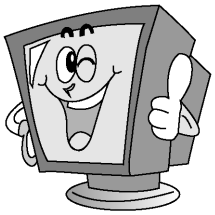

第 3 章　“登录”模块的布局

教学目标

- ✧ 了解 Activity 对应的 UI 布局创建过程。
- ✧ 掌握 LinearLayout 的常用属性及其使用方法。
- ✧ 掌握 EditText 的常用属性及其使用方法。
- ✧ 掌握 Android 图片不同分辨率的适配。
- ✧ 掌握 res/values 文件夹下各类资源文件的使用方法。
- ✧ 掌握 shape 的使用方法。
- ✧ 掌握 selector 的使用方法。

3.1　工作任务概述

本章工作任务主要完成“良心食品”App 登录界面的 UI 布局，具体 UI 效果如图 3-1 所示。

图 3-1　“良心食品”App 登录界面

3.2 预 备 知 识

3.2.1 View 与 ViewGroup 布局

Android 的 UI 界面都是由 View 和 ViewGroup 及其派生类组合而成的。其中，View 是所有 UI 组件的基类；而 ViewGroup 是容纳这些 UI 组件的容器，其本身也是由 View 派生的。

View 是 Android 平台用户 UI 界面的最基础单元。View 类为其 widgets（工具）子类奠定了基础。View 组件是可见的视觉组件，在其内部不能再置入其他组件。

ViewGroup 类为其 Layouts（布局）子类奠定了基础。ViewGroup 组件是不可见的容器组件，用来设置其容器的 View 组件和 ViewGroup 组件的排列规则。View 与 ViewGroup 的关系如图 3-2 所示，一般在 Android Studio 中可以通过 Component Tree 视图查看它们之间的树形结构。

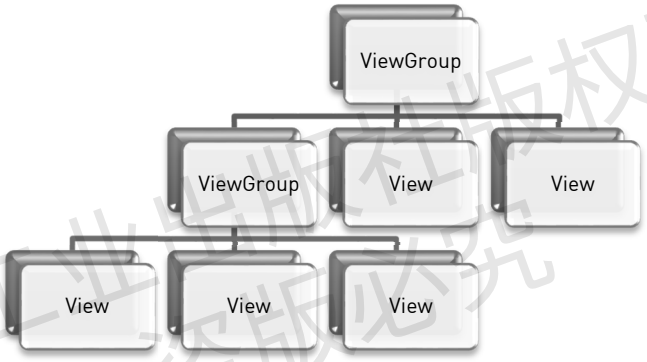

图 3-2 View 与 ViewGroup 的关系

3.2.2 LinearLayout

1. LinearLayout 简介

LinearLayout 是线性布局控件，其包含的子控件以横向或竖向的方式排列，并按照相对位置来排列所有的 widgets 或其他的 containers，当超过边界时，某些控件将缺失或消失。因此，一个垂直列表的每一行只有一个 widget 或 container（两者的宽度不限）；一个水平列表只有一个行高（高度为最高子控件的高度加上边框高度）。LinearLayout 保持其所包含的 widget 或 container 之间的间隔及相互对齐方式（相对于一个控件的右对齐、中间对齐或左对齐）。

2. 线性布局常用属性及其作用（见表 3-1）

表 3-1 线性布局常用属性及其作用

属　　性	作　　用
android:contentDescription	定义简要描述视图内容的文本

续表

属　　性	作　　用
android:layout_width; android:layout_height	这两个属性可以简单理解为View的宽与高，它们的值选项中的match_parent、wrap_content、fill_parent代表此View在父View中的宽与高的确定方式。match_parent、fill_parent代表此View的宽（或高）和父View的宽（或高）相等，wrap_content代表此View的宽高值会按照包裹自身内容的方式来确定
android:orientation	设置其内容的对齐方向（vertical表示垂直线性布局，horizontal表示水平线性布局）
android:gravity	指定该对象中放置内容的对齐方式
android:layout_gravity	相对于它的父元素的对齐方式
android:layout_weight	通过设置控件的layout_weight属性控制各个控件在布局中的相对大小。线性布局会根据该控件的layout_weight属性值与其所处布局中所有控件的layout_weight属性值之和的比值为该控件分配占用的区域

