# 第5章 对话框、菜单和状态栏消息

作为用户交互的重要工具和手段，对话框、菜单和状态栏消息在UI设计中起着重要的作用。对话框是一种显示在Activity上的界面元素，一般用于给出提示信息或弹出一个与主进程直接相关的子程序；菜单能够在不占用界面空间的前提下为应用程序提供相应的功能和界面；状态栏消息是一种具有全局效果的提醒机制，不会打断用户当前的操作。

## 5.1 对 话 框

在Android中，对话框是一种显示在Activity上的界面元素，是作为Activity的一部分被创建和显示的。当显示对话框时，当前Activity失去焦点而由对话框负责所有的交互。一般来说，对话框用于给出提示信息或弹出一个与主进程直接相关的子程序。常用的对话框有提示对话框（AlertDialog）、进度对话框（ProgressDialog）、日期选择对话框（DatePickerDialog）、时间选择对话框（TimePickerDialog）等，其中AlertDialog是最常用的对话框。各类对话框与所属类之间的继承关系如图5-1所示。

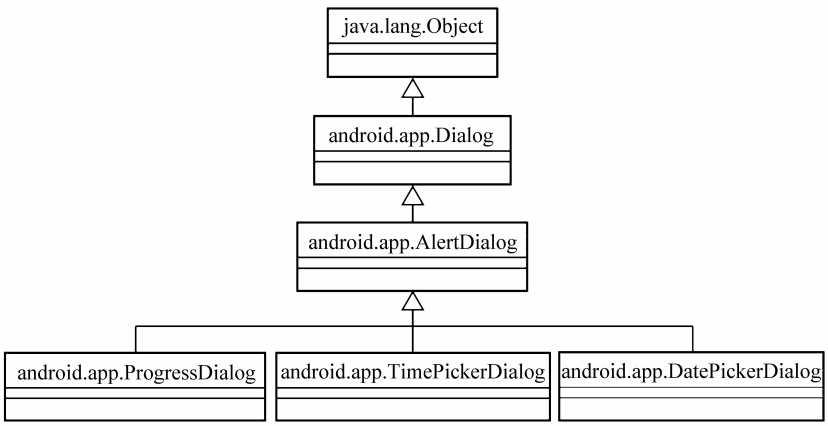


图5-1 各类对话框与所属类之间的继承关系

在程序中通过调用回调方法onCreateDialog()可以完成对话框实例的创建和显示。该方法需要传入代表对话框的ID参数，当对话框第一次被显示时会创建此ID的Dialog实例，之后不再重复创建该实例。

通常调用 showDialog(int)方法显示对话框，调用 dismissDialog(int)方法关闭对话框。每次对话框被显示之前都会调用 onPrepareDialog()方法，所以如果不重写该方法，每次显示的对话框都是最初创建的那个。也可以直接调用 Dialog 对象的 dismiss()或 cancel()方法关闭对话框。但通过这种方法关闭的对话框并不会彻底销毁，Android 会在后台保留其状态，因此可以为对话框设置 DialogInterface. onDismissListener()的监听，并重写其中的 onDismiss()方法来解决这一问题。如果需要让对话框在关闭之前彻底被清除，可以调用 removeDialog()方法并传入 Dialog 的 ID 值来彻底释放对话框资源。

### 5.1.1　提示对话框 AlterDialog

本节介绍对话框中最常用的 AlterDialog 的设计方法。AlterDialog 是一个消息提示对话框，能构造默认的 3 个按钮，分别为“是”“否”和“取消”。创建 AlertDialog 对话框的主要步骤如下。

