Android程序设计

对话框和菜单

2019.5.12

isszym　sysu.edu.cn

官方文档（中文） 官方文档（英文） runoob

目录

- 对话框(DialogBox)
- 工具栏(ActionBar)
- 菜单(Menu)
- 桌面管理器(WindowManager)
- 桌面图标(AppWidgets)
- Toast和Notification

对话框

- 对话框概述
- AlertDialog
  简单对话框
  简单列表对话框
  单项选择对话框
  多项选择对话框
  自定义列表项对话框
  自定义View对话框
- PopupWindow
- ProgressDialog
- DatePickerDialog

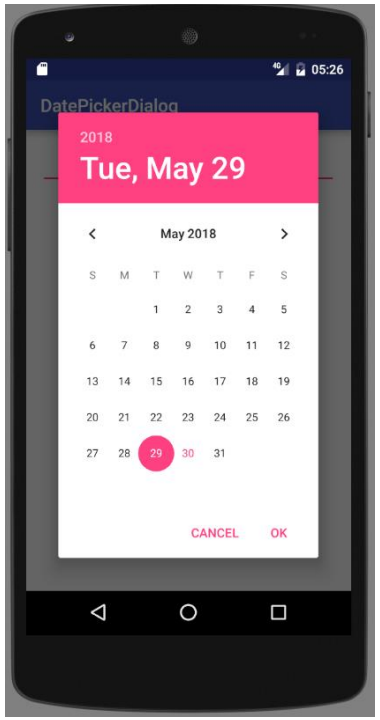
对话框概述

对话框是在当前Activity上临时弹出一个窗口，在点击其中的项目或按钮之后会自动关闭。采用对话框的一个好处是不用增加Activity就可以完成消息显示、单项选择等功能。本节给出了以下对话框：

AlertDialog
确认信息、单项选择或多项选择

PopupWindow
弹出一个消息窗口

ProgressDialog
显示进度



DatePickerDialog
选择日期

AlertDialog对话框

- AlertDialog对话框主要有下面几种

(1) 简单对话框　(2) 简单列表对话框　(3) 单项选择对话框　(4) 多项选择对话框

- 设计AlertDialog对话框

（1）通过AlertDialog类的Builder方法在当前上下文环境下创建AlertDialog对话框，对该对话框设置图标（Icon）、标题（Title）、信息（Message）、列表项（Items）、按钮（Buttons）之后把它显示出来。

（2）列表项可以是普通列表项（Items）、单选项（SingleChoiceItems）和多选项（MultiChoiceItems)。按钮可以是取消按钮（NegativeButton）、确定按钮（PositiveButton）和中立按钮（NeutralButton）。

（3）列表项、单选项和按钮项都可以设置onClick事件，被点击后会发生Click事件并自动退出对话框。

```
dialogBuilder = new AlertDialog.Builder(mContext);
alertDialog = dialogBuilder.setIcon(R.mipmap.sysu)
        .setTitle("简单对话框：")
        .setMessage("这是一个普通AlertDialog,\n可以加入三个按钮：取消，中立和确定")
        .setNegativeButton("取消", new DialogInterface.OnClickListener() {
            @Override
            public void onClick(DialogInterface dialog, int which) {
                Toast.makeText(mContext, "你点击了取消按钮", Toast.LENGTH_SHORT).show();
            }
        }).create();            // 创建AlertDialog对象
alertDialog.show();             // 显示对话框
```

- 例子：AlertDialog

MainActivity.java

```
public class MainActivity extends AppCompatActivity {
    TextView show;
    String[] items = new String[]{
            "数据科学与计算机学院", "数学学院", "物理学院", "化学学院", "生命科学学院"};
    private Context mContext;
    private boolean[] checkItems;
    private AlertDialog alertDialog = null;
    private AlertDialog.Builder dialogBuilder = null;
    private int pos = 0;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        show = (TextView) findViewById(R.id.show);
        mContext = this;
    }
    public void simple(View source) { ... }
    public void simpleList(View source) { ... }
    public void singleChoice(View source) { ... }
    public void multiChoice(View source) { ... }
    public void customList(View source) { ... }
    public void customView(View source) { ... }
}
```

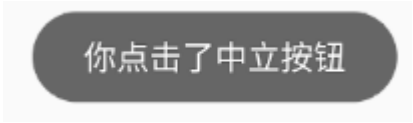
activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout 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"
    android:paddingBottom="@dimen/activity_vertical_margin"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    tools:context="com.example.isszym.alertdialog.MainActivity">

    <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
        android:orientation="vertical"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:gravity="center_horizontal">
        <!-- 显示一个普通的文本编辑框组件 -->
        <EditText
            android:id="@+id/show"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:editable="false"/>
```

```
<!-- 定义一个普通的按钮组件 -->
<Button
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:text="简单对话框"
    android:onClick="simple"
    />
<!-- 定义一个普通的按钮组件 -->
<Button
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:text="简单列表项对话框"
    android:onClick="simpleList"
    />
<!-- 定义一个普通的按钮组件 -->
<Button
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:text="单选列表项对话框"
    android:onClick="singleChoice"
    />
<!-- 定义一个普通的按钮组件 -->
<Button
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:text="多选列表项对话框"
    android:onClick="multiChoice"
    />
<!-- 定义一个普通的按钮组件 -->
<Button
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:text="自定义列表项对话框"
    android:onClick="customList"
    />
<!-- 定义一个普通的按钮组件 -->
<Button
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:text="自定义View对话框"
    android:onClick="customView"
    />
</LinearLayout>
</RelativeLayout>
```

简单对话框


02:34
AlertDialog
简单对话框
简单对话框：
这是一个普通AlertDialog,
可以加入三个按钮：取消，中立和确定
中立
取消
确定
自定义VIEW对话框
点击
你点击了中立按钮


你点击了中立按钮


你点击了确定按钮


你点击了取消按钮

```
public void simple(View source) {
    dialogBuilder = new AlertDialog.Builder(mContext);
    alertDialog = dialogBuilder.setIcon(R.mipmap.sysu)
            .setTitle("简单对话框：")
            .setMessage("这是一个普通AlertDialog,\n可以加入三个按钮：取消，中立和确定")
            .setNegativeButton("取消", new DialogInterface.OnClickListener() {
                @Override
                public void onClick(DialogInterface dialog, int which) {
                    Toast.makeText(mContext, "你点击了取消按钮", Toast.LENGTH_SHORT).show();
                }
            })
            .setPositiveButton("确定", new DialogInterface.OnClickListener() {
                @Override
                public void onClick(DialogInterface dialog, int which) {
                    Toast.makeText(mContext, "你点击了确定按钮", Toast.LENGTH_SHORT).show();
                }
            })
            .setNeutralButton("中立", new DialogInterface.OnClickListener() {
                @Override
                public void onClick(DialogInterface dialog, int which) {
                    Toast.makeText(mContext, "你点击了中立按钮", Toast.LENGTH_SHORT).show();
                }
            }).create();              // 创建AlertDialog对象
    alertDialog.show();               // 显示对话框
}
```

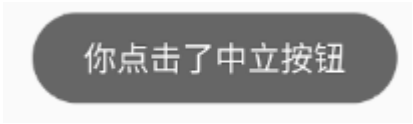
简单列表对话框

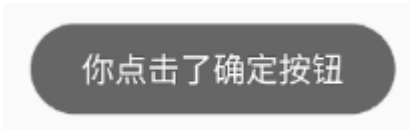

02:43
AlertDialog
简单列表对话框
数据科学与计算机学院
数学学院
物理学院
化学学院
生命科学学院
中立
取消
确定
选中了"数据科学与计算机学院"
点击列表
你点击了中立按钮
你点击了确定按钮
你点击了取消按钮
选中了"数据科学与计算机学院"

```
public void simpleList(View source) {
    dialogBuilder = new AlertDialog.Builder(mContext);
    alertDialog = dialogBuilder
            .setIcon(R.mipmap.sysu)
            .setTitle("简单列表对话框")
            .setItems(items, new OnClickListener() {
                @Override
                public void onClick(DialogInterface dialog, int which) {
                    Toast.makeText(mContext, "选中了\"" + items[which] + "\"",
                                                                Toast.LENGTH_SHORT).show();
                }
            })
            .setNegativeButton("取消", new DialogInterface.OnClickListener() {
                @Override
                public void onClick(DialogInterface dialog, int which) {
                    Toast.makeText(mContext, "点击了取消按钮", Toast.LENGTH_SHORT).show();
                }
            })
            .setPositiveButton("确定", new DialogInterface.OnClickListener() {
                public void onClick(DialogInterface dialog, int which) {
                    Toast.makeText(mContext, "点击了确定按钮", Toast.LENGTH_SHORT).show();
                }
            })
            .setNeutralButton("中立", new DialogInterface.OnClickListener() {
                public void onClick(DialogInterface dialog, int which) {
                    Toast.makeText(mContext, "点击了中立按钮", Toast.LENGTH_SHORT).show();
                }
            }).create();              // 创建AlertDialog对象
    alertDialog.show();
}
```

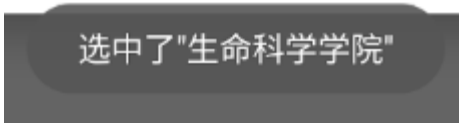
单项选择对话框


03:12
AlertDialog
单项选择对话框
数据科学与计算机学院
数学学院
物理学院
化学学院
生命科学学院
中立
取消
确定
选中了"生命科学学院"


你点击了中立按钮


你点击了确定按钮


你点击了取消按钮


选中了"生命科学学院"

