
第四章 UI布局

北京邮电大学 计算机学院

刘伟

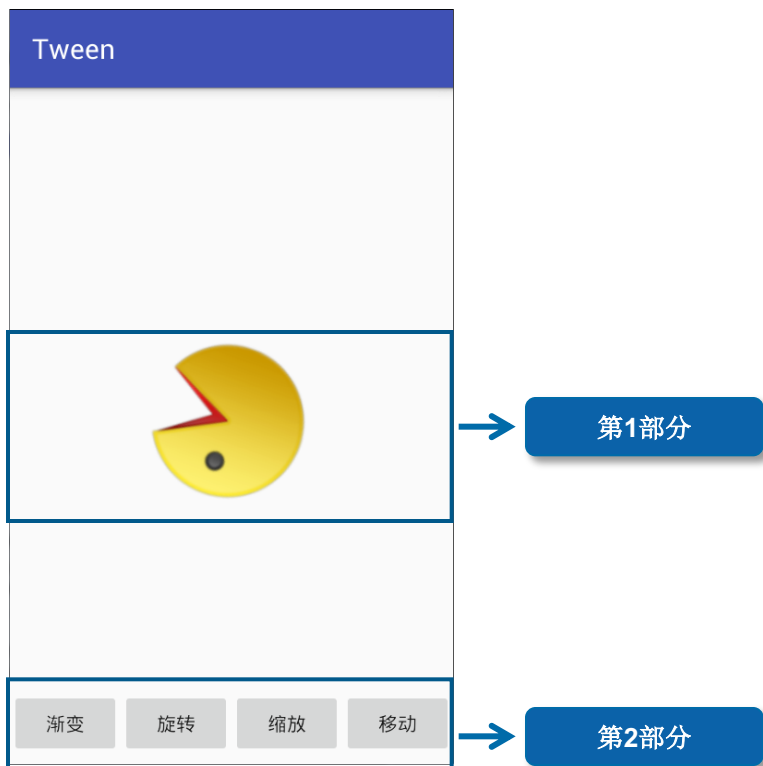
w.liu@foxmail.com

本章重点

- 布局的创建
- 布局的类型
- 常用控件
- 常见对话框

■ 4.1 布局的创建

■ 4.1.1 关于布局



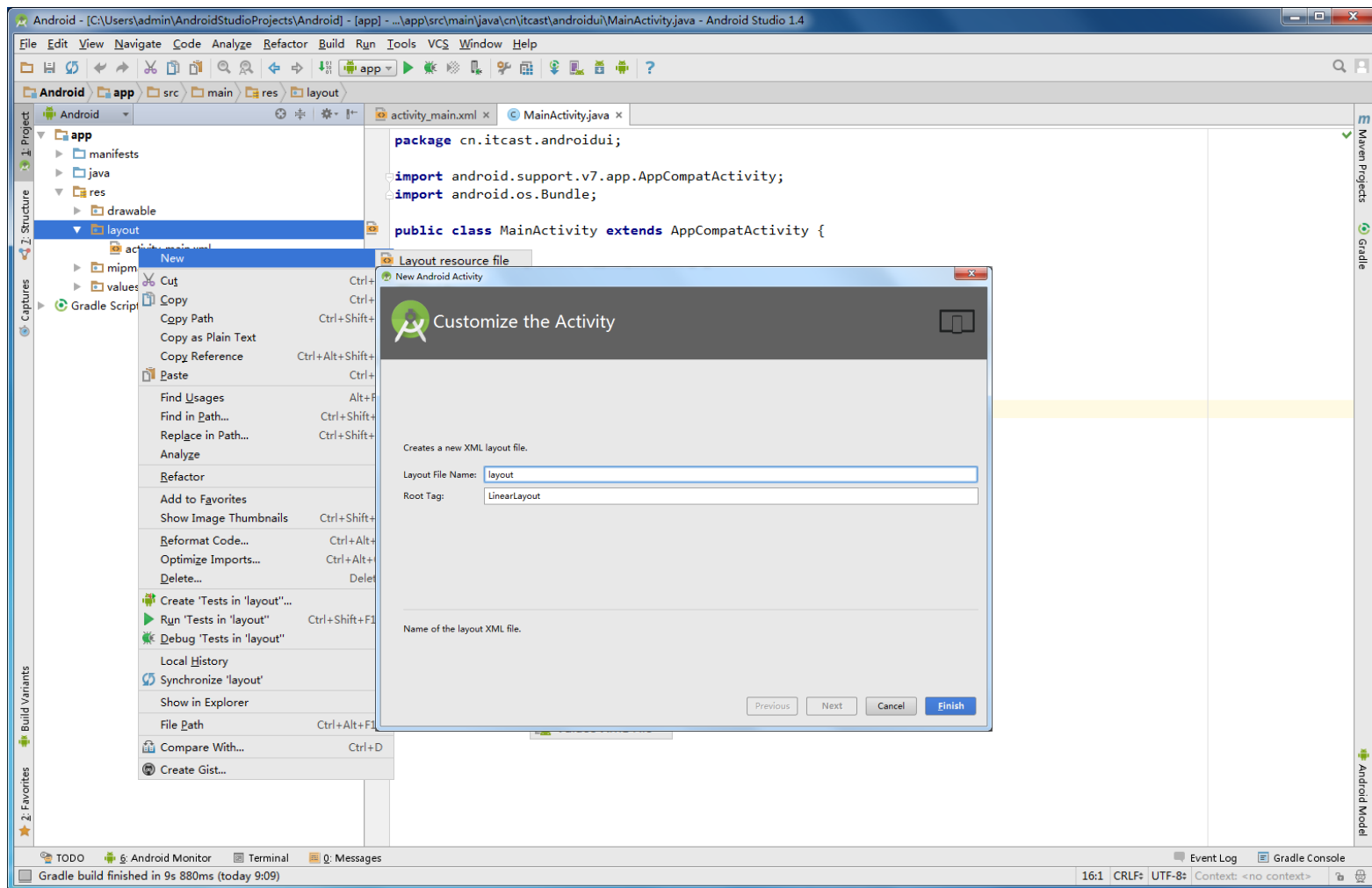