**步骤 1**：获得 AlertDialog 的静态内部类 Builder 对象，并由该对象创建对话框。

**步骤 2**：通过 Builder 对象设置对话框的标题、文字等属性。表 5-1 列出了 Builder 对象的部分常用方法。

**表 5-1　Builder 对象的部分常用方法**

| 方　法　名 | 说　　明 |
|---|---|
| setIcon() | 设置对话框的图标 |
| setTitle() | 设置对话框的标题 |
| setMessage() | 设置对话框的提示文字 |
| setItems() | 设置对话框要显示的一个列表 |
| setSingleChoiceItems() | 设置对话框显示一个单选的列表 |
| setMultiChoiceItems() | 设置对话框显示一系列的复选框 |
| setPositiveButton() | 给对话框添加“是”按钮 |
| setNegativeButton() | 给对话框添加“否”按钮 |
| setNeutralButton() | 给对话框添加“取消”按钮 |
| setView() | 给对话框设置自定义样式 |
| create() | 创建对话框 |
| show() | 显示对话框 |

可通过调用 setIcon()方法设置显示在对话框标题左侧的图标，setTitle()方法设置对话框的标题，setMessage()方法设置对话框中显示的文字信息，setView()方法设置对话框中显示的内容。其中 setView()方法的参数为一个 View 实例名，调用该方法可以在对话框中显示一个布局或 Widget 控件对象。

**步骤 3**：设置对话框的按钮以及单击按钮将要响应的事件处理程序。

对话框中可以有“是”“否”和“取消”3 个按钮，生成器 Builder 负责设置对话框上的按

钮,并为按钮注册 OnClickListener 监听器。OnClickListener 在 android. content. DialogInterface 包中,事件处理方法是 onClick()方法。无论用户点击哪一个按钮,对话框都会消失,并导致接口中的 onClick()方法被调用。onClick()方法的定义如下。

```
abstract void onClick(DialogInterface dialog, int which)
```

其中,参数 dialog 就是当前要消失的对话框,参数 which 是用户点击的按钮,其取值可以是 DialogInterface. BUTTON _ NEGATIVE、DialogInterface. BUTTON _ NEUTRAL、DialogInterface. BUTTON_POSITIVE。

**步骤 4**:调用 Builder 对象的 create()方法创建对话框 AlertDialog 对象。

**步骤 5**:调用 AlertDialog 对象或 Builder 对象的 show()方法显示对话框。调用 hide()方法可以隐藏对话框。

如果不希望用户点击设备的"返回"按钮使对话框消失,而是要求用户必须点击对话框中的按钮,则可以通过调用 AlertDialog 对象的 setCancelable(false)方法来进行设置。

### 1. 创建简单的提示对话框

简单的提示对话框仅包括文字标题、标题左侧的图标、文字提示信息、按钮等基本元素。

**【例 5-1】** 工程 Demo_05_AlertDialog 演示了 AlterDialog 对话框的用法。

当点击按钮后,弹出 AlertDialog 对话框,运行效果如图 5-2 所示,主要代码见代码段 5-1。

**代码段 5-1 创建简单的 AlterDialog**

```
btnstart.setOnClickListener(new View.OnClickListener() {
    public void onClick(View v) {
        AlertDialog.Builder myDialog =new AlertDialog.Builder(MainActivity.this);
        //创建 AlertDialog.Builder 对象
        myDialog.setIcon(R.mipmap.ic_launcher);              //设置图标
        myDialog.setTitle("提示");                           //设置标题
        myDialog.setMessage("这是一个 AlertDialog 对话框!"); //设置显示消息
        myDialog.setNegativeButton("取消", null);            //"取消"按钮
        myDialog.setPositiveButton("确定", new OnClickListener() {
            @Override
            public void onClick(DialogInterface dialog, int which) {
                Toast.makeText(getApplicationContext(), "您点击了确定按钮!",
                    Toast.LENGTH_LONG).show();
            }
        });                                                  //"确定"按钮
        AlertDialog alertDialog=myDialog.create();
        alertDialog.show();
    }
});
```

### 2. 创建其他风格的提示对话框

除了按钮对话框，还可以创建列表、单选、多选对话框。通过调用 setItems()方法，可以在 AlertDialog 中添加列表项；通过调用 setSingleChoiceItems()/setMultiChoiceItems()方法，可以在 AlertDialog 中添加单选/多选按钮。