```
public void singleChoice(View source) {
    dialogBuilder = new AlertDialog.Builder(mContext);
    alertDialog = dialogBuilder
            .setIcon(R.mipmap.sysu)
            .setTitle("单项选择对话框")
            .setSingleChoiceItems(items, 1, new OnClickListener() {// 默认选中第二项（索引为1）
                @Override
                public void onClick(DialogInterface dialog, int which) {
                    Toast.makeText(mContext, "选中了\"" + items[which] + "\"",
                                   Toast.LENGTH_SHORT).show();
                }
            })
            .setNegativeButton("取消", new DialogInterface.OnClickListener() {
                public void onClick(DialogInterface dialog, int which) {
                    Toast.makeText(mContext, "点击了取消按钮", Toast.LENGTH_SHORT).show();
                }
            })
            .setPositiveButton("确定", new DialogInterface.OnClickListener() {
                public void onClick(DialogInterface dialog, int which) {
                    Toast.makeText(mContext, "点击了确定按钮", Toast.LENGTH_SHORT).show();
                }
            })
            .setNeutralButton("中立", new DialogInterface.OnClickListener() {
                public void onClick(DialogInterface dialog, int which) {
                    Toast.makeText(mContext, "点击了中立按钮", Toast.LENGTH_SHORT).show();
                }
            }).create();              // 创建AlertDialog对象
    alertDialog.show();
}
```

多项选择对话框


03:17
AlertDialog
多项选择对话框
数据科学与计算机学院
数学学院
物理学院
化学学院
生命科学学院
中立
取消
确定
你点击了中立按钮
你点击了取消按钮
点击了确定按钮,选中了3个项目

```
public void multiChoice(View source) {
    dialogBuilder = new AlertDialog.Builder(mContext);
    alertDialog = dialogBuilder
            .setIcon(R.mipmap.sysu)
            .setTitle("多项选择对话框")
            // 设置单选列表项，默认选中第二项和第四项（索引为1和3）
            .setMultiChoiceItems(items, new boolean[]{false, true, false, true, false}, null)
            .setNegativeButton("取消", new DialogInterface.OnClickListener() {
                public void onClick(DialogInterface dialog, int which) {
                    Toast.makeText(mContext, "点击了取消按钮", Toast.LENGTH_SHORT).show();
                }
            })
            .setPositiveButton("确定", new DialogInterface.OnClickListener() {
                public void onClick(DialogInterface dialog, int which) {
                    ListView list = alertDialog.getListView();
                    Toast.makeText(mContext, "点击了确定按钮,选中了"
                            + list.getCheckedItemCount() + "个项目", Toast.LENGTH_SHORT).show();

                }
            })
            .setNeutralButton("中立", new DialogInterface.OnClickListener() {
                public void onClick(DialogInterface dialog, int which) {
                    Toast.makeText(mContext, "点击了中立按钮", Toast.LENGTH_SHORT).show();
                }
            }).create();              // 创建AlertDialog对象
    alertDialog.show();
}
```

自定义列表项对话框


04:07
AlertDialog
简单对话框
自定义列表项对话框
数据科学与计算机学院
数学学院
物理学院
化学学院
生命科学学院
点击
点击
点击了第0个项目:数据科学与计算机学院
点击了第4个项目:生命科学学院

```
public void customList(View source) {
    dialogBuilder = new AlertDialog.Builder(mContext);
    alertDialog = dialogBuilder
            .setIcon(R.mipmap.sysu)
            .setTitle("自定义列表项对话框")
            .setAdapter(
                    new ArrayAdapter<String>(mContext,
                            R.layout.array_item, items),
                    new OnClickListener() {
                        @Override
                        public void onClick(DialogInterface dialog, int which) {
                            pos = which;
                            ListView list = alertDialog.getListView();
                            Toast.makeText(mContext, "点击了第" + which + "个项目: "
                                    + list.getItemAtPosition(pos), Toast.LENGTH_SHORT).show();
                        }
                    }
            )
            .create();              // 创建AlertDialog对象
    alertDialog.show();
}
```

array_item.xml

```
<?xml version="1.0" encoding="utf-8"?>
<TextView xmlns:android="http://schemas.android.com/apk/res/android"
    android:id="@+id/TextView"
    android:textColor="#f00f"       android:textSize="20dp"
    android:shadowColor="#777"      android:shadowRadius="2"
    android:shadowDx="5"            android:shadowDy="5"
   android:paddingLeft="12dp"
   android:layout_width="match_parent"
   android:layout_height="wrap_content" />
```

自定义View对话框


04:12
AlertDialog
自定义View对话框
学号: 1500001
姓名: David
手机号码: 13600000001
中立
取消
确定
自定义列表项对话框


点击了确定按钮,学号为"1500001"

```
public void customView(View source) {
    TableLayout studentForm = (TableLayout) getLayoutInflater()
            .inflate(R.layout.student, null);
    dialogBuilder = new AlertDialog.Builder(mContext);
    alertDialog = dialogBuilder
            .setIcon(R.mipmap.sysu)
            .setTitle("自定义View对话框")
            .setView(studentForm)
            .setNegativeButton("取消", new DialogInterface.OnClickListener() {
                @Override
                public void onClick(DialogInterface dialog, int which) {
                    Toast.makeText(mContext, "点击了取消按钮", Toast.LENGTH_SHORT).show();
                }
            })
            .setPositiveButton("确定", new DialogInterface.OnClickListener() {
                @Override
                public void onClick(DialogInterface dialog, int which) {
                    EditText stuNum=(EditText)alertDialog.findViewById(R.id.stu_num);
                    Toast.makeText(mContext, "点击了确定按钮,学号为\""
                            + stuNum.getText() + "\"", Toast.LENGTH_SHORT).show();
                }
            })
            .setNeutralButton("中立", new DialogInterface.OnClickListener() {
                @Override
                public void onClick(DialogInterface dialog, int which) {
                    Toast.makeText(mContext, "点击了中立按钮", Toast.LENGTH_SHORT).show();
                }
            })
            .create();              // 创建AlertDialog对象
    alertDialog.show();
}
```

studentForm.xml

```
<?xml version="1.0" encoding="utf-8"?>
<TableLayout xmlns:android="http://schemas.android.com/apk/res/android"
          android:id="@+id/studentForm"
          android:layout_width="match_parent"
          android:layout_height="match_parent"
    android:padding="12dp">
    <TableRow>
        <TextView
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:text="学号:"
            android:textSize="20dp"/>
        <EditText
            android:id="@+id/stu_num"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:hint="请填写学号"
            android:textSize="20dp"
            android:selectAllOnFocus="true"/>
    </TableRow>
    <TableRow>
        <TextView
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:text="姓名:"
            android:textSize="20dp"/>
        <EditText
            android:id="@+id/stu_name"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:textSize="20dp"
            android:hint="请填写姓名"/>
    </TableRow>
    <TableRow>
        <TextView
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:text="手机号码:"
            android:textSize="20dp"/>
        <EditText
            android:id="@+id/phone_num"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:hint="请填写手机号码"
            android:textSize="20dp"
            android:inputType="phone"
            android:selectAllOnFocus="true" />
    </TableRow>
</TableLayout>
```

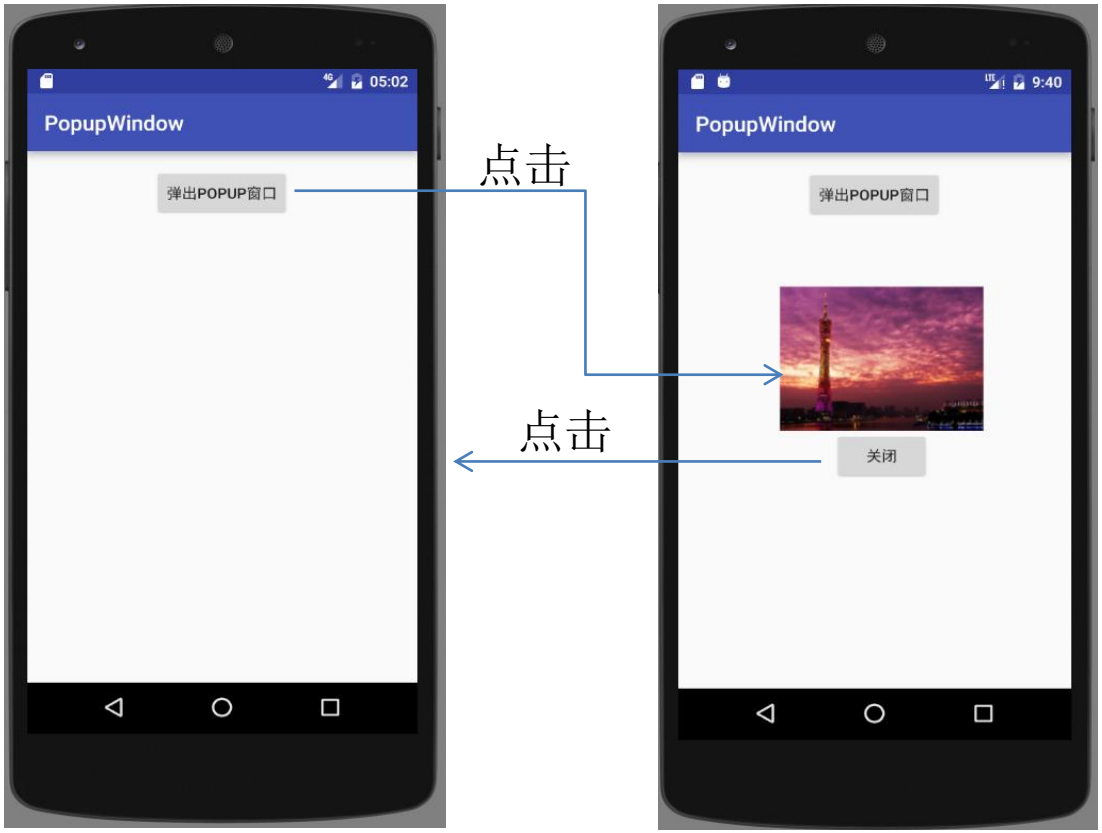
AndroidManifest.xml

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.isszym.alertdialog">

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:supportsRtl="true"
        android:theme="@style/AppTheme">
        <activity android:name=".MainActivity">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>

</manifest>
```