■ 4.1 布局的创建

■ 4.1.1 关于布局

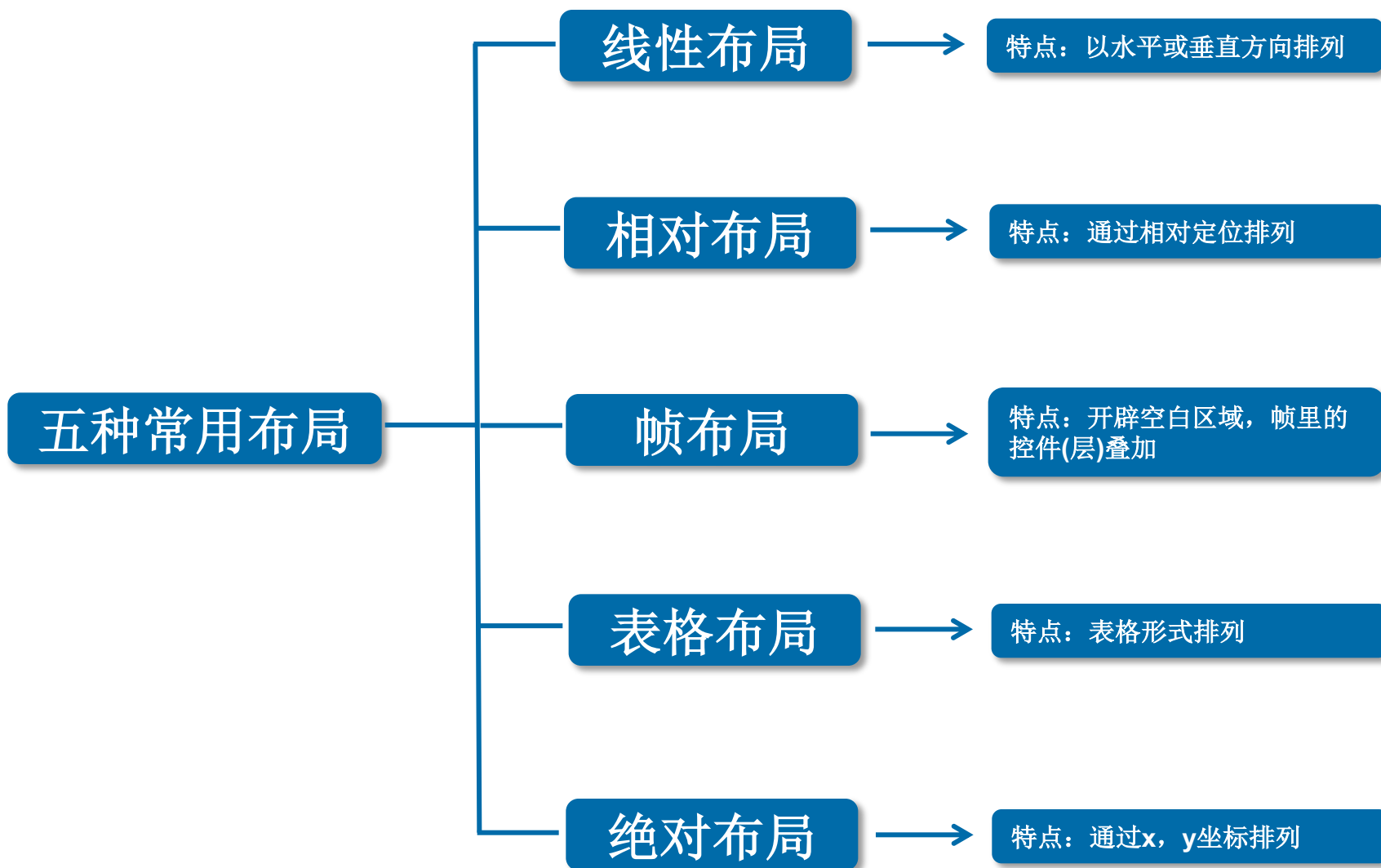
- 在Android程序中界面是通过布局文件设定的，在每个应用程序创建时会默认包含一个主界面布局，该布局位于res/layout目录中。
- 实际开发中每个应用程序都包含多个界面，而程序默认提供的一个主界面布局无法满足需求，因此经常会在程序中添加多个布局。

