 [getVibrationPattern()](https://developer.android.google.cn/reference/android/app/NotificationChannel" \l "getVibrationPattern())、[getSound()](https://developer.android.google.cn/reference/android/app/NotificationChannel" \l "getSound()) 和 [getImportance()](https://developer.android.google.cn/reference/android/app/NotificationChannel" \l "getImportance())等方法。

1. **删除通知渠道**

可以通过调用NotificationManager.[deleteNotificationChannel(channel\_id)](https://developer.android.google.cn/reference/android/app/NotificationManager" \l "deleteNotificationChannel(java.lang.String)) 删除通知渠道。

代码如下：

  NotificationManager notificationManager =  
            (NotificationManager) getSystemService(Context.NOTIFICATION\_SERVICE);  
    String id = "my\_channel\_01";  
    notificationManager.deleteNotificationChannel(id);

1. 创建通知

示例的代码使用 Android 支持库中的 [NotificationCompat](https://developer.android.google.cn/reference/androidx/core/app/NotificationCompat) 相关的API。这些API可以添加在较新版本 Android上的可用的功能，同时仍向后兼容Android 4.0（API 级别 14）。。

1. **添加支持库**

在build.gradle 文件是否包含以下依赖项：

dependencies {  
        implementation "com.android.support:support-compat:28.0.0"  
    }

注意：com.android.support 组中的其他库也将 support-compat 包含为传递依赖项。因此，如果已在使用其他支持库 API，或许能够直接访问 [NotificationCompat](https://developer.android.google.cn/reference/androidx/core/app/NotificationCompat)，而不需要具有与以上所示完全一样的依赖项。

1. **创建基本通知**

首先，需要使用 [NotificationCompat.Builder](https://developer.android.google.cn/reference/androidx/core/app/NotificationCompat.Builder) 对象设置通知内容和渠道。以下是一些基本设置：

* 小图标，通过 [setSmallIcon()](https://developer.android.google.cn/reference/androidx/core/app/NotificationCompat.Builder" \l "setSmallIcon(int)) 设置。这是必需设置的。
* 标题，通过 [setContentTitle()](https://developer.android.google.cn/reference/androidx/core/app/NotificationCompat.Builder" \l "setContentTitle(java.lang.CharSequence)) 设置。
* 正文文本，通过 [setContentText()](https://developer.android.google.cn/reference/androidx/core/app/NotificationCompat.Builder" \l "setContentText(java.lang.CharSequence)) 设置。
* 通知优先级，通过 [setPriority()](https://developer.android.google.cn/reference/androidx/core/app/NotificationCompat.Builder" \l "setPriority(int)) 设置。优先级确定通知在 Android 7.1 和更低版本上的干扰程度。（对于 Android 8.0 和更高版本，必须设置渠道重要性，忽略优先级。）

代码如下：

    NotificationCompat.Builder builder = new NotificationCompat.Builder(this, CHANNEL\_ID)  
            .setSmallIcon(R.drawable.notification\_icon)  
            .setContentTitle(textTitle)  
            .setContentText(textContent)  
            .setPriority(NotificationCompat.PRIORITY\_DEFAULT);

注意：[NotificationCompat.Builder](https://developer.android.google.cn/reference/androidx/core/app/NotificationCompat.Builder) 构造函数要求提供渠道 ID。这是兼容 Android 8.0（API 级别 26）及更高版本所必需的，因为8.0以上版本要求通知必须有一个渠道ID，如果通知没有设置渠道ID或设置了一个没有注册的渠道ID则通知不会被显示。渠道ID会被较旧版本(Android 8.0以下)忽略。

默认情况下，通知的文字内容会被截断放在一行。如果您想要更长的通知，可以使用 [setStyle()](https://developer.android.google.cn/reference/androidx/core/app/NotificationCompat.Builder" \l "setStyle(android.support.v4.app.NotificationCompat.Style)) 添加样式模板来启用可展开的通知。下面介绍一下游戏当中最常用展开式通知的样式：演示