PopupWindow

先利用PopupWindow得到一个包含某个布局的弹出式窗口(popup)，然后利用showAtLocation把它定位显示出来，最后用popup.dismiss()关闭弹出式窗口。

```
View root = getLayoutInflater().inflate(R.layout.popup, null);
final PopupWindow popup = new PopupWindow(root, 560, 720); //view,width,height
popup.showAtLocation(findViewById(R.id.bn),Gravity.CENTER, 20,20); //gravity,offsetX,offsetY
```



```
public class MainActivity extends AppCompatActivity {
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        // 装载R.layout.popup对应的界面布局
        View root = getLayoutInflater().inflate(R.layout.popup, null);
        final PopupWindow popup = new PopupWindow(root, 560, 720);//,width,height
        Button button = (Button) findViewById(R.id.bn);
        button.setOnClickListener(new OnClickListener() {
            @Override
            public void onClick(View v){
                //将PopupWindow显示在指定位置： : parent,gravity, offsetX, offsetY
                //以下拉方式显示: popup.showAsDropDown(v);
                popup.showAtLocation(findViewById(R.id.bn),
                        Gravity.CENTER, 20,20);//gravity,offsetX,offsetY
            }
        });
        root.findViewById(R.id.close).setOnClickListener(
                new OnClickListener() {
                    public void onClick(View v) {
                        popup.dismiss(); // ①  关闭PopupWindow
                    }
                });
    }
}
```

activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout 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"
    android:paddingBottom="@dimen/activity_vertical_margin"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    tools:context="com.example.isszym.popupwindow.MainActivity">

    <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
        android:orientation="vertical"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:gravity="center_horizontal">
        <Button android:id="@+id/bn"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="弹出Popup窗口"  />
    </LinearLayout>
</RelativeLayout>
```

popup.xml

```
<?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"
   android:gravity="center_horizontal">
<ImageView
   android:layout_width="240dp"
   android:layout_height="wrap_content"
   android:src="@drawable/gzt"
   />
<Button
   android:id="@+id/close"
   android:layout_width="wrap_content"
   android:layout_height="wrap_content"
   android:text="关闭"
   />
</LinearLayout>
```

ProgressDialog

显示水平进度条：

```
pd = new ProgressDialog(MainActivity.this);
pd.setMax(MAX_PROGRESS);
pd.setTitle("任务完成百分比");              // 设置标题
pd.setMessage("耗时任务的完成百分比"); // 设置提示信息
pd.setCancelable(false);                    // 设置是否能用“取消”按钮关闭
pd.setProgressStyle(ProgressDialog.STYLE_HORIZONTAL);// 使用水平进度条
pd.setIndeterminate(false);                 // 设置是否显示进度
pd.show();
```

设置进度（百分比）：

```
pd.setProgress(progress);
```

使用环形进度条（不能设置进度）

```
pd.setProgressStyle(ProgressDialog.STYLE_SPINNER);
```

组合设置方法：

```
ProgressDialog.show(this, "任务执行中", "任务执行中，请等待", false, true);
```

参数：context,title,message,indeterminate,cancelable

项目名： **ProgressDialog**

```
public class MainActivity extends AppCompatActivity {
    final static int MAX_PROGRESS = 100;  // 该程序模拟填充长度为100的数组
    private int[] data = new int[50];
    int progressStatus = 0;  // 记录进度对话框的完成百分比
    int hasData = 0;
    ProgressDialog pd1,pd2;
    Handler handler = new Handler(){
        @Override
        public void handleMessage(Message msg) {  // 定义一个负责更新的进度的Handler
            if (msg.what == 0x123){  // 表明消息是由该程序发送的
                pd2.setProgress(progressStatus);
            }
        }
    };
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
    }
    public void showSpinner(View source) {          // 调用静态方法显示环形进度条
        ProgressDialog.show(this, "任务执行中"
                , "任务执行中，请等待", false, true); // ①
    }
```

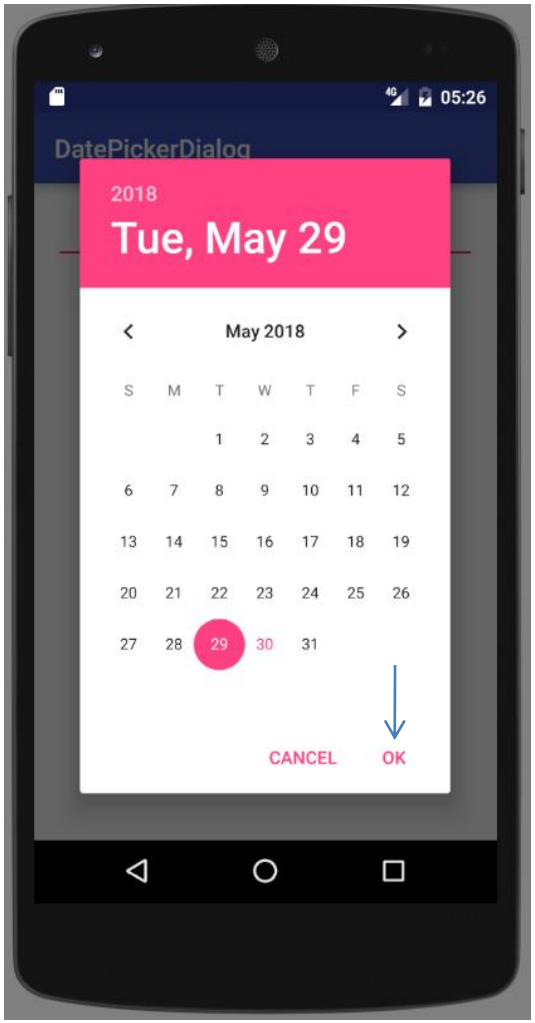
```
public void showIndeterminate(View source) {
    pd1 = new ProgressDialog(MainActivity.this);
    pd1.setTitle("任务正在执行中");  // 设置对话框的标题
    pd1.setMessage("任务正在执行中，敬请等待...");// 设置显示内容
    pd1.setCancelable(true);  // 设置对话框能用“取消”按钮关闭
    pd1.setProgressStyle(ProgressDialog.STYLE_HORIZONTAL);  // 设置进度条风格
    pd1.setIndeterminate(true);   // 设置对话框的进度条是否显示进度
    pd1.show(); // ②
}
public void showProgress(View source) {
    progressStatus = 0;   // 将进度条的完成进度重设为0
    hasData = 0;   // 重新开始填充数组
    pd2 = new ProgressDialog(MainActivity.this);
    pd2.setMax(MAX_PROGRESS);
    pd2.setTitle("任务完成百分比");   // 设置对话框的标题
    pd2.setMessage("耗时任务的完成百分比");  // 设置对话框显示的内容
    pd2.setCancelable(false);   // 设置对话框不能用“取消”按钮关闭
    pd2.setProgressStyle(ProgressDialog.STYLE_HORIZONTAL);
    pd2.setIndeterminate(false);          // 设置对话框的进度条是否显示进度
    pd2.show(); // ③
```

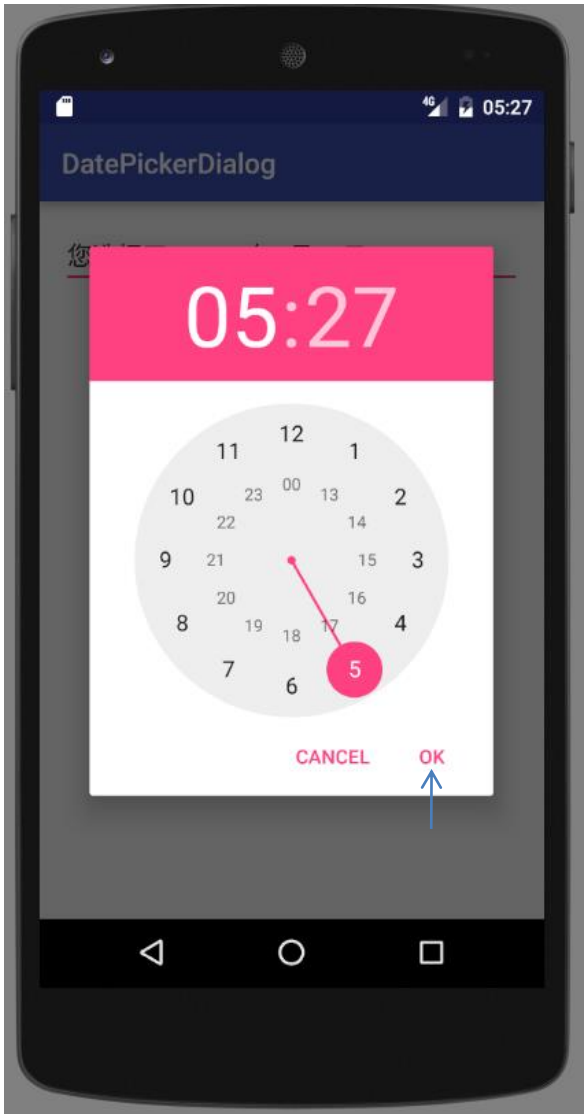
```
    new Thread(){
            public void run() {
                while (progressStatus < MAX_PROGRESS) {
                    progressStatus = MAX_PROGRESS
                            * doWork() / data.length; // 获取耗时操作的完成百分比
                    handler.sendEmptyMessage(0x123); // 发送空消息到Handler
                }
                if (progressStatus >= MAX_PROGRESS) {  // 如果任务已经完成
                    pd2.dismiss();   // 关闭对话框
                }
            }
        }.start();
    }
    // 模拟一个耗时的操作
    public int doWork() {
        data[hasData++] = (int) (Math.random() * 100);
        try {
            Thread.sleep(100);
        }
        catch (InterruptedException e) {
            e.printStackTrace();
        }
        return hasData;
    }
}
```

activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout 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"
    android:paddingBottom="@dimen/activity_vertical_margin"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    tools:context="com.example.isszym.progressdialog.MainActivity">
    <LinearLayout
        xmlns:android="http://schemas.android.com/apk/res/android"
        android:orientation="vertical"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:gravity="center_horizontal">
        <Button
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:text="环形进度条"
            android:onClick="showSpinner" />
         <Button
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:text="不显示进度的进度条"
            android:onClick="showIndeterminate"/>
        <Button
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:text="显示进度的进度条"
            android:onClick="showProgress" />
    </LinearLayout>
</RelativeLayout>
```