■ 4.1 布局的创建

■ 4.1.2 步骤



■ 4.2 布局的类型



■ 4.2 布局的类型

■ 4.1.1 线性布局

- 线性布局（LinearLayout）主要以水平或垂直方式来显示界面中的控件。
当控件水平排列时，显示顺序依次为从左到右，当控件垂直排列时，显示顺序依次为从上到下。

orientation属性

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:orientation="vertical">
</LinearLayout>
```

■ 4.2 布局的类型

■ 4.1.1 线性布局-注意事项

- 当控件水平排列时，控件属性layout_width只能设置为wrap_content（包裹内容让当前控件根据控件内容大小自动伸缩），否则其余控件会被挤出屏幕右侧不显示。同理，如果控件垂直排列也会出现同样情况。
- 当控件水平排列时，如果控件未占满一行，会留有空白区域，这样既不美观又浪费空间。此时，可以利用layout_weight属性解决这个问题，该属性被称为权重，通过比例调整布局中所有控件的大小。

<Button

android:layout_width="0dp"

android:layout_height="wrap_content"

android:layout_weight="2"/>

注意：当控件使用权重属性时，布局宽度属性值通常设置为0。

■ 4.2 布局的类型

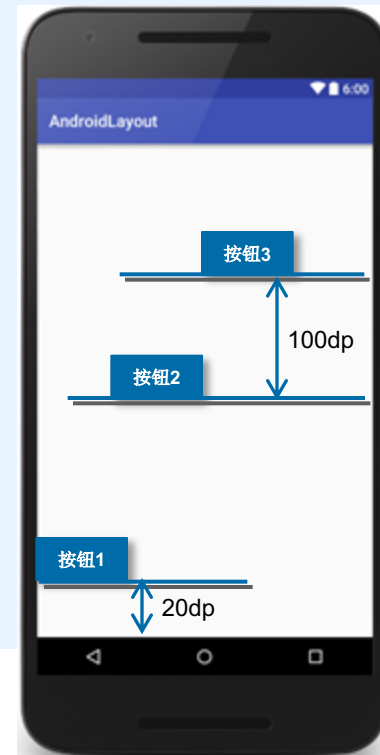
■ 4.2.2 相对布局

- 相对布局（RelativeLayout）是通过相对定位的方式指定控件位置，即以其它控件或父容器为参照物，摆放控件位置。
- 在设计相对布局时要遵循控件之间的依赖关系，后放入控件的位置依赖于先放入的控件。

■ 4.2 布局的类型

■ 4.2.2 相对布局

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingBottom="20dp">
    .....
</RelativeLayout>
```



■ 4.2 布局的类型

■ 4.2.2 相对布局-控件位置属性

控件属性	功能描述
android:layout_centerInParent	设置当前控件位于父布局的中央位置
android:layout_centerVertical	设置当前控件位于父布局的垂直居中位置
android:layout_centerHorizontal	设置当前控件位于父控件的水平居中位置
android:layout_above	设置当前控件位于某控件上方
android:layout_below	设置当前控件位于某控件下方
android:layout_toLeftOf	设置当前控件位于某控件左侧
android:layout_alignParentTop	设置当前控件停靠于布局顶端
android:layout_alignParentLeft	设置当前控件停靠于布局左侧
android:layout_alignParentRight	设置当前控件停靠于布局右侧
android:layout_alignParentBottom	设置当前控件停靠于布局底端
android:layout_alignTop	设置当前控件的上边界与某控件的上边界对齐
android:layout_alignBottom	设置当前控件的下边界与某控件的下边界对齐
android:layout_alignLeft	设置当前控件的左边界与某控件的左边界对齐
android:layout_alignRight	设置当前控件的右边界与某控件的右边界对齐