3.2.3　Android 中控件的 margin 属性和 padding 属性

Android 属性中的 margin 属性和 padding 属性是布局中比较常用的两个属性，这两个属性主要用来设置边距。

margin 属性：设置控件距离其父控件或兄弟控件的边距。

padding 属性：设置控件距离其子控件或其内部的内容（如文本）的边距。

margin 属性与 padding 属性示意图如图 3-3 所示。若以控件 B 为主，设置控件 B 的 margin 属性和 padding 属性，则控件 A 是控件 B 的父控件，控件 C（或内容 C）是控件 B 的子控件或内部内容。控件 B 的 margin 属性设置的是控件 B 与控件 A 之间的距离；控件 B 的 padding 属性设置的是控件 B 与控件 C（或内容 C）之间的距离。

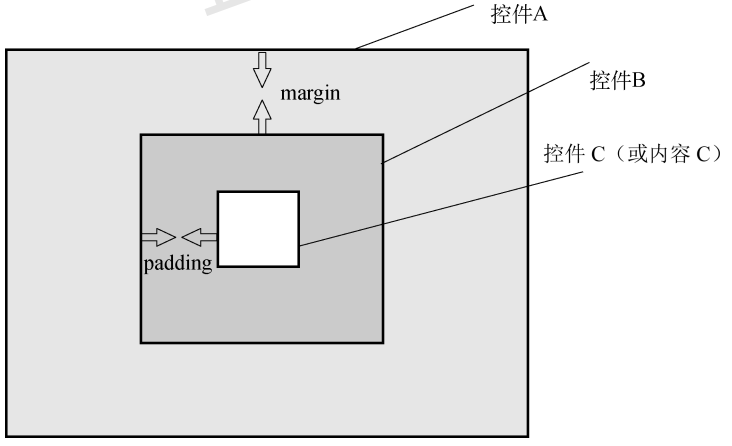

图 3-3　margin 属性与 padding 属性示意图

3.2.4　EditText 组件

在 Android 中，EditText（编辑框）用于在屏幕上显示文本输入框，在其中可以输入单行

文本，也可以输入多行文本，还可以输入指定格式的文本（如密码、电话号码和邮箱等）。EditText 组件的常用属性及其作用如表 3-2 所示。

表 3-2 EditText 组件的常用属性及其作用

属　性	作　用
android:hint	设置显示在编辑框中的提示信息
android:numeric	设置编辑框中输入的数据类型：integer（正整数）、signed（带符号整数，有正负之分）和 decimal（浮点数）
android:singleLine	设置是否单行输入，一旦设置为 true，则文字不会自动换行
android:password	设置文本是否以密码形式显示
android:textColor	设置文字颜色
android:textStyle	设置文字样式：bold、italic、bolditalic
android:textSize	设置文字大小
android:textColorHighlight	设置被选中文字的底色，默认为蓝色
android:textColorHint	设置提示信息文字的颜色，默认为灰色
android:textScaleX	设置字间距
android:typeface	设置字型：normal、sans、serif、monospace
android:background	设置背景
android:layout_weight	设置权重
android:drawableBottom	在文字的下方输出一个 drawable，如图片
android:drawableLeft	在文字的左边输出一个 drawable，如图片
android:drawableRight	在文字的右边输出一个 drawable，如图片
android:drawableTop	在文字的上方输出一个 drawable，如图片
android:drawablePadding	设置 text 与 drawable 的间隔。该属性与 drawableLeft、drawableRight、drawableTop、drawableBottom 结合使用可设置为负数，单独使用时没有效果
android:editable	设置是否可编辑
Android:maxlength	设置编辑框的最大可输入字符数