DatePickerDialog




05:27
DatePickerDialog
05:27
12
1
2
3
4
5
6
7
8
9
10
11
00
13
14
15
16
17
18
19
20
21
22
23
CANCEL
OK


05:28
DatePickerDialog
您选择了：5时27分
设置日期
设置时间

调用DatePickerDialog实例的show方法会在当前界面弹出日期对话框，其构造器的第二个参数为日期监听器实例，选择日期时会调用其回调函数onDateSet。

```
DatePickerDialog dp = new DatePickerDialog(context,
    new DatePickerDialog.OnDateSetListener(){
        @Override
        public void onDateSet(DatePicker dp, int year,int month,int dayOfMonth){
            ...
        }
    },
    year, month, day);  // int类型
dp.show();
```

调用TimePickerDialog实例的show方法会在当前界面弹出时间对话框，其构造器的第二个参数为时间监听器实例，选择时间时会调用其回调函数onTimeSet。

```
TimePickerDialog tp =  new TimePickerDialog(this,
        new TimePickerDialog.OnTimeSetListener(){ // 绑定监听器
                @Override
                void onTimeSet(TimePicker tp,int hourOfDay,int minute) { ... }
         }, hour,minute,true); //true表示采用24小时制
tp.show();
```

项目名：DatePickerDialog

```
public class MainActivity extends AppCompatActivity {
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        Button dateBn = (Button)findViewById(R.id.dateBn);
        Button timeBn = (Button)findViewById(R.id.timeBn);
        dateBn.setOnClickListener(new OnClickListener() {
            @Override
            public void onClick(View source)  {
                Calendar c = Calendar.getInstance();
                new DatePickerDialog(MainActivity.this, // 建对话（继承类）
                        new DatePickerDialog.OnDateSetListener(){// 继承监听器(接口)
                            @Override
                            public void onDateSet(DatePicker dp, int year,
                                                  int month, int dayOfMonth) {
                                EditText show = (EditText) findViewById(R.id.show);
                                show.setText("您选择了：" + year + "年" + (month + 1)
                                        + "月" + dayOfMonth + "日");
                            }
                        },
                        c.get(Calendar.YEAR),c.get(Calendar.MONTH),
                        c.get(Calendar.DAY_OF_MONTH)).show();
            }
        });
```

```
        //为“设置时间”按钮绑定监听器
        timeBn.setOnClickListener(new OnClickListener()  {
            @Override
            public void onClick(View source) {
                Calendar c = Calendar.getInstance();
                // 创建一个TimePickerDialog实例，并把它显示出来
                new TimePickerDialog(MainActivity.this,
                        new TimePickerDialog.OnTimeSetListener(){ // 绑定监听器
                            @Override
                            public void onTimeSet(TimePicker tp, int hourOfDay,
                                                  int minute) {
                                EditText show = (EditText) findViewById(R.id.show);
                                show.setText("您选择了：" + hourOfDay + "时"
                                        + minute + "分");
                            }
                        }
                        , c.get(Calendar.HOUR_OF_DAY) //设置初始时间
                        , c.get(Calendar.MINUTE)
                        , true).show(); //true表示采用24小时制
            }
        });
    }
```

activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout 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"
    android:paddingBottom="@dimen/activity_vertical_margin"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    tools:context="com.example.isszym.datepickerdialog.MainActivity">
    <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
        android:orientation="vertical"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:gravity="center_horizontal">
        <EditText
            android:id="@+id/show"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:editable="false"
            />
        <LinearLayout
            android:orientation="horizontal"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:gravity="center"
            >
```

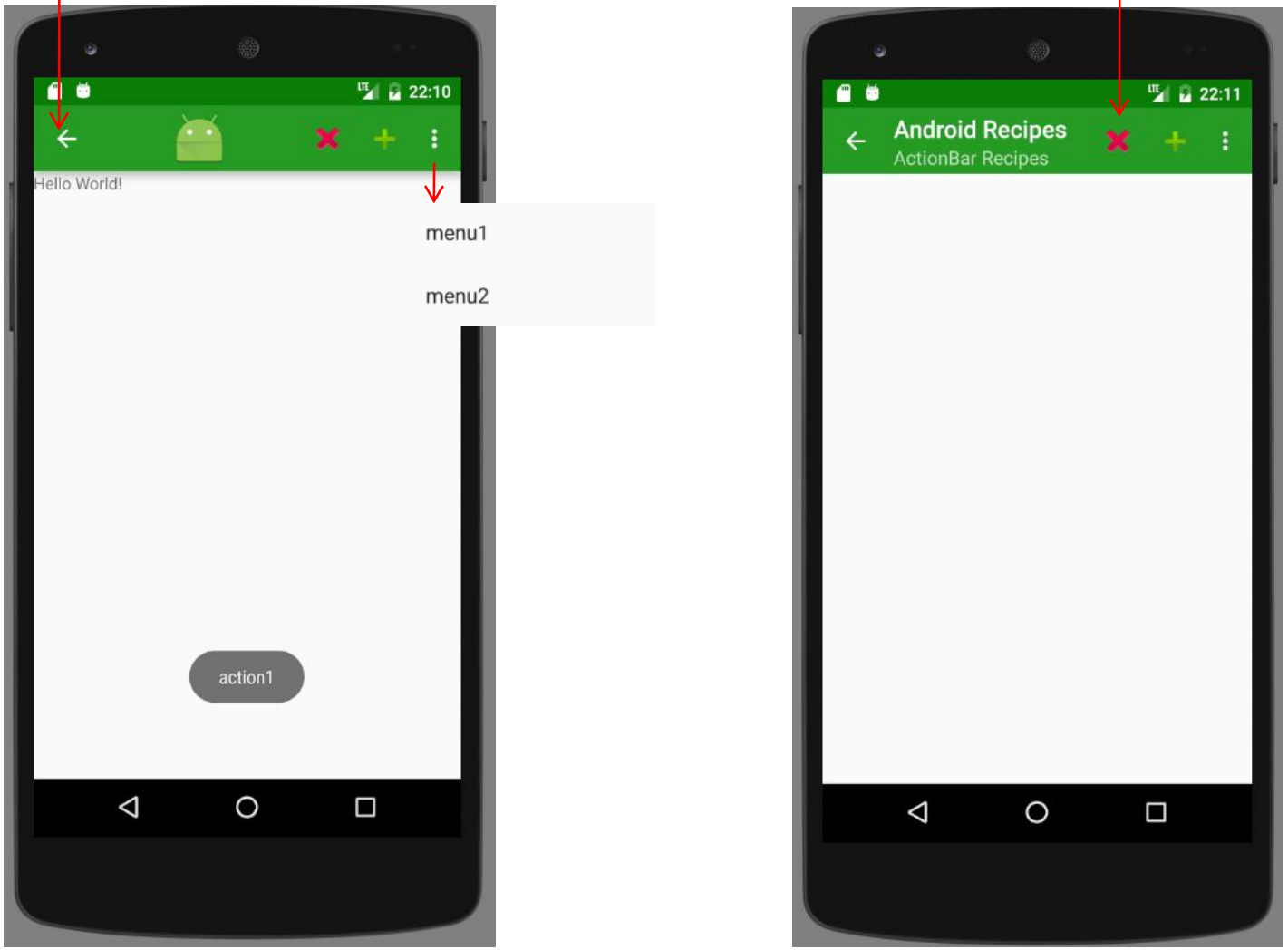
```
            <Button
                android:id="@+id/dateBn"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:text="设置日期"
                />
             <Button
                android:id="@+id/timeBn"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:text="设置时间"
                />
        </LinearLayout>
    </LinearLayout>
</RelativeLayout>
```

工具栏

参考

点击进入上一个Activity

关闭当前Activity，进入上一个Activity



在Activity的回调函数中扩展Action的xml文件，就可以显示出ActionBar，xml文件见下页：

```
public class SupportActionActivity extends AppCompatActivity {
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
    }

    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        getMenuInflater().inflate(R.menu.support, menu);
        return true;
    }
}
```

如果要显示回退键，需要执行以下语句：

```
ActionBar actionBar = getSupportActionBar();
actionBar.setDisplayHomeAsUpEnabled(true);  //显示回退按钮
```

support.xml