■ 4.2 布局的类型

■ 4.2.2 相对布局-控件位置属性

控件属性	功能描述
android:layout_marginTop	设置当前控件上边界与某控件的距离
android:layout_marginBottom	设置当前控件底边界与某控件的距离
android:layout_marginLeft	设置当前控件左边界与某控件的距离
android:layout_marginRight	设置当前控件右边界与某控件的距离

■ 4.2 布局的类型

■ 4.2.2 相对布局-控件内边距属性

控件属性	功能描述
android:paddingTop	设置布局顶部内边距的距离
android:paddingBottom	设置布局底部内边距的距离
android:paddingLeft	设置布局左边内边距的距离
android:paddingRight	设置布局右边内边距的距离
android:padding	设置布局四周内边距的距离

■ 4.2 布局的类型

■ 4.2.2 相对布局-常用单位

- 为了让程序拥有更好的屏幕适配能力，在指定控件和布局宽高时应尽量避免将控件宽高设置为固定值。但特殊情况下，需要使用指定宽高值时，可以选择使用以下四种单位：

- px：像素，即在屏幕中可以显示最小元素单位。
- pt：磅数，一磅等于1/72英寸，一般pt会作为字体的单位来显示。
- dp：基于屏幕密度的抽象单位。不同设备有不同的显示效果，根据设备分辨率的不同来确定控件的尺寸。
- sp：可伸缩像素，采用与dp相同的设计理念，推荐设置文字大小时使用。

■ 4.2 布局的类型

■ 4.2.3 帧布局

- 帧布局（FrameLayout）为每个加入其中的控件创建一个空白区域（称为一帧，每个控件占据一帧）。
- 所有控件都默认显示在屏幕左上角，按照先后放入的顺序重叠摆放。帧布局的大小由内部最大控件的决定。

■ 4.2 布局的类型

■ 4.2.3 帧布局

```
<FrameLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:foreground="@mipmap/ic_launcher"
    android:foregroundGravity="left" >
</FrameLayout>
```



■ 4.2 布局的类型

■ 4.2.4 表格布局

- 表格布局（TableLayout）是以表格形式排列控件的，通过行和列将界面划分为多个单元格，每个单元格都可以添加控件。
- 表格布局需要和TableRow配合使用，每一行都由TableRow对象组成，因此TableRow的数量决定表格的行数。而表格的列数是由包含最多控件的TableRow决定的，例如第1个TableRow有两个控件，第2个TableRow有三个控件，则表格列数为3。

■ 4.2 布局的类型

■ 4.2.4 表格布局

```
<TableLayout xmlns:android="http://schemas.android.com/apk/res/android"
```

```
    android:layout_width="wrap_content"
```

```
    android:layout_height="wrap_content"
```

```
    android:stretchColumns="2">
```

```
    <TableRow>
```

```
        <Button
```

```
            android:layout_width="wrap_content"
```

```
            android:layout_height="wrap_content"
```

```
            android:layout_column="0"
```

```
            android:text="按钮1" />
```

```
        </TableRow>
```

```
</TableLayout>
```



■ 4.2 布局的类型

■ 4.2.4 表格布局-属性

表格布局属性

布局属性	功能描述
android:stretchColumns	设置该列被拉伸
android:shrinkColumns	设置该列被收缩
android:collapseColumns	设置该列被隐藏

表格布局控件属性

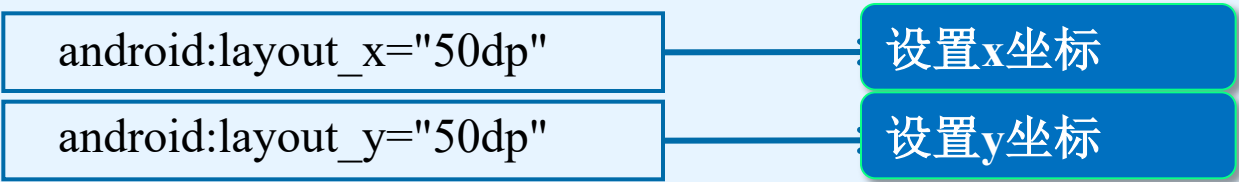
控件属性	功能描述
android:layout_column	设置该单元显示位置
android:layout_span	设置该单元格占据几行，默认为1行