NotificationCompat.Builder builder = new NotificationCompat.Builder(this, CHANNEL\_ID)  
            .setSmallIcon(R.drawable.notification\_icon)  
            .setContentTitle("My notification")  
            .setContentText("Much longer text that cannot fit one line...")  
            **.setStyle(new NotificationCompat.BigTextStyle()  
                    .bigText("Much longer text that cannot fit one line...")) // 设置长文本**  
            .setPriority(NotificationCompat.PRIORITY\_DEFAULT);

1. **设置通知的点按操作**

每个通知都应该对点按操作做出响应，通常是在应用中打开对应于该通知的 Activity。所以必须指定通过 [PendingIntent](https://developer.android.google.cn/reference/android/app/PendingIntent) 对象定义的内容 Intent，并将其传递给 [setContentIntent()](https://developer.android.google.cn/reference/androidx/core/app/NotificationCompat.Builder" \l "setContentIntent(android.app.PendingIntent))。代码如下：

// 创建一个跳转到指定页面到intent对象

**Intent intent = new Intent(this, Test.class);  
    PendingIntent pendingIntent = PendingIntent.getActivity(this, 0, intent, 0);**  
    NotificationCompat.Builder builder = new NotificationCompat.Builder(this, CHANNEL\_ID)  
            .setSmallIcon(R.drawable.notification\_icon)  
            .setContentTitle("My notification")  
            .setContentText("Hello World!")  
            .setPriority(NotificationCompat.PRIORITY\_DEFAULT)  
            // 设置 当点击该通知时所要触发的对应操作的intent  
            **.setContentIntent(pendingIntent)**

// 用户点按通知后自动移除通知 演示

.setAutoCancel(true);

参数解释：

PendingIntent.getActivity（[Context](http://developer.android.com/reference/android/content/Context.html) context, int requestCode, [Intent](http://developer.android.com/reference/android/content/Intent.html) intent, int flags）

第一个参数连接上下文的context

第二个参数是对PendingIntent的描述，请求码不同Intent就不同，可以通过requestCode和context确定系统中是否存在相同描述的PendingIntent对象。

第三个参数是一个Intent对象，包含跳转目标

第四个参数有4种状态

FLAG\_CANCEL\_CURRENT、FLAG\_UPDATE\_CURRENT、FLAG\_NO\_CREATE、FLAG\_ONE\_SHOT

常用的有2种：

FLAG\_CANCEL\_CURRENT：如果当前系统中已经存在一个相同描述的PendingIntent对象，那么就先将已有的PendingIntent移除，然后重新生成一个PendingIntent对象。

FLAG\_UPDATE\_CURRENT：如果系统中有一个描述相同的PendingInent，那么系统将使用该PendingIntent对象，但是会使用新的Intent来更新这个PendingIntent中的Intent对象数据，例如更新Intent中的Extras。演示

1. **显示通知**

要显示通知，需要调用 [NotificationManagerCompat.notify()](https://developer.android.google.cn/reference/androidx/core/app/NotificationManagerCompat" \l "notify(int, android.app.Notification))，并将通知的唯一 ID 和 [NotificationCompat.Builder.build()](https://developer.android.google.cn/reference/androidx/core/app/NotificationCompat.Builder" \l "build()) 的结果（ Notification对象）传递给它。例如：

NotificationManagerCompat notificationManager = NotificationManagerCompat.from(this);  
    notificationManager.notify(notificationId, builder.build());

注意：记得保存您传递到 [NotificationManagerCompat.notify()](https://developer.android.google.cn/reference/androidx/core/app/NotificationManagerCompat" \l "notify(int, android.app.Notification)) 的通知 ID，因为如果之后您想要[更新](https://developer.android.google.cn/training/notify-user/build-notification" \l "Updating)或[移除通知](https://developer.android.google.cn/training/notify-user/build-notification" \l "Removing)，将需要使用这个 ID。

1. **更新通知**演示

要更新通知，只需要再次调用[NotificationManagerCompat.notify(id, notification)](https://developer.android.google.cn/reference/androidx/core/app/NotificationManagerCompat" \l "notify(int, android.app.Notification))，并且将需要更新的通知ID和新的Notification对象作为参数传入即可。如果该ID对应的通知不存在，则会创建一个新的通知。