【例 5-2】　工程 Demo_05_ListDialog 演示了列表对话框的用法。

在 values 下 arrays.xml 文件中定义字符串数组，内容如代码段 5-2 所示。

**代码段 5-2　定义字符串数组**

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <string-array name ="msa">
        <item>音乐</item>
        <item>体育</item>
        <item>美术</item>
    </string-array>
</resources>
```

当点击按钮后，弹出 AlertDialog 对话框，运行效果如图 5-3 所示。当在对话框中选择了一项之后，Activity 中 TextView 的显示内容随之更新。创建 AlertDialog 对话框的主要代码见代码段 5-3。


图 5-2　AlterDialog 提示对话框


图 5-3　列表提示框

**代码段 5-3　创建列表对话框**

```
btnstart.setOnClickListener(new View.OnClickListener() {
    public void onClick(View v) {
        AlertDialog.Builder myDialog =new AlertDialog.Builder(MainActivity.this);
        myDialog
            .setTitle("列表提示框,请选择")      //设置对话框的标题
```

```
            .setItems(                              //用字符串数组设置列表中的各个属性
                R.array.favor, new DialogInterface.OnClickListener() {
                                                    //设置监听器
                    public void onClick(DialogInterface dialog, int which) {
                        TextView massage = (TextView) findViewById(R.id.text1);
                        massage.setText("您选择了:"+getResources().getStringArray
                            (R.array.favor)[which]);
                    }
                })
            .show();
    }
});
```

**【例 5-3】** 工程 Demo_05_RadioButtonDialog 演示了单选按钮对话框的用法。

本例使用与例 5-2 工程中相同的数组资源,当点击按钮后,弹出单选按钮对话框,运行效果如图 5-4 所示。创建单选按钮对话框的主要代码见代码段 5-4。


图 5-4 创建单选按钮提示框

**代码段 5-4 创建单选按钮对话框**

```
btnstart.setOnClickListener(new View.OnClickListener() {
    public void onClick(View v) {
        AlertDialog.Builder myDialog =new AlertDialog.Builder(MainActivity.this);
        myDialog
            .setTitle("单选列表提示框,请选择")              //设置对话框的标题
            .setSingleChoiceItems(                      //设置单选列表选项
                R.array.favor, 0, new DialogInterface.OnClickListener() {
                    public void onClick(DialogInterface dialog, int which) {
                        TextView message = (TextView) findViewById(R.id.text1);
```

```
                        message.setText("您选择了 1:"+getResources().
                            getStringArray(R.array.favor)[which]);
                }
            });
            myDialog.setPositiveButton("确定",null);
            myDialog.setNegativeButton("取消",null);
            myDialog.show();
        }
    });
```

### 3. 创建具有复杂界面的提示对话框

可以将定制的 View 作为其内容显示在对话框中，这样就可以实现在对话框中显示较为复杂的内容。

**【例 5-4】** 工程 Demo_05_ViewDialog 演示了如何将定制的 View 作为其内容显示在对话框中，该程序通过点击按钮来弹出一个用来显示登录界面的对话框。

为了实现上述功能，需要为对话框设计相应的布局。在 res\layout 下创建一个 XML 布局文件 dialog_view.xml，布局的主要内容是提示文字及对应的两个文本输入框，分别用于输入用户名和密码，代码见代码段 5-5。对话框中的“退出”和“确定”等按钮不需要在布局文件中设定，而是在该对话框被实例化后通过调用 setPositiveButton()和 setNegativeButton()方法来添加，并在其中设定点击“退出”按钮和“确定”按钮对应的事件处理程序。

**代码段 5-5　dialog_view.xml 布局**

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical"
    android:layout_width="match_parent"
    android:layout_height="match_parent">
    <TextView
        android:id="@+id/username_view"
        android:layout_height="wrap_content"
        android:layout_width="wrap_content"
        android:layout_marginTop="20dip"
        android:layout_marginLeft="20dip"
        android:layout_marginRight="20dip"
        android:text="用户名:"/>
    <EditText
        android:id="@+id/username_edit"
        android:layout_height="wrap_content"
        android:layout_width="match_parent"
        android:layout_marginLeft="20dip"
```

```
        android:layout_marginRight="20dip" /><!--对应用户名的输入框 -->
    <TextView
        android:id="@+id/password_view"
        android:layout_height="wrap_content"
        android:layout_width="wrap_content"
        android:layout_marginTop="20dip"
        android:layout_marginLeft="20dip"
        android:layout_marginRight="20dip"
        android:text="密码:"/>
    <EditText
        android:id="@+id/password_edit"
        android:layout_height="wrap_content"
        android:layout_width="match_parent"
        android:layout_marginLeft="20dip"
        android:layout_marginRight="20dip"
        android:inputType="textPassword"/><!--对应"密码"的输入框 -->
</LinearLayout>
```