■ 4.2 布局的类型

■ 4.2.5 绝对布局

- 绝对布局 (AbsoluteLayout) 是通过指定x、 y坐标来控制每一个控件位置的。

```
<AbsoluteLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent">
    <Button
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_x="50dp"
        android:layout_y="50dp"
        android:text="按钮1"/>
</AbsoluteLayout>
```

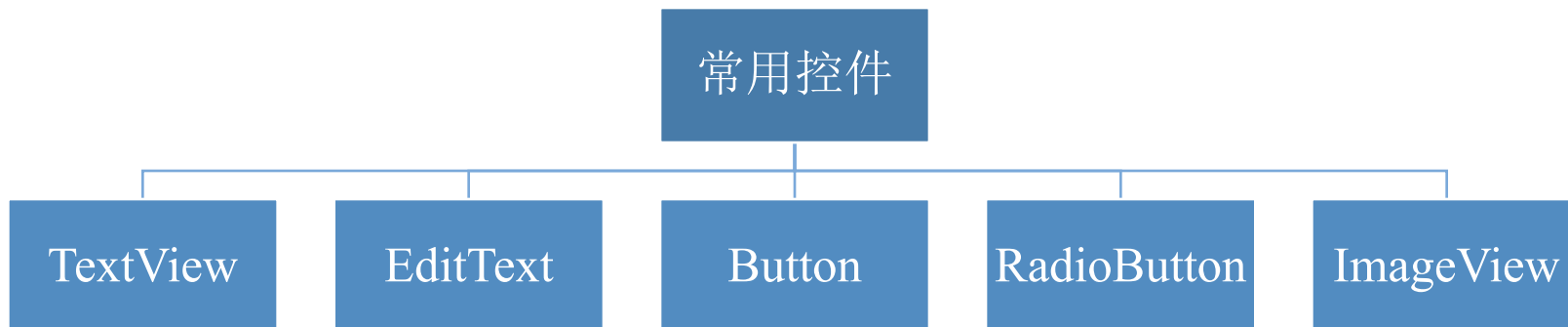


设置x坐标

设置y坐标

■ 4.3 常用控件

■ 控件的分类



■ 4.3 常用控件

■ 4.3.1 TextView

TextView是用于显示文字(字符串)的控件，可在代码中通过设置属性改变文字的大小、颜色、样式等功能。

```
<TextView  
    android:layout_width="match_parent"  
    android:layout_height="wrap_content"  
    android:text="Hello World! "  
    android:textColor="#000000"  
    android:textSize="25sp"  
    android:gravity="center"  
>
```

■ 4.3 常用控件

■ 4.3.1 TextView属性

- 1、android:autoLink设置是否当文本为URL链接/email/电话号码/map时，文本显示为可点击的链接。可选值(none/web/email/phone/map/all)
- 2、android:autoText如果设置，将自动执行输入值的拼写纠正。此处无效果，在显示输入法并输入的时候起作用。
- 3、android:bufferType指定getText()方式取得的文本类别。选项editable 类似于StringBuilder可追加字符，也就是说getText后可调用append方法设置文本内容。spannable 则可在给定的字符区域使用样式，参见[这里1](#)、[这里2](#)。
- 4、android:capitalize设置英文字母大写类型。此处无效果，需要弹出输入法才能看得到，参见EditView此属性说明。
- 5、android:cursorVisible设定光标为显示/隐藏，默认显示。
- 6、android:digits设置允许输入哪些字符。如“1234567890.+*/*% ()”
- 7、android:drawableBottom在text的下方输出一个drawable，如图片。如果指定一个颜色的话会把text的背景设为该颜色，并且同时和background使用时覆盖后者。
- 8、android:drawableLeft在text的左边输出一个drawable，如图片。
- 9、android:drawablePadding设置text与drawable(图片)的间隔，与drawableLeft、drawableRight、drawableTop、drawableBottom一起使用，可设置为负数，单独使用没有效果。
- 10、android:drawableRight在text的右边输出一个drawable。
- 11、android:drawableTop在text的正上方输出一个drawable。
- 12、android:editable设置是否可编辑。
- 13、android:editorExtras设置文本的额外的输入数据。
- 14、android:ellipsize设置当文字过长时,该控件该如何显示。有如下值设置：“start”——省略号显示在开头;“end”——省略号显示在结尾;“middle”——省略号显示在中间;“marquee”——以跑马灯的方式显示(动画横向移动)