```
<?xml version="1.0" encoding="utf-8"?>
<menu xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto">
    <item android:id="@+id/action1"
        android:title="action1"
        android:icon="@android:drawable/ic_delete"
        app:showAsAction="ifRoom" />
    <item android:id="@+id/action2"
        android:title="action2"
        android:icon="@android:drawable/ic_input_add"
        app:showAsAction="ifRoom" />
    <item android:id="@+id/menu1"
        android:title="menu1"
        android:orderInCategory="100"
        app:showAsAction="never"/>
    <item android:id="@+id/menu2"
        android:title="menu2"
        android:orderInCategory="110"
        app:showAsAction="never"/>
</menu>
```

showAsAction取值(可以混合使用):

ifRoom 有空间则显示
never 只在溢出列表中显示标题
always 无论是否溢出，总会显示。
withText 尽可能显示标题，但是图标优先
collapseActionView 折叠到一个按钮。 一般要配合ifRoom一起使用才会有效果。

showAsAction属性共有五个值：ifRoom,never,always,withText,collapseActionView，可以混合使用。

- ifRoom 会显示在Item中，但是如果已经有4个或者4个以上的Item时会隐藏在溢出列表中。当然个数并不仅仅局限于4个，依据屏幕的宽窄而定
- never 永远不会显示。只会在溢出列表中显示，而且只显示标题，所以在定义item的时候，最好把标题都带上。
- always 无论是否溢出，总会显示。
- withText值示意Action bar要显示文本标题。Action bar会尽可能的显示这个标题，但是，如果图标有效并且受到Action bar空间的限制，文本标题有可能显示不全。
- collapseActionView 声明了这个操作视窗应该被折叠到一个按钮中，当用户选择这个按钮时，这个操作视窗展开。否则，这个操作视窗在默认的情况下是可见的，并且即便在用于不适用的时候，也要占据操作栏的有效空间。 一般要配合ifRoom一起使用才会有效果。

项目名：**ActionBar**

```
public class SupportActionActivity extends AppCompatActivity {
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        ActionBar actionBar = getSupportActionBar();
        actionBar.setDisplayHomeAsUpEnabled(true);  //显示回退按钮
        actionBar.setDisplayShowCustomEnabled(true);//允许在title栏显示自定义View
        ImageView imageView = new ImageView(this);
        imageView.setImageResource(R.mipmap.ic_launcher);
        imageView.setScaleType(ImageView.ScaleType.CENTER);
        LayoutParams lp = new LayoutParams(
                LayoutParams.MATCH_PARENT,
                LayoutParams.MATCH_PARENT);
        actionBar.setCustomView(imageView, lp);
    }

    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        getMenuInflater().inflate(R.menu.support, menu);
        return true;
    }
```

```
    @Override
    public boolean onOptionsItemSelected(MenuItem item) {
        switch (item.getItemId()) {
            case android.R.id.home:
                Intent intent =
                          new Intent(this,SupportToolbarActivity.class);
                startActivity(intent);
                break;
            case R.id.action1:
                Toast.makeText(this, "action1", Toast.LENGTH_SHORT).show();
                break;
            case R.id.action2:
                Toast.makeText(this, "action2", Toast.LENGTH_SHORT).show();
                break;
            case R.id.menu1:
                Toast.makeText(this, "menu1", Toast.LENGTH_SHORT).show();
                break;
            case R.id.menu2:
                Toast.makeText(this, "menu2", Toast.LENGTH_SHORT).show();
                break;
            default:
                return super.onOptionsItemSelected(item);
        }
        return true;
    }
}
```

```java
public class SupportToolbarActivity extends AppCompatActivity {
    private Toolbar mToolbar;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_toolbar);

        //We have to tell the activity where the toolbar is
        mToolbar = (Toolbar) findViewById(R.id.toolbar);
        setSupportActionBar(mToolbar);
        ActionBar actionBar = getSupportActionBar();
        //Display home with the "up" arrow indicator
        actionBar.setDisplayHomeAsUpEnabled(true);
        actionBar.setTitle("Android Recipes");
        actionBar.setSubtitle("ActionBar Recipes");
    }

    @Override
    protected void onPostCreate(Bundle savedInstanceState) {
        super.onPostCreate(savedInstanceState);
    }
```

```
    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        getMenuInflater().inflate(R.menu.support, menu);
        return true;
    }
    @Override
    public boolean onOptionsItemSelected(MenuItem item) {
        switch (item.getItemId()) {
            case android.R.id.home:
              Toast.makeText(this, "home",Toast.LENGTH_SHORT).show(); break;
            case R.id.action1:
              Intent intent =
                       new Intent(this,SupportActionActivity.class);
              startActivity(intent);                    break;
            case R.id.action2:
               Toast.makeText(this,"action2",Toast.LENGTH_SHORT).show();break;
            case R.id.menu1:
               Toast.makeText(this,"menu1",Toast.LENGTH_SHORT).show(); break;
            case R.id.menu2:
               Toast.makeText(this,"menu2",Toast.LENGTH_SHORT).show(); break;
            default:
               return super.onOptionsItemSelected(item);
        }
        return true;
    }
}
```

activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<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"  >

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Hello World!" />
</LinearLayout>
```

activity_toolbar.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    android:orientation="vertical"
    android:layout_width="match_parent"
    android:layout_height="match_parent">
    <android.support.v7.widget.Toolbar
        android:id="@+id/toolbar"
        android:layout_height="wrap_content"
        android:layout_width="match_parent"
        android:minHeight="?attr/actionBarSize"
        android:background="?attr/colorPrimary"
        app:theme="@style/ThemeOverlay.AppCompat.Dark.ActionBar"/>

    <!-- Remaining application view contents here -->

</LinearLayout>
```

support.xml

```
<?xml version="1.0" encoding="utf-8"?>
<menu xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto">
    <item android:id="@+id/action1"
        android:title="action1"
        android:icon="@android:drawable/ic_delete"
        app:showAsAction="ifRoom" />
    <item android:id="@+id/action2"
        android:title="action2"
        android:icon="@android:drawable/ic_input_add"
        app:showAsAction="ifRoom" />
    <item android:id="@+id/menu1"
        android:title="menu1"
        android:orderInCategory="100"
        app:showAsAction="never"/>
    <item android:id="@+id/menu2"
        android:title="menu2"
        android:orderInCategory="110"
        app:showAsAction="never"/>
</menu>
```

showAsAction取值(可以混合使用):

ifRoom 有空间则显示
never 只在溢出列表中显示标题
always 无论是否溢出，总会显示。
withText 尽可能显示标题，但是图标优先
collapseActionView 折叠到一个按钮。 一般要配合ifRoom一起使用才会有效果。

res\drawable\actionbar_background.xml

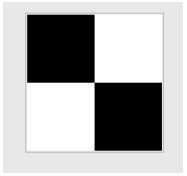
```
<?xml version="1.0" encoding="utf-8"?>
<bitmap xmlns:android="http://schemas.android.com/apk/res/android"
    android:src="@drawable/checkers"
    android:tileMode="repeat" />
```

res\drawable\actionbar_item_background.xml

```
<?xml version="1.0" encoding="utf-8"?>
<selector xmlns:android="http://schemas.android.com/apk/res/android" >
    <item android:state_pressed="true" android:state_enabled="true"
        android:drawable="@drawable/actionbar_item_background_pressed" />
    <item android:drawable="@android:color/transparent" />
</selector>
```

res\drawable\actionbar_item_background_pressed.xml

```
<?xml version="1.0" encoding="utf-8"?>
<shape xmlns:android="http://schemas.android.com/apk/res/android"
    android:shape="rectangle">
    <solid  android:color="#7CCC" />
</shape>
```



Drawable/checkers.png

res\drawable\divider.xml

```
<?xml version="1.0" encoding="utf-8"?>
<shape xmlns:android="http://schemas.android.com/apk/res/android"
    android:shape="rectangle">
    <solid  android:color="#00A"/>
    <size  android:width="1dp" android:height="1dp" />
</shape>
```

res\values\styles.xml

```
<?xml version="1.0" encoding="utf-8"?>
<resources
    >
    <!-- Defines a "theme" that will apply to the entire application,
         or at least a handful of its activities -->
    <style name="AppTheme" parent="@style/Theme.AppCompat.Light.DarkActionBar">
        <!-- Provide decor theme colors -->
        <item name="colorPrimary">@color/primaryGreen</item>
        <item name="colorPrimaryDark">@color/darkGreen</item>
        <item name="colorAccent">@color/accentGreen</item>

    </style>
    <!-- The toolbar will replace the standard action bar, so we
         need a theme that removes the action bar from the window decor -->
    <style name="AppToolbarTheme" parent="@style/Theme.AppCompat.Light.NoActionBar">
        <!-- Provide decor theme colors -->
        <item name="colorPrimary">@color/primaryGreen</item>
        <item name="colorPrimaryDark">@color/darkGreen</item>
        <item name="colorAccent">@color/accentGreen</item>