3.2.5 Android 图片不同分辨率的适配

1. 尺寸概念

（1）px（pixels）：像素，屏幕上的点，不同设备显示效果相同。例如，HVGA 表示 320px×480px。

（2）in：英寸，屏幕的物理尺寸，1in=2.54cm。例如，手机的屏幕大小为 5in、4in，这些尺寸是屏幕的对角线长度，如 4in 表示手机屏幕（可视区域）的对角线长度是 4×2.54=10.16cm。

（3）pt（point）：标准长度单位，1pt=1/72in，用于印刷业，iOS 字体单位，Android 项目开发不涉及。

（4）dpi（dots per inch）：打印分辨率，每英寸所能打印的点数（每英寸包含的像素数，即打印精度。例如，对于分辨率为 320px×480px、宽为 2in、高为 3in 的手机，其屏幕每英寸包含的像素的数量为 320/2=160dpi（横向）或 480/3=160dpi（纵向），160 就是这部手机的 dpi，横向和纵向的这个值都是相同的（因为大部分手机屏幕使用正方形的像素点）。

（5）ppi（pixels per inch）：图像分辨率、像素密度，指图像每英寸所包含的像素数。

（6）density：屏幕密度。density 和 dpi 的关系为 1density=dpi/160。

（7）dp（dip，device independent pixels）：设备独立像素，是 Android 特有的单位，与密度无关，是基于屏幕密度的抽象单位。对于分辨率为 320px×480px 同时 dpi 为 160 的显示器，1dp=1px。

（8）sp（scaled pixels）：放大像素，与刻度无关，是文字大小单位，可以根据用户的文字大小首选项进行缩放。sp 也是 Android 特有的单位。由 TextView 的源码可知，Android 默认使用 sp 作为文字大小单位。以 160ppi 屏幕为标准，当字体大小为 100%时，1sp=1px。

2. 换算关系

（1）px = dp×（dpi / 160）。

小贴士

用 sp 和 dp 代替 px 的原因是它们不会随 ppi 的变化而变化，在物理尺寸相同、ppi/dpi 不同的情况下，它们呈现的高度是相同的，也就是说，sp 和 dp 更接近物理呈现，而 px 则不行。

（2）ppi=$\sqrt{(\text{长度像素数}^2+\text{宽度像素数}^2)}$/屏幕对角线英寸数。

3. 区分标准

Google 官方指定的 dpi 区分标准如表 3-3 所示。

表 3-3　Google 官方指定的 dpi 区分标准

名　　称	ppi范围	dpi范围	图片icon尺寸
drawable-ldpi	120～160	0～120	36×36
drawable-mdpi	160～240	120～160	48×48
drawable-hdpi	240～320	160～240	72×72
drawable-xhdpi	320～480	240～320	96×96
drawable-xxhdpi	480～640	320～480	144×144
drawable-xxxhdpi	640～800	480～640	192×192

下面举例说明 Android 手机如何找到与之适配的图片。例如，某款手机配置为 1080px×1920px 和 400dpi，则对应 drawable-xxhdpi 文件夹，Android 会自动优先在 drawable-xxhdpi 文

件夹中寻找对应的图片。如果找到对应图片则加载，此时图片在手机屏幕上显示的就是其本身的大小；如果未找到，Android 会到更高分辨率的 drawable-xxxhdpi 文件夹中寻找，若一直寻找到最高分辨率的文件夹也没有的话，就开始由高到低依次查找低分辨率的文件夹，即从 drawable- xhdpi 文件夹一直查找到 drawable-ldpi 文件夹。

3.2.6 res/values 文件夹下常用的 XML 资源文件

在所有 XML 资源文件的目录设置中，最常使用的文件夹是 res\values，在此文件夹中一般会创建 string.xml、color.xml、dimens.xml、styles.xml 四种类型的 XML 资源文件。

1. string.xml（文字资源文件）

为了体现国际化及减小 App 的体积，降低数据的冗余，在 Android 开发中会把应用程序中出现的文字单独存放在 string.xml 中。作为 Android 应用开发人员，一定要养成良好的编程习惯。