■ 4.3 常用控件

■ 4.3.1 TextView属性

- 15、`android:freezesText`设置保存文本的内容以及光标的位置。
- 16、`android:gravity`设置文本位置，如设置成“center”，文本将居中显示。
- 17、`android:hintText`为空时显示的文字提示信息，可通过`textColorHint`设置提示信息的颜色。此属性在`EditText`中使用，但是这里也可以用。
- 18、`android:imeOptions`附加功能，设置右下角IME动作与编辑框相关的动作，如`actionDone`右下角将显示一个“完成”，而不设置默认是一个回车符号。这个在`EditText`中再详细说明，此处无用。
- 19、`android:imeActionId`设置IME动作ID。
- 20、`android:imeActionLabel`设置IME动作标签。
- 21、`android:includeFontPadding`设置文本是否包含顶部和底部额外空白，默认为true。
- 22、`android:inputMethod`为文本指定输入法，需要完全限定名(完整的包名)。例如：`com.google.android.inputmethod.pinyin`，但是这里报错找不到。
- 23、`android:inputType`设置文本的类型，用于帮助输入法显示合适的键盘类型。在`EditText`中再详细说明，这里无效果。
- 24、`android:linksClickable`设置链接是否点击连接，即使设置了`autoLink`。
- 25、`android:marqueeRepeatLimit`在`ellipsize`指定`marquee`的情况下，设置重复滚动的次数，当设置为`marquee_forever`时表示无限次。
- 26、`android:ems`设置`TextView`的宽度为N个字符的宽度。这里测试为一个汉字字符宽度
- 27、`android:maxEms`设置`TextView`的宽度为最长为N个字符的宽度。与`ems`同时使用时覆盖`ems`选项。
- 28、`android:minEms`设置`TextView`的宽度为最短为N个字符的宽度。与`ems`同时使用时覆盖`ems`选项。
- 29、`android:maxLength`限制显示的文本长度，超出部分不显示。
- 30、`android:lines`设置文本的行数，设置两行就显示两行，即使第二行没有数据。
- 31、`android:maxLines`设置文本的最大显示行数，与`width`或者`layout_width`结合使用，超出部分自动换行，超出行数将不显示。

■ 4.3 常用控件

■ 4.3.1 TextView属性

- 32、android:minLines设置文本的最小行数，与lines类似。
- 33、android:lineSpacingExtra设置行间距。
- 34、android:lineSpacingMultiplier设置行间距的倍数。如” 1.2”
- 35、android:numeric如果被设置，该TextView有一个数字输入法。此处无用，设置后唯一效果是TextView有点击效果，此属性在EdtiView将详细说明。
- 36、android:password以小点”.”显示文本
- 37、android:phoneNumber设置为电话号码的输入方式。
- 38、android:privateImeOptions设置输入法选项，此处无用，在EditText将进一步讨论。
- 39、android:scrollHorizontally设置文本超出TextView的宽度的情况下，是否出现横拉条。
- 40、android:selectAllOnFocus如果文本是可选择的，让他获取焦点而不是将光标移动为文本的开始位置或者末尾位置。TextView中设置后无效果。
- 41、android:shadowColor指定文本阴影的颜色，需要与shadowRadius一起使用。
- 42、android:shadowDx设置阴影横向坐标开始位置。
- 43、android:shadowDy设置阴影纵向坐标开始位置。
- 44、android:shadowRadius设置阴影的半径。设置为0.1就变成字体的颜色了，一般设置为3.0的效果比较好。
- 45、android:singleLine设置单行显示。如果和layout_width一起使用，当文本不能全部显示时，后面用“...”来表示。如android:text="test_ singleLine "
android:singleLine="true" android:layout_width="20dp"将只显示“t...”。如果不设置singleLine或者设置为false，文本将自动换行
- 46、android:text设置显示文本。

■ 4.3 常用控件

■ 4.3.1 TextView属性

47、android:textAppearance设置文字外观。如“?android:attr/textAppearanceLargeInverse”这里引用的是系统自带的一个外观，?表示系统是否有这种外观，否则使用默认的外观。可设置的值如下：

textAppearanceButton/textAppearanceInverse/textAppearanceLarge/textAppearanceLargeInverse/textAppearanceMedium/textAppearanceMediumInverse/textAppearanceSmall/textAppearanceSmallInverse

48、android:textColor设置文本颜色

49、android:textColorHighlight被选中文字的底色，默认为蓝色

50、android:textColorHint设置提示信息文字的颜色，默认为灰色。与hint一起使用。

51、android:textColorLink文字链接的颜色。

52、android:textScaleX设置文字之间间隔，默认为1.0f。

53、android:textSize设置文字大小，推荐度量单位”sp”，如”15sp”