    </style>
</resources>
```

AndroidManifest.xml

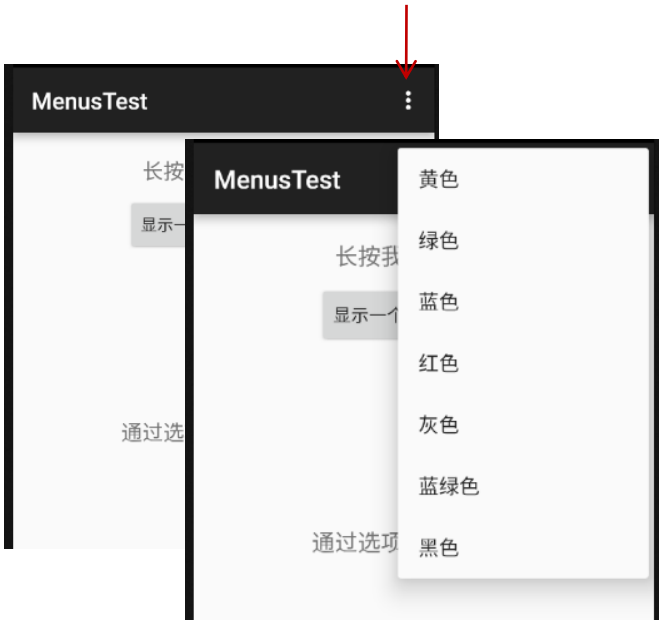
```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.isszym.actionbar">
    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name">
        <activity
            android:name=".SupportActionActivity"
            android:label="@string/label_actionbar"
            android:theme="@style/AppTheme">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
        <activity
            android:name=".SupportToolbarActivity"
            android:label="@string/label_toolbar"
            android:theme="@style/AppToolbarTheme">
            <intent-filter>
                <action android:name="android.intent.action.MAIN1" />
                <category android:name="android.intent.category.LAUNCHER1" />
            </intent-filter>
        </activity>
    </application>
</manifest>
```

菜单

参考 参考 参考

Android中的菜单有如下几种：

（1）**OptionMenu**：选项菜单，android中最常见的菜单，通过Menu键来调用。
（2）**PopupMenu**：弹出式菜单，点击控件将显示悬浮菜单框。
（3）**ContextMenu**：上下文菜单，通过长按某个视图组件后出现的菜单，该组件需注册上下文菜单。



（1）OptionMenu

（2）Popup Menu

（3）Context Menu

如何使用OptionMenu? 利用Activity的回调函数

public boolean **onCreateOptionsMenu**(Menu menu)：调用OptionMenu，在这里完成菜单初始化

public boolean **onOptionsItemSelected**(MenuItem item)：菜单项被选中时触发，这里完成事件处理

public void **onOptionsMenuClosed**(Menu menu)：菜单关闭会调用该方法

public boolean **onPrepareOptionsMenu**(Menu menu)：选项菜单显示前会调用该方法， 可在这里进行菜单的调整(动态加载菜单列表)

public boolean **onMenuOpened**(int featureId, Menu menu)：选项菜单打开以后会调用这个方法

加载OptionMenu的方式：

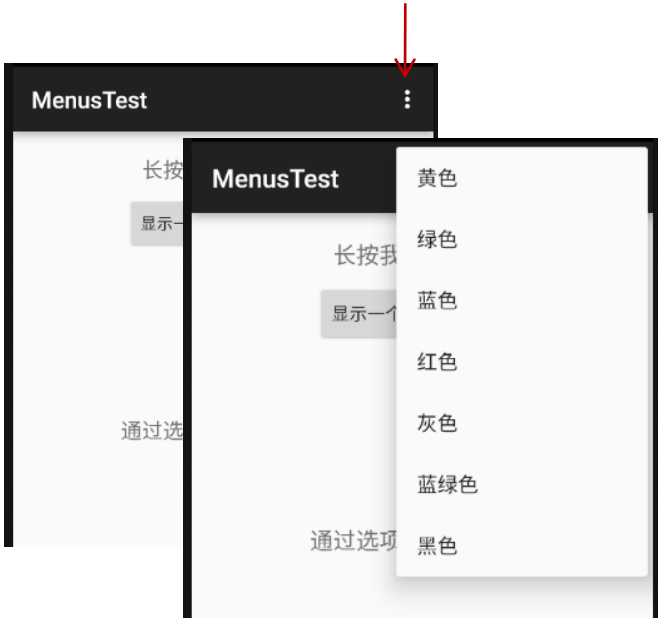
（1）直接通过编写菜单XML文件，然后调用：

getMenuInflater().inflate(R.menu.menu_main, menu);

（2）通过代码动态添加，onCreateOptionsMenu的参数menu，调用add方法添加菜单，add(菜单项的组号，ID，排序号，标题)，另外如果排序号是按添加顺序排序的话都填0即可！

使用OptionMenu的例子

```
@Override
public boolean onCreateOptionsMenu(Menu menu) {
    // Inflate the menu; this adds items to the action bar if it is present.
    menu.add(1, RED, 4, "红色");
    menu.add(1, GREEN, 2, "绿色");
    menu.add(1, BLUE, 3, "蓝色");
    menu.add(1, YELLOW, 1, "黄色");
    menu.add(1, GRAY, 5, "灰色");
    menu.add(1, CYAN, 6, "蓝绿色");
    menu.add(1, BLACK, 7, "黑色");
    return true;
}

@Override
public boolean onOptionsItemSelected(MenuItem item)
{
    int id = item.getItemId();
    switch (id) {
        case RED:
            tv_test.setTextColor(Color.RED);
            break;
        case GREEN:
            tv_test.setTextColor(Color.GREEN);
            break;
```



```
            case BLUE:
                tv_test.setTextColor(Color.BLUE);
                break;
            case YELLOW:

                tv_test.setTextColor(Color.YELLOW);
                break;
            case GRAY:
                tv_test.setTextColor(Color.GRAY);
                break;
            case CYAN:
                tv_test.setTextColor(Color.CYAN);
                break;
            case BLACK:
                tv_test.setTextColor(Color.BLACK);
                break;
        }
        return super.onOptionsItemSelected(item);
    }
```

PopupMenu(弹出式菜单)

在指定View下显示一个弹出菜单，菜单选项可以来自于Menu资源。

使用PopupMenu的例子

menu/menu_popup.xml

```
<?xml version="1.0" encoding="utf-8"?>
<menu xmlns:android="http://schemas.android.com/apk/res/android">
    <item android:id="@+id/lpig" android:title="小猪" />
    <item android:id="@+id/bpig" android:title="大猪" />
</menu>
```

```
btn_show_menu = (Button) findViewById(R.id.btn_show_menu);
btn_show_menu.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        PopupMenu popup = new PopupMenu(MainActivity.this,btn_show_menu);
        popup.getMenuInflater().inflate(R.menu.menu_pop, popup.getMenu());
        popup.setOnMenuItemClickListener(new PopupMenu.OnMenuItemClickListener() {
            @Override
            public boolean onMenuItemClick(MenuItem item) {
                switch (item.getItemId()){
                    case R.id.lpig:
                        Toast.makeText(MainActivity.this,
                                       "你点了小猪~",Toast.LENGTH_SHORT).show();
                        break;
                    case R.id.bpig:
                        Toast.makeText(MainActivity.this,
                                       "你点了大猪~",Toast.LENGTH_SHORT).show();
                        break;
                }
                return true;
            }
        });
        popup.show();
    }
});
```

如何使用ContextMenu？

（1）重写onCreateContextMenu()方法。
（2）为view组件注册上下文菜单，使用registerForContextMenu()方法，参数是View。
（3）重写onContextItemSelected()方法为菜单项指定事件监听器。

menu/sub_menu.xml

```
<?xml version="1.0" encoding="utf-8"?>
<menu xmlns:android="http://schemas.android.com/apk/res/android">
    <item android:id="@+id/submenu" android:title="子菜单使用演示~">
        <menu>
            <group android:checkableBehavior = "none">
                <item android:id="@+id/one" android:title = "子菜单一"/>
                <item android:id="@+id/two" android:title = "子菜单二"/>
                <item android:id="@+id/three" android:title = "子菜单三"/>
            </group>
        </menu>
    </item>
</menu>
```

```
tv_context = (TextView) findViewById(R.id.tv_context);
registerForContextMenu(tv_context);
```

```
@Override
public void onCreateContextMenu(ContextMenu menu, View v,
                                ContextMenu.ContextMenuInfo menuInfo) {

    MenuInflater inflator = new MenuInflater(this);
    inflator.inflate(R.menu.menu_sub, menu);
    super.onCreateContextMenu(menu, v, menuInfo);
}

@Override
public boolean onContextItemSelected(MenuItem item) {
    switch (item.getItemId()) {
        case R.id.one:
            Toast.makeText(MainActivity.this,"你点击了子菜单一",Toast.LENGTH_SHORT).show();
            break;
        case R.id.two:
            item.setCheckable(true);
            Toast.makeText(MainActivity.this,"你点击了子菜单二",Toast.LENGTH_SHORT).show();
            break;
        case R.id.three:
            Toast.makeText(MainActivity.this,"你点击了子菜单三",Toast.LENGTH_SHORT).show();
            item.setCheckable(true);
            break;
    }
    return true;
}
```

桌面管理器

```
public class WidgetMainActivity extends Activity {
    private Button btn;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_imitate_widget_main);
        btn = (Button) findViewById(R.id.btn);
        btn.setOnClickListener(new OnClickListener() {
            @Override
            public void onClick(View v) {
                funClick();
            }
        });
    }
    public void funClick() {
        this.startService(new Intent(getApplicationContext(), MserServes.class));
        // new TableShowView(this).fun(); 如果只是在activity中启动
        // 当activity跑去后台的时候[暂停态，或者销毁态] 设置的显示到桌面的view也会消失
        // 所以这里采用的是启动一个服务，服务中创建我们需要显示到table上的view，
        // 并将其注册到windowManager上
        this.finish();
    }
}
```

```
public class TableShowView extends View {
    // 如果是想显示歌词则继承TextView并复写ondraw方法。
    // 开启一个线程不断的调用ondraw方法去更改你所写的继承自TextView的内容
    // 这里随便写了个集成自view的= =这个不是重点
    Context                 c;
    WindowManager           mWM;       // WindowManager
    WindowManager.LayoutParams mWMParams; // WindowManager参数
    View                    win;
    int tag = 0;
    int oldOffsetX;
    int oldOffsetY;

    public TableShowView(Context context) {
        // TODO Auto-generated constructor stub
        super(context);
        c = context;
    }

    public void fun() {
        // 设置载入view WindowManager参数
        mWM = (WindowManager) c.getSystemService(Context.WINDOW_SERVICE);
        win = LayoutInflater.from(c).inflate(R.layout.activity_imitate_widget_ctrl_wi
        win.setBackgroundColor(Color.TRANSPARENT);
        // 这里是随便载入的一个布局文件
```