（1）在 string.xml 文件中添加字符串，具体代码如下。

```
1. <?xml version="1.0" encoding="utf-8"？ >
2. <resources>
3.       <string name="hello">Hello World，MainActivity！ </string>
4.       <string name="app_name">TestExample01</string>
5. </resources>
```

（2）在 Java 源代码中使用 getString(R.string.app_name)。

（3）在 UI 布局文件中使用 android:text="@string/ app_name"。

2. colors.xml（颜色资源文件）

color.xml 文件中主要设置应用程序中所需的颜色。Android 的文字颜色定义方式采用类似网页格式的颜色定义方式，即常见的十六进制法。颜色设置语法表如表 3-4 所示。

表 3-4 颜色设置语法表

颜 色 语 法	语 法 帮 助	示范（采用十六进制）	颜 色
#RGB	无Alpha，8位表示法	#00f	蓝色
#ARGB	有Alpha，8位表示法	#800f	半透明蓝色
＃RRGGBB	无Alpha，16位表示法	#0000ff	蓝色
＃AARRGGBB	有Alpha，16位表示法	#800000ff	半透明蓝色

（1）在 color.xml 文件中添加颜色配置信息，具体代码如下。

```
1.  <?xml version="1.0" encoding="utf-8"?>
2.  <resources>
3.          <drawablename="red">#f00</drawable>
4.          <drawablename="green">#0f0</drawable>
5.          <drawablename="gray">#ccc</drawable>
6.  </resources>
```

（2）在 Java 源代码中使用 getResources().getColor(R.color.green)。

（3）在 UI 布局文件中使用 android:textColor="@color/greeen"。

3. dimens.xml（尺寸资源文件）

dimens.xml 可用于设置组件的大小及文字大小，它提供了如表 3-5 所示的几种尺寸定义方式。

表 3-5　尺寸定义表

尺寸格式	帮　助	描　述
px	pixel	以像素为单位
in	inches	以英寸为单位
mm	millimeter	以毫米为单位
pt	points	1pt=1/72英寸
dp或dip	density independent pixels	1dp=1/60英寸
sp	scale pixels	通常用于指定字体的大小，当用户修改手机显示的字体时，字体大小会随之改变

（1）在 dimens.xml 中添加尺寸配置信息，具体代码如下。

```
1.  <?xml version="1.0" encoding="utf-8"?>
2.  <resources>
3.      <dimen name="btn_width">30mm</dimen>
4.  </resources>
```

（2）在 Java 源代码中使用 getResources().getDimension(R.dimen. btn_width)。

（3）在 UI 布局文件中使用 android:layout_width="@dimen/ btn_width"。

4. styles.xml（主题风格资源文件）

styles.xml 类似于网站的样式表文件，属于更高级的 XML 资源文件，它是一个多属性的 XML 资源文件。在 Android Studio 中，styles.xml 文件会默认产生一个名字为 AppTheme 的样式，该样式是项目程序的主题样式。

（1）在 styles.xml 中添加样式信息，具体代码如下。

```
1.  <resources>
2.      <style name="text_font">
3.              <item name="android:textColor">#05b</item>
4.              <item name="android:textSize">18sp</item>
5.              <item name="android:textStyle">bold</item>
6.      </style>
7.  </resources>
```

（2）在 Java 源代码中使用 setTheme(R.style. text_font)。

（3）在 UI 布局文件中使用 style="@style/ text_font"。

3.2.7 shape

1. shape 简介

shape 是用于定义一些形状的样式，通常用于在 Android 开发中控制控件的背景。shape 共有 6 个属性，分别是 corners、padding、size、solid、stroke、gradient。