54、android:textStyle设置字形[bold(粗体) 0, italic(斜体) 1, bolditalic(又粗又斜) 2] 可以设置一个或多个，用“|”隔开

55、android:typeface设置文本字体，必须是以下常量值之一：normal 0, sans 1, serif 2, monospace(等宽字体) 3]

56、android:height设置文本区域的高度，支持度量单位：px(像素)/dp/sp/in/mm(毫米)

57、android:maxHeight设置文本区域的最大高度

58、android:minHeight设置文本区域的最小高度

59、android:width设置文本区域的宽度，支持度量单位：px(像素)/dp/sp/in/mm(毫米)，与layout_width的区别看这里。

60、android:maxWidth设置文本区域的最大宽度

61、android:minWidth设置文本区域的最小宽度

■ 4.3 常用控件

■ 4.3.2 EditText

EditText继承自TextView，可以进行编辑操作，将用户信息传递给Android程序。还可以为EditText控件设置监听器，用来测试用户输入的内容是否合法。

```
<EditText  
    android:layout_width="match_parent"  
    android:layout_height="wrap_content"  
    android:hint="请输入姓名"  
    android:maxLines="2"  
    android:textColor="#000000"  
    android:textSize="20sp"  
    android:textStyle="italic"  
>
```

■ 4.3 常用控件

■ 4.3.3 Button

Button是按钮，是用于响应用户的一系列点击事件，使程序更加流畅和完整。

```
<Button  
    android:id="@+id/btn"  
    android:text="按钮"  
    android:layout_width="match_parent"  
    android:layout_height="wrap_content"  
    android:onClick="click"  
>
```

■ 4.3 常用控件

■ 4.3.3 Button-点击事件实现方式

指定Button的onClick属性方式

- 首先在layout文件中指定onClick属性:

```
android:onClick="click"
```

- 然后在Activity中实现这个click方法

```
public void click(View v){  
    Log.i("指定onClick属性方式", "button is clicked");  
}
```

注意：Activity中实现的方法名称要与onClick属性设置的名称一致。

■ 4.3 常用控件

■ 4.3.3 Button-点击事件实现方式

独立类方式

- 首先为按钮设置监听器:

```
btn.setOnClickListener(myListener);
```

- 在onCreate()方法外实现接口

```
onClickListener myListener = new OnClickListener() {  
    @Override  
    public void onClick(View v) {  
        Log.i("独立类方式", "button is clicked");  
    }  
}
```

■ 4.3 常用控件

■ 4.3.3 Button-点击事件实现方式

接口方式

- 首先当前Activity实现OnClickListener接口

```
public class MainActivity extends Activity implements OnClickListener;
```

- 然后实现接口方法

```
@Override  
  
public void onClick(View v) {  
    Log.i("接口方式", "button is clicked");  
}
```

- 最后绑定到Button上

```
btn.setOnClickListener(this);
```

■ 4.3 常用控件

■ 4.3.3 Button-点击事件实现方式

匿名内部类方式

- 在Activity中添加匿名内部类

```
btn.setOnClickListener(new View.OnClickListener() {  
    @Override  
    public void onClick(View v) {  
        Log.i("匿名内部类方式", "button is clicked");  
    }  
});
```


■ 4.3 常用控件

■ 4.3.4 RadioButton

- RadioButton为单选按钮，它需要与RadioGroup配合使用，提供两个或多个互斥的选项集。
- RadioGroup是单选组合框，可容纳多个RadioButton，并把它们组合在一起，实现单选状态。

■ 4.3 常用控件

■ 4.3.4 RadioButton

```
<RadioGroup  
    android:layout_width="match_parent"  
    android:layout_height="wrap_content"  
    android:orientation="vertical">
```



控制RadioButton按钮的排列方向

```
<RadioButton  
    android:layout_width="wrap_content"  
    android:layout_height="wrap_content"  
    android:text="男" />
```

```
<RadioButton  
    android:layout_width="wrap_content"  
    android:layout_height="wrap_content"  
    android:text="女" />
```

```
</RadioGroup>
```

■ 4.3 常用控件

■ 4.3.4 RadioButton-设置监听事件

利用setOnCheckedChangeListener()监听RadioGroup控件状态，通过if语句判断被选中RadioButton的id。

```
radioGroup.setOnCheckedChangeListener(new  
    RadioGroup.OnCheckedChangeListener() {  
    @Override  
    public void onCheckedChanged(RadioGroup group, int checkedId) {  
        if (checkedId == R.id.rbtn) {  
            textView.setText("您的性别是：男");  
        } else {  
            textView.setText("您的性别是：女");  
        }  
    }  
});
```