示例程序中,当点击"弹出对话框"按钮后弹出 AlertDialog 对话框,实际运行效果如图 5-5 所示。创建对话框的主要代码见代码段 5-6。


图 5-5 显示复杂界面的提示对话框

如果主界面中按钮被点击,则定义一个 LayoutInflater 类的实例。LayoutInflater 类的作用类似于 findViewById(),不同点是 LayoutInflater 是用来引入 layout 下的 XML 布局文件并且实例化,而 findViewById()是引入 XML 文件中定义的具体 Widget 控件对象(如 Button、TextView 等)。这里通过调用 LayoutInflater 实例的 inflate()方法引入 XML 布局文件 dialog_view. xml,然后通过调用 AlertDialog 对象的 setView(View)方法,在对话框中显示这个布局文件。

**代码段 5-6　定义对话框及其按钮功能**

```
btnstart.setOnClickListener(new View.OnClickListener() {
    public void onClick(View v) {
        LayoutInflater dialogInflater =LayoutInflater.from(MainActivity.this);
        final View myViewOnDialog =dialogInflater.inflate(R.layout.dialog_
                view, null);                                    //引入布局
        AlertDialog myDialogInstance =new AlertDialog.Builder(MainActivity.this)
            .setTitle("用户登录界面")
            .setView(myViewOnDialog)
                            //参数为上面定义的 View 实例名,显示 dialog_view 布局
            .setPositiveButton("确定", new DialogInterface.OnClickListener() {
                                                                //"确定"按钮
                public void onClick(DialogInterface dialog, int whichButton) {
                                                                //侦听是否点击
                    Toast.makeText(getApplicationContext(), "感谢您输入了信息,
                        再见",Toast.LENGTH_LONG).show();
                }
            })
            .setNegativeButton("退出", new DialogInterface.OnClickListener() {
                                                                //"退出"按钮
                public void onClick(DialogInterface dialog, int whichButton) {
                                                                //是否点击
                    MainActivity.this.finish();                 //退出程序
                }
            })
            .create();
        myDialogInstance.show();                                //显示对话框
    }
});
```

## 5.1.2　进度条对话框 ProcessDialog

进度条对话框 ProgressDialog 除了 AlertDialog 的功能外，还能显示进度圈或进度条。当进行一个比较耗时的操作时，弹出一个 ProgressDialog 对话框可以使界面更友好。

可以直接通过 new 的方式来创建一个 ProgressDialog，通过调用 setProgressStyle()方法设置进度条的样式。进度条有两种样式，一种是水平的进度条，另一种是圆圈进度条。创建进度条后，一般会启动另外一个线程调用 incrementProgressBy()方法来设置进度条上显示的进度。