```
win.setOnTouchListener(new OnTouchListener() {
  // 触屏监听
  float  lastX, lastY;
  public boolean onTouch(View v, MotionEvent event) {
     final int action = event.getAction();
     float x = event.getX();
     float y = event.getY();
     if(tag == 0){
        oldOffsetX= mWMParams.x; // 偏移量
        oldOffsetY = mWMParams.y; // 偏移量
     }
     if (action == MotionEvent.ACTION_DOWN) {
        lastX = x;
        lastY = y;
     }
     else if (action == MotionEvent.ACTION_MOVE) {
        mWMParams.x += (int) (x - lastX); // 偏移量
        mWMParams.y += (int) (y - lastY); // 偏移量
        tag = 1;
        mWM.updateViewLayout(win, mWMParams);
     }
```

```
        else if (action ==  MotionEvent.ACTION_UP){
                int newOffsetX = mWMParams.x; int newOffsetY = mWMParams.y;
                if(oldOffsetX == newOffsetX && oldOffsetY == newOffsetY){
                   Toast.makeText(c, "你点到我了.....疼！！！！", 1).show();
                }else {
                   tag = 0;
                }
            }
            return true;
        }
    });
    WindowManager wm = mWM;
    WindowManager.LayoutParams wmParams = new WindowManager.LayoutParams();
    mWMParams = wmParams;
    wmParams.type = 2002; // type是关键，这里的2002表示系统级窗口，或2003。
    wmParams.flags = 40;// 这句设置桌面可控
    wmParams.width = 300;
    wmParams.height = 300;
    wmParams.format = -3; // 透明
    wm.addView(win, wmParams);// 这句是重点 给WindowManager中丢入刚才设置的值
                              // 只有addview后才能显示到页面上去。
    // 注册到WindowManager win是要刚才随便载入的layout，
    //wmParams是刚才设置的WindowManager参数集
    // 效果是将win注册到WindowManager中并且它的参数是wmParams中设置
}

}
```

```java
public class MserServes extends Service {
    //这个类只是为了在activity点击button后 在开启一个service

    @Override
    public IBinder onBind(Intent intent) {
        // TODO Auto-generated method stub
        return null;
    }

    public void onCreate() {
        //创建service时一个 实例化一个TableShowView对象
        // 并且调用他的fun()方法把它注册到windowManager上
        super.onCreate();
        new TableShowView(getApplicationContext()).fun();
        System.out.println("开启Mser服务......");
    }
    @Override
    public int onStartCommand(Intent intent, int flags, int startId) {
        // TODO Auto-generated method stub
        return super.onStartCommand(intent, flags, startId);
    }
}
```

AndroidManifest.xml

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.isszym.newnettraffic">
    <uses-permission android:name="android.permission.SYSTEM_ALERT_WINDOW" />
    <uses-permission android:name="com.android.vending.BILLING" />
    <uses-permission android:name="android.permission.SYSTEM_OVERLAY_WINDOW" />

    <application
        android:allowBackup="true"
        android:icon="@drawable/ic_launcher"
        android:label="@string/app_name"
        android:theme="@style/AppTheme" >
        <activity
            android:name="com.example.isszym.newnettraffic.WidgetMainActivity"
            android:label="@string/app_name" >
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
        <service android:name="com.example.isszym.newnettraffic.MserServes" />
    </application>

</manifest>
```

采用 SDK Version 16

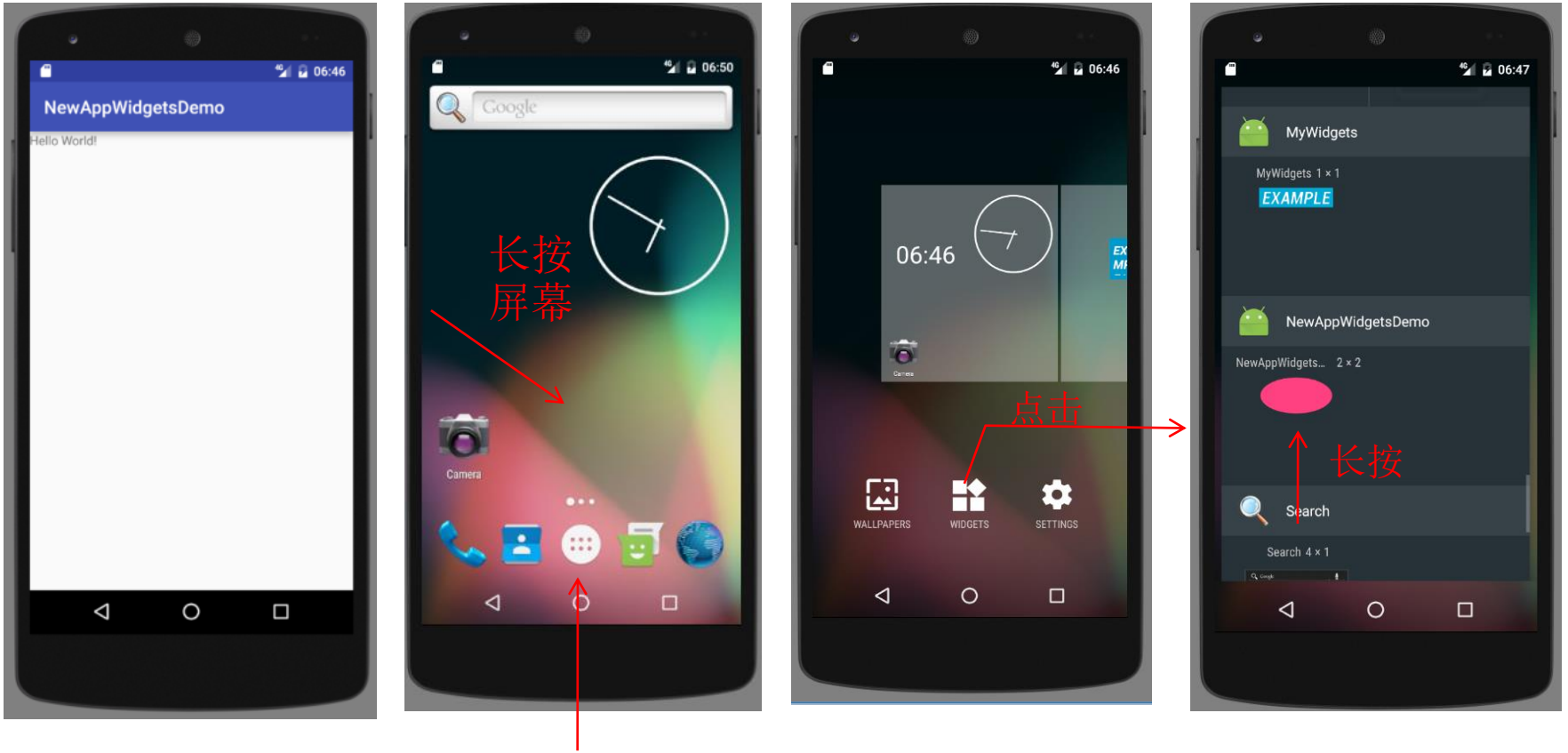
build.gradle

```
android {
    compileSdkVersion 23
    buildToolsVersion "24.0.3"
    defaultConfig {
        applicationId "com.example.isszym.newnettraffic"
        minSdkVersion 16
        targetSdkVersion 16
        versionCode 1
        versionName "1.0"
        testInstrumentationRunner "android.support.test.runner.AndroidJUnitRunner"
    }
    buildTypes {
        release {
            minifyEnabled false
            proguardFiles getDefaultProguardFile('proguard-android.txt'),
                                                  'proguard-rules.pro'
        }
    }
}
```

低版本权限控制简单

桌面图标

建立桌面图标的方法：



* 直接点击中间的app集然后长按拉出app会建立快捷方式（非桌面图标）


06:51
App info


06:52
Google
Camera
06:52


06:53
Remove
长按后
拖至此
处删除
Camera
06:53

```
public class MainActivity extends AppCompatActivity {
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
    }
}
```

activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent">
    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Hello World!" />
</LinearLayout>
```

```
public class ConfigureActivity extends Activity {
    @Override
    protected void onCreate(@Nullable Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        Log.i("zyq","ConfigureActivity:onCreate");
        setResult(RESULT_OK);
        finish();
    }
}
```

```
public class AppWidgetReceiver extends AppWidgetProvider {
    @Override
    public void onReceive(Context context, Intent intent) {
        super.onReceive(context, intent);
        Log.i("zyq","AppWidgetReceiver:onReceive");

    }
    @Override
    public void onUpdate(Context context, AppWidgetManager appWidgetManager, int[] appWidgetIds){
        super.onUpdate(context, appWidgetManager, appWidgetIds);
        Log.i("zyq","AppWidgetReceiver:onUpdate");
        RemoteViews remoteViews = new RemoteViews(context.getPackageName(),
                                        R.layout.app_widget_layout);
        PendingIntent i = PendingIntent.getActivity(context,0,
                                        new Intent(context,MainActivity.class),0);
        remoteViews.setOnClickPendingIntent(R.id.text_clock_container,i);
        for (int i1 = 0;i1<appWidgetIds.length;i1++){
            appWidgetManager.updateAppWidget(appWidgetIds[i1], remoteViews);
        }
    }
    @Override
    public void onAppWidgetOptionsChanged(Context context,
        AppWidgetManager appWidgetManager, int appWidgetId, Bundle newOptions) {
        super.onAppWidgetOptionsChanged(context, appWidgetManager,
                                        appWidgetId, newOptions);
        Log.i("zyq","AppWidgetReceiver:onAppWidgetOptionsChanged");
    }
```