■ 4.3 常用控件

■ 4.3.5 ImageView

ImageView是视图控件，它继承自View，其功能是在屏幕中显示图像。ImageView类可以从各种来源加载图像（如资源库或网络），并提供缩放、裁剪、着色（渲染）等功能。

```
<ImageView  
    android:layout_width="wrap_content"  
    android:layout_height="wrap_content"  
    android:background="@drawable/bg" />
```

```
<ImageView  
    android:layout_width="100dp"  
    android:layout_height="100dp"  
    android:src="@android:drawable/sym_def_app_icon"/>
```

■ 4.4 常见对话框

■ 常见对话框分类

□ 对话框也是程序与用户交互的一种方式，通常用于显示当前程序提示信息以及相关说明，以小窗口形式展现。

□ 常见对话框有以下几种：

普通对话框

进度条对话框

单选对话框

消息对话框

多选对话框

自定义对话框

■ 4.4 常见对话框

■ 4.4.1 普通对话框

普通对话框（Dialog）一般只会显示提示信息，并具有确定和取消按钮。

```
AlertDialog dialog;  
  
dialog = new AlertDialog.Builder(this)  
    .setTitle("Dialog对话框")  
    .setMessage("是否确定退出？")  
    .setIcon(R.mipmap.ic_launcher)  
    .setPositiveButton("确定", null)  
    .setNegativeButton("取消", null)  
    .create();  
  
dialog.show();
```

■ 4.4 常见对话框

■ 4.4.2 单选对话框

单选对话框和RadioButton作用类似，只能选择一个选项，它是通过AlertDialog对象调用setSingleChoiceItems()方法创建的。

```
new AlertDialog.Builder(this)
    .setTitle("请选择性别")
    .setIcon(R.mipmap.ic_launcher)
    .setSingleChoiceItems(new String[]{"男", "女"}, 0,
        new DialogInterface.OnClickListener() {
            public void onClick(DialogInterface dialog, int which) {
            }
        })
    .setPositiveButton("确定", null)
    .show();
```

■ 4.4 常见对话框

■ 4.4.3 多选对话框

多选对话框通常需要在需要勾选多种选项时使用，例如添加兴趣爱好、喜爱的电影等。创建多选对话框与创建单选对话框类似，调用 `setMultiChoiceItems()` 方法就可实现。

```
new AlertDialog.Builder(this)
    .setTitle("请添加兴趣爱好！")
    .setIcon(R.mipmap.ic_launcher)
    .setMultiChoiceItems(new String[]{"旅游", "美食", "汽车", "宠物"},
        null,
        null)
    .setPositiveButton("确定", null)
    .show();
```


■ 4.4 常见对话框

■ 4.4.4 进度条对话框

进度条对话框一般在应用程序实现耗时操作时使用。Android中提供了两种进度条样式，圆形进度条和水平进度条。

```
ProgressDialog prodialog;  
prodialog = new ProgressDialog(this);  
prodialog.setTitle("进度条对话框");  
prodialog.setIcon(R.mipmap.ic_launcher);  
prodialog.setMessage("正在下载请等候...");  
prodialog.setProgressStyle(ProgressDialog.STYLE_HORIZONTAL);  
prodialog.show();
```

■ 4.4 常见对话框

■ 4.4.5 消息对话框

消息对话框（Toast）是轻量级信息提醒机制，显示在应用程序界面的最上层，一段时间后自动消失不会打断当前操作，也不获得焦点。

```
Toast.makeText ( this, "Hello,Toast" , Toast.LENGTH_SHORT ).show();
```

■ 4.4 常见对话框

■ 4.4.6 自定义对话框

- 为了提高用户体验，达到更理想的效果，可根据需求自定义对话框样式。具体创建步骤如下：
 - 创建布局
 - 创建一个自定义对话框的布局文件（ my_dialog.xml ），布局中需要设定对话框的标题、对话框内容以及确定和取消按钮。
 - 创建自定义对话框
 - 创建一个类MyDialog继承自Dialog类，主要用于初始化自定义对话框中的控件以及响应按钮的点击事件。
 - 使用自定义对话框
 - 在MainActivity中，调用MyDialog的构造方法将自定义对话框显示出来

■ 4.3 常用控件

■ 4.3.6 实战演练——QQ登录界面

1 功能描述： 搭建QQ登录界面UI。

2 技术要点： 使用前几节学习的布局 and 控件搭建。

3 实现步骤：

- ① 导入图片资源
- ② 思考界面结构
- ③ 布局文件中编写界面代码

案例代码（详见教材P23—P25）

