使用 Notification

1. 实训目的

（1）了解通知的两种显示方式、通知的进阶技巧；

（2）掌握通知的创建方法；

（3）掌握点击通知事件；

（4）掌握通知的取消方法。

2. 实训要求

（1）创建通知；

（2）编写通知的点击事件；

（3）取消通知；

（4）完成实验报告。

3. 实训指导

（1）创建通知

1）创建 NotificationManager 对通知进行管理。

```
NotificationManager
manager=(NotificationManager)getSystemService(NOTIFICATION_SERVICE);
```

2）使用 Builder 构造器创建 Notification 对象。

```
Notification notification=new NotificationCompat.Builder(context).build();
```

进行一些基本的设置：

```
Notification notification=new NotificationCompat.Builder(this)
        .setContentTitle("this is content title")//用于指定通知的标题内容
        .setContentText("this is content text")//用于指定通知的正文内容
        .setWhen(System.currentTimeMillis())//用于指定通知被创建的时间，以毫秒
为单位
        .setSmallIcon(R.mipmap.ic_launcher)//用于设置通知的小图标
       .setLargeIcon(BitmapFactory.decodeResource(getResources(),R.mipmap.ic_l
auncher))
         .build();
```

3）调用 NotificationManager 的 notify()方法让通知显示出来。

```
manager.notify(1,notification);
```

（2）通知的点击效果

使用 PendingIntent

Intent 表示立刻执行某个操作,PendingIntent 更加倾向于在某个合适的时机去执行某个动作。得到一个 pendingIntent 对象，使用方法类的静态方法 getActivity(Context, int, Intent, int)，getBroadcast(Context, int, Intent, int)，getService(Context, int, Intent, int) 分别对应着 Intent 的 3 个行为，跳转到一个 activity 组件、打开一个广播组件和打开一个服务组件。参数有 4 个，比较重要的事第三个和第一个，其次是第四个和第二个。可以看到，要得到这个对象，必须传入一个 Intent 作为参数，必须有 context 作为参数。其余两个 int 传入 0 即可。

案例：通知的点击效果

第一步：Layout 中的 activity_main.xml(仅设置触发按钮)：

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
xmlns:tools="http://schemas.android.com/tools"
android:id="@+id/activity_main"
android:layout_width="match_parent"
android:layout_height="match_parent" >
<Button
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
   android:text="send notice"
    android:id="@+id/send_notice"
    android:onClick="show1"/>
</LinearLayout>
```

第二步：Layout 中的跳转页面 activity_content.xml(仅设置显示文本)：

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
xmlns:tools="http://schemas.android.com/tools"
android:id="@+id/activity_notification"
```

```
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context="com.example.cyy.notificationtest.NotificationActivity">
    <TextView
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:text="this is notification layout"
        android:textSize="24sp"
        android:layout_centerInParent="true"/>
</RelativeLayout>
```

第三步：java（主界面按钮的点击事件）

实现代码 MainActivity.java：

```
    public class MainActivity extends AppCompatActivity implements
View.OnClickListener{
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        Button sendNotice=(Button)findViewById(R.id.send_notice);
      sendNotice.setOnClickListener(this);
    }
    @Override
    public void onClick(View view) {
        switch (view.getId()){
            case R.id.send_notice:
                Intent intent=new Intent(this,NotificationActivity.class);
                PendingIntent pi=PendingIntent.getActivity(this,0,intent,0);
                NotificationManager
manager=(NotificationManager)getSystemService(NOTIFICATION_SERVICE);
```

```
                Notification notification=new NotificationCompat.Builder(this)
                .setContentTitle("this is content title")//用于指定通知的标题内容
                .setContentText("this is content text")//用于指定通知的正文内容
                .setWhen(System.currentTimeMillis())//用于指定通知被创建的时间,以
毫秒为单位
                .setSmallIcon(R.mipmap.ic_launcher)//用于设置通知的小图标
                .setLargeIcon(BitmapFactory.decodeResource(getResources(),R.mipm
ap.ic_launcher))
    //setContentIntent()方法传入 PendingIntent 实例
                .setContentIntent(pi)
                .build();
                manager.notify(1,notification);
                break;
            default:
                break;
        }
    }
}
```

第四步：取消通知

第一种方法:NotificationCompat.Builder 中再连一个 setAutoCancel()方法。

```
Notification notification=new NotificationCompat.Builder(this)
...
.setAutoCancel(true)
```

第二种方法：调用 NotificationManager 的 cancel()方法

```
protected void onCreate(Bundle savedInstanceState) {
super.onCreate(savedInstanceState);
setContentView(R.layout.activity_notification);
NotificationManager
```

```
manager=(NotificationManager)getSystemService(NOTIFICATION_SERVICE);
   //参数 1 为这条通知设置的 id 为 1
    manager.cancel(1);
}
```

（3）通知的进阶技巧

NotificationCompat.Builder 中提供了非常丰富的的 API 来让我们创建出更加多样的通知效果

在通知发出的时候播放一段音频：

```
.setSound(Uri.fromFile(new File("/system/media/audio/ringtones/Luna.ogg")))
```

在通知来的时候让手机进行振动：

```
.setVibrate(new long[] {0, 1000, 1000, 1000})
```

不过要想控制手机振动还需要声明权限：

```
<mainfest xmlns:android="http://schemas.android.com/apk/res/android"
      ...
    <uses-permission android:name="android.permission.VIBRATE"/>
    ...
</manifest>
```

在通知来的时候让手机 LED 灯显示：

```
.setLights(Color.GREEN, 1000, 1000)
```

也可以直接使用通知的默认效果，它会根据当前的手机环境来决定播放什么铃声，以及如何振动，写法如下：

```
.setDefaults(NotificationCompat.DEFAULT_ALL)
```