2. 在 Android Studio 中添加 shape 的方法

（1）在 Project 视图中右击 res 文件夹，依次单击“New”→“Android Resource File”选项，新建文件，如图 3-4 所示。

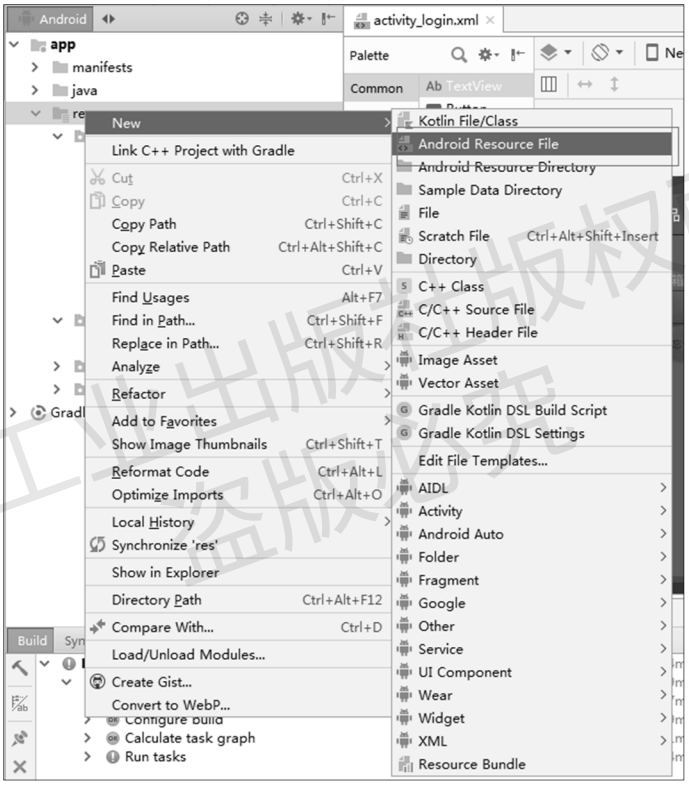

图 3-4 新建文件

（2）在打开的“New Resource File”对话框中，除了要通过“File Name”为新文件命名，还需要将“Resource type”修改为“Drawable”，并将“Root element”修改为“shape”，如图 3-5 所示。