如果只希望在第一次创建通知的时候通过声音、震动、视觉提示用户，而之后的更新不提示用户，可以通过调用[setOnlyAlertOnce()](https://developer.android.google.cn/reference/androidx/core/app/NotificationCompat.Builder" \l "setOnlyAlertOnce(boolean))方法实现。

1. 移除通知 演示

可以通过以下方式移除通知：

1.用户自己关闭通知。

2.用户点击通知，且在创建的通知的时候调用了[setAutoCancel()](https://developer.android.google.cn/reference/androidx/core/app/NotificationCompat.Builder" \l "setAutoCancel(boolean))方法。

3.对特定通知ID调用[cancel()](https://developer.android.google.cn/reference/android/app/NotificationManager" \l "cancel(int)) 方法。例如NofiticationManager.cancel(Id)。

4.调用[cancelAll()](https://developer.android.google.cn/reference/android/app/NotificationManager" \l "cancel(int))方法。移除之前发出的所有通知。

5.如果创建通知的时候使用 [setTimeoutAfter()](https://developer.android.google.cn/reference/androidx/core/app/NotificationCompat.Builder" \l "setTimeoutAfter(long)) 设置了超时，系统会在持续时间过后取消通知。

7.设置锁定屏幕下的通知显示范围

要设置锁定屏幕中通知的可见详情级别，需要调用 [setVisibility()](https://developer.android.google.cn/reference/androidx/core/app/NotificationCompat.Builder" \l "setVisibility(int)) 并指定以下值之一：

1.[VISIBILITY\_PUBLIC](https://developer.android.google.cn/reference/androidx/core/app/NotificationCompat" \l "VISIBILITY_PUBLIC) 显示通知的完整内容。

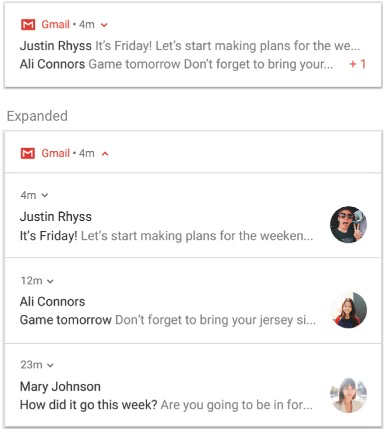
2.[VISIBILITY\_SECRET](https://developer.android.google.cn/reference/androidx/core/app/NotificationCompat" \l "VISIBILITY_SECRET) 不在锁定屏幕上显示该通知的任何部分。

3.[VISIBILITY\_PRIVATE](https://developer.android.google.cn/reference/androidx/core/app/NotificationCompat" \l "VISIBILITY_PRIVATE) 显示基本信息，例如通知图标和内容标题，但隐藏通知的完整内容。

但是，对于通知在锁定屏幕上是否可见，用户始终拥有最终控制权，可以根据应用的通知渠道来设置可见级别。

1. 创建一组通知

从 Android 7.0（API 级别 24）开始，可以在一个通知组中显示相关通知（以前称为“捆绑式”通知）。下图为收起和展开后的通知组：



设置通知组步骤：

1. 为该通知组定义一个通知组的唯一标识符字符串。
2. 对于想要添加到通知组中的每条通知，只需调用 [setGroup()](https://developer.android.google.cn/reference/androidx/core/app/NotificationCompat.Builder" \l "setGroup(java.lang.String)) 并传入通知组名称即可。

3.额外创建一个(只能创建一个)通知作为该通知组的通知摘要，调用setGroupSummary(true)设置通知摘要。

例如：

**String GROUP\_KEY\_WORK\_EMAIL = "com.android.example.WORK\_EMAIL";**  
    Notification newMessageNotification = new NotificationCompat.Builder(MainActivity.this, CHANNEL\_ID)  
            .setSmallIcon(R.drawable.new\_mail)  
            .setContentTitle(emailObject.getSenderName())  
            .setContentText(emailObject.getSubject())  
            .setLargeIcon(emailObject.getSenderAvatar())  
            **.setGroup(GROUP\_KEY\_WORK\_EMAIL)**

**.setGroupSummary**(true)// 仅当要将该通知作为通知组摘要时才需要设置。  
            .build();

演示