**【例 5-5】** 示例工程 Demo_05_ProcessDialog，演示了两种样式的进度条对话框。

定义进度条对话框的主要代码如代码段 5-7 所示，两种进度条的运行结果如图 5-6 所示。

**代码段 5-7　定义进度条对话框**

```
button1.setOnClickListener(new Button.OnClickListener() {
    public void onClick(View v) {
        ProgressDialog progressDialog =new ProgressDialog(MainActivity.this);
        //实例化一个 ProgressDialog
        progressDialog.setTitle("提示信息");
        progressDialog.setMessage("正在下载中,请稍候......");
        progressDialog.setProgressStyle(ProgressDialog.STYLE_SPINNER);
        //设置 ProgressDialog 的显示样式,STYLE_SPINNER 代表的是圆圈进度条
        progressDialog.show();                                    //显示进度对话框
    }
});
button2.setOnClickListener(new Button.OnClickListener() {
    public void onClick(View v) {
        ProgressDialog progressDialog =new ProgressDialog(MainActivity.this);
        progressDialog.setTitle("提示信息");
        progressDialog.setMessage("正在下载中,请稍候......");
        //设置最大进度,ProgressDialog 的进度范围是 1~10 000
        progressDialog.setMax(100);
        progressDialog.setProgressStyle(ProgressDialog.STYLE_HORIZONTAL);
        //设置 ProgressDialog 的显示样式,STYLE_HORIZONTAL 代表的是水平进度条
        progressDialog.show();
    }
});
```

本示例程序中,点击屏幕其他部分,进度条对话框就会消失。如果希望点击时不消失,可以调用 ProgressDialog 对象的 setCancelable(false)方法,这样对话框就不能被取消。


图 5-6　两种进度条对话框

### 5.1.3　日期和时间选择对话框

Android 系统提供了一些与日期和时间有关的控件，常用的有 DatePicker、TimePicker、DatePickerDialog、TimePickerDialog 等。DatePicker 和 TimePicker 用来实现日期和时间输入，DatePickerDialog 和 TimePickerDialog 用来显示日期选择和时间选择对话框。

**1. DatePicker 和 TimePicker**

DatePicker 和 TimePicker 都继承自 android. widget. FrameLayout 类，在 android. widget 包中。DatePicker 用来实现日期的输入设置，日期的设置范围为 1900 年 1 月 1 日至 2100 年 12 月 31 日。改变日期会触发 onDateChanged 事件，所以要监听日期值的改变，需要实现接口 android. widget. DatePicker. OnDateChangedListener 中的 onDateChanged ( ) 方法。TimePicker 向用户显示一天中的时间，并允许用户进行选择。时间的改变会触发 OnTimeChanged 事件，所以要监听时间值的改变，需要实现接口 android. widget. TimePicker. OnTimeChangedListener 中的 onTimeChanged()方法。

**【例 5-6】** 示例工程 Demo_05_DateAndTimePicker 演示了 DatePicker 和 TimePicker 控件的用法。

程序主界面设置了两个按钮。点击第一个按钮，跳转到 DatePickerActivity，显示 DatePicker 控件；点击第二个按钮，跳转到 TimePickerActivity，显示 TimePicker 控件。

DatePickerActivity 类的主要代码如代码段 5-8 所示。

**代码段 5-8　DatePickerActivity 类的主要代码**

```
//package 和 import 语句略
public class DatePickerActivity extends Activity {
    private DatePicker myDatePicker;
    private TextView textDate;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.date);
        textDate = (TextView)findViewById(R.id.textDate);
        myDatePicker = (DatePicker)findViewById(R.id.datePicker);
        Calendar calendar =Calendar.getInstance(Locale.CHINA);
        int year =calendar.get(Calendar.YEAR);
        int monthOfYear =calendar.get(Calendar.MONTH);
        int dayOfMonth =calendar.get(Calendar.DAY_OF_MONTH);
        myDatePicker.init(year, monthOfYear, dayOfMonth, new
            OnDateChangedListener() {
            @Override
            public void onDateChanged(DatePicker view, int year, int monthOfYear,
```