```
    @Override
    public void onDeleted(Context context, int[] appWidgetIds) {
        super.onDeleted(context, appWidgetIds);
        Log.i("zyq","AppWidgetReceiver:onDeleted");
    }

    @Override
    public void onEnabled(Context context) {
        super.onEnabled(context);
        Log.i("zyq","AppWidgetReceiver:onEnabled");
    }
    @Override
    public void onDisabled(Context context) {
        super.onDisabled(context);
        Log.i("zyq","AppWidgetReceiver:onDisabled");
    }
    @Override
    public void onRestored(Context context, int[] oldWidgetIds, int[] newWidgetIds) {
        super.onRestored(context, oldWidgetIds, newWidgetIds);
        Log.i("zyq","AppWidgetReceiver:onRestored");
    }
}
```

app_widget_layout.xml

```
<?xml version="1.0" encoding="utf-8"?>
<FrameLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:id="@+id/text_clock_container">
<TextClock
    android:textAlignment="center"
    android:textSize="48sp"
    android:id="@+id/text_clock"
    android:layout_width="match_parent"
    android:layout_height="wrap_content" />
</FrameLayout>
```

xml\app_widget_provider_info.xml

```
<?xml version="1.0" encoding="utf-8"?>
<appwidget-provider xmlns:android="http://schemas.android.com/apk/res/android"
    android:initialLayout="@layout/app_widget_layout"
    android:minHeight="80dp"
    android:minWidth="80dp"
    android:minResizeHeight="80dp"
    android:minResizeWidth="80dp"
    android:resizeMode="vertical"
    android:widgetCategory="home_screen"
    android:previewImage="@drawable/shape_xxx"
    android:updatePeriodMillis="10000"
    android:configure="com.example.isszym.newappwidgetsdemo.ConfigureActivity">

</appwidget-provider>
```

drawable\shape_xxx.xml

```
<?xml version="1.0" encoding="utf-8"?>
<shape xmlns:android="http://schemas.android.com/apk/res/android"
    android:shape="oval">
    <size android:height="40dp"
        android:width="80dp"/>
    <solid android:color="@color/colorAccent"/>

</shape>
```

Toast和Notification

```
Toast.makeText(MainActivity.this, "Button2 is clicked!",
                                        Toast.LENGTH_SHORT).show();
```

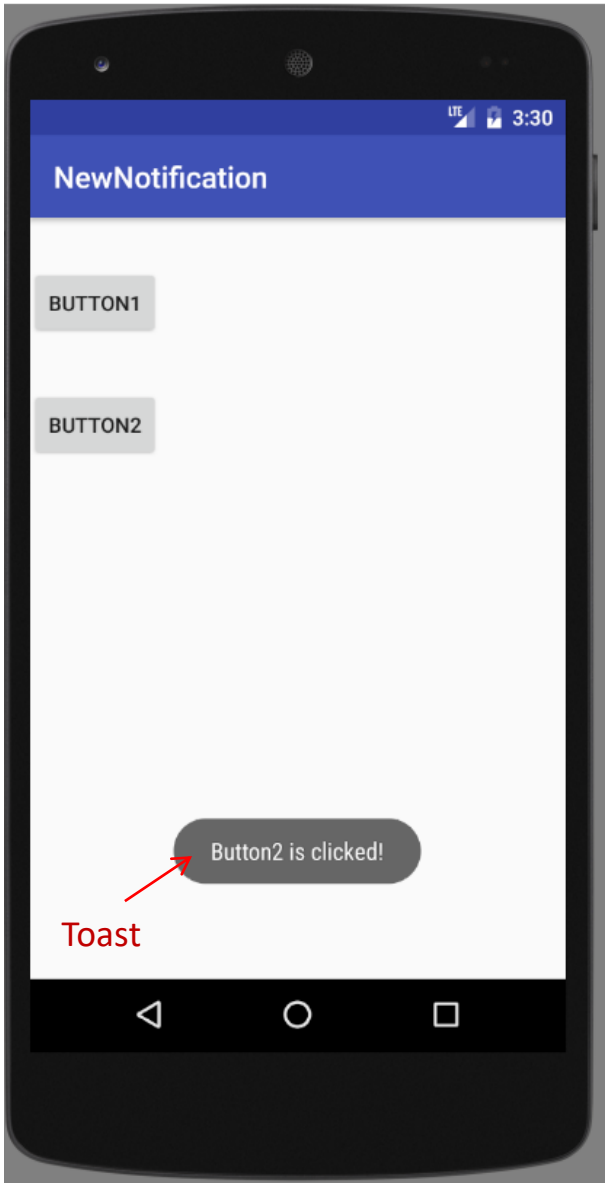
第一个参数：Context
第二个参数：显示内容
第三个参数：显示时间的长短

```
Notification.Builder builder = new Notification.Builder(this)
            .setTicker("显示于屏幕顶端状态栏的文本")
            .setSmallIcon(R.mipmap.ic_launcher)
            .setContentTitle("My notification")
             .setContentText("Hello World!");
    Notification note = builder.build();
notificationManager.notify(NotificationId, note);
```

点击BUTTON2后

Notification

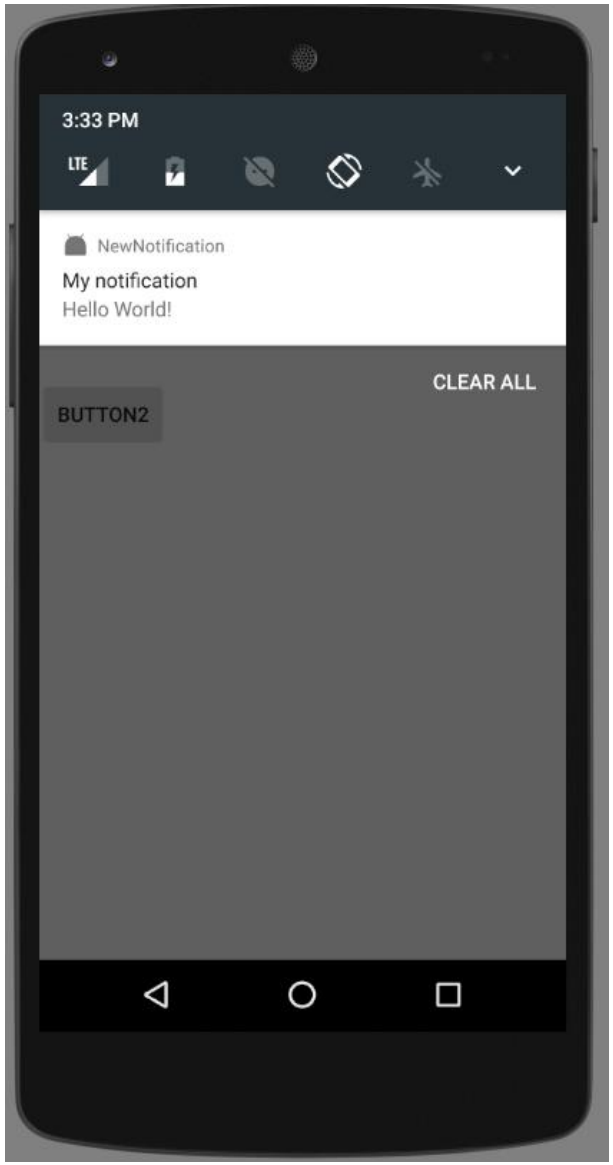
Toast



Notification

点击
BUTTON1
后得到

点击Notification后



```
import android.app.NotificationManager;
import android.app.PendingIntent;
import android.content.Context;
import android.content.Intent;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.Toast;
import android.app.Notification;

public class MainActivity extends AppCompatActivity {
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        Button btn2 = (Button) findViewById(R.id.button2);
        btn2.setOnClickListener(new View.OnClickListener(){
          @Override
          public void onClick(View v){
             Toast.makeText(MainActivity.this,
                 "Button2 is clicked!", Toast.LENGTH_SHORT).show();
          }
       });
   }
```

参考

```
  // 把事件处理器直接绑定到标签上
  public void clickHandler1(View source) {
    int NotificationId = 001,requestCode=100;
    NotificationManager notificationManager =
                      (NotificationManager)getSystemService(Context.NOTIFICATION_SERVICE);
    Notification.Builder builder = new Notification.Builder(this)
            .setTicker("显示于屏幕顶端状态栏的文本")
            .setSmallIcon(R.mipmap.ic_launcher)
            .setContentTitle("My notification")
             .setContentText("Hello World!");
    Notification note = builder.build();
    //FLAG_ONGOING_EVENT表明有程序在运行，该Notification不可由用户清除
    note.flags = Notification.FLAG_ONGOING_EVENT;
    //通过Intent，使得点击Notification之后会启动新的Activity
    Intent i = new Intent(this, Main2Activity.class);
    //该标志位表示如果Intent要启动的Activity在栈顶，则无须创建新的实例
    i.setFlags(Intent.FLAG_ACTIVITY_SINGLE_TOP);
    PendingIntent pendingIntent = PendingIntent.getActivity(this, requestCode, i,
                                                            PendingIntent.FLAG_UPDATE_CURRENT);
    notificationManager.notify(NotificationId, note);
  }
}
```

activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:id="@+id/activity_main"
    android:layout_width="match_parent"
    android:layout_height="match_parent">

    <Button
        android:text="Button2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="34dp"
        android:id="@+id/button2"
        android:layout_below="@+id/button1"
        android:layout_alignParentStart="true" />

    <Button
        android:text="Button1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="33dp"
        android:id="@+id/button1"
        android:onClick="clickHandler1"
        android:layout_alignParentStart="true" />
</RelativeLayout>
```