（3）单击 OK 按钮，在项目的 res/drawable 文件夹中添加一个名为 test.xml 的 shape 文件。

（4）打开 test.xml 文件，在该文件内添加相应的属性即可。

小贴士

shape 文件存放于 drawable 文件夹中，可以把 shape 文件看成图片，在实际应用中以图片的方式应用即可。

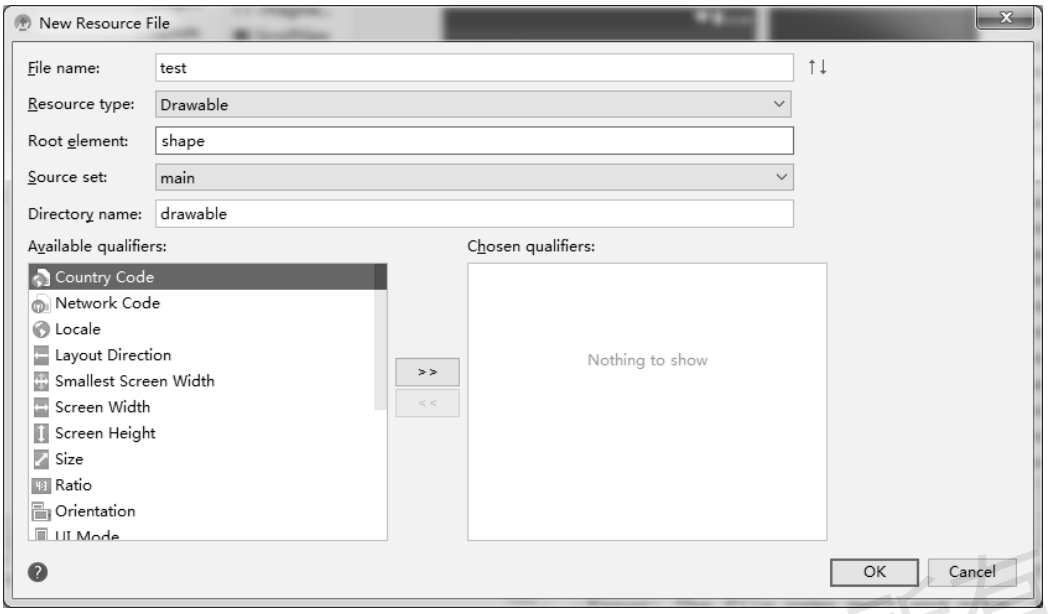

图 3-5　New Resource File 对话框

3. shape 常用属性简介

（1）corners：用于控制边框 4 个角的大小，如果默认是 0dp 的话就是直角，如果设置值大于 0dp 就会产生圆角的效果。corners 有 5 个属性，它们的作用如表 3-6 所示。

表 3-6　corners 的属性的作用

属　性	作　用
android:radius	设置4个角的圆角大小
android:topLeftRadius	设置左上角的圆角大小
android:topRightRadius	设置右上角的圆角大小
android:bottomLeftRadius	设置左下角的圆角大小
android:bottomRightRadius	设置右下角的圆角大小

corners 案例 1：

```
<corners android:radius="10dp"/>
```

corners 案例 1 效果如图 3-6 所示。

图 3-6　corners 案例 1 效果

corners 案例 2：

```
1.  <corners
2.      android:bottomRightRadius="10dp"
```

```
3.      android:topLeftRadius="10dp"/>
```

corners 案例 2 效果如图 3-7 所示。

图 3-7　corners 案例 2 效果

（2）padding：用于控制背景边框与背景中内容的距离，即用于控制内边距。padding 有 4 个属性，它们的作用如表 3-7 所示。

表 3-7　padding 的属性的作用

属　　性	作　　用
android:left	设置左内边距
android:right	设置右内边距
android:top	设置上内边距
android:bottom	设置下内边距

padding 案例：

```
1.  <padding
2.      android:top="20dp"
3.      android:bottom="20dp"
4.      android:left="40dp"
5.      android:right="40dp"/>
```

添加 padding 前的效果如图 3-8 所示；添加 padding 后的效果如图 3-9 所示。

图 3-8　添加 padding 前的效果

图 3-9　添加 padding 后的效果

（3）size：用于设置背景的大小，有 android:height 和 android:width 两个属性，这两个属性不能设置为 match_parent 或 wrap_content，只能设置为具体的数值。另外，如果这两个属性设置得过小以至于比背景上的控件还要小的时候，系统不会以设置的数值为准，而会以包裹住控件的最小的高和宽来作为背景的高和宽；如果设置得较大，则会以设置的数值为准。size 的两个属性的作用如下。

① android:width：用于设置背景的宽度。

② android:height：用于设置背景的高度。

size 案例 1：

```
1.  <size
2.      android:height="200dp"
3.      android:width="200dp"/>
```

size 案例 1 效果如图 3-10 所示。

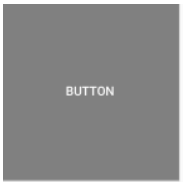

图 3-10　size 案例 1 效果

size 案例 2：

```
1.  <size
2.      android:height="2dp"
3.      android:width="2dp"/>
```

size 案例 2 效果如图 3-11 所示。

（4）solid：用于控制背景颜色，它只有一个 android:color 属性。

solid 案例：

```
1.  <solid
2.      android:color="#FF4081"/>
```

soild 案例效果如图 3-12 所示。

图 3-11　size 案例 2 效果

图 3-12　solid 案例效果

（5）stroke：用于控制背景的边框。stroke 共有 4 个属性，它们的作用如表 3-8 所示。

表 3-8　stroke 的属性的作用

属　　性	作　　用
android:width	用于控制边框的宽度
android:color	用于控制边框的颜色
android:dashWidth	用于控制虚线段的长度
android:dashGap	用于控制虚线之间的距离

注意观察 android:dashGap 和 android:dashWidth 控制的边框是否为虚线，如果这两个属性同时设置为正数，那么边框就是虚线，这两个属性只要有一个没有设置（或被设置）为 0dp，那边框就是实线。

stroke 案例：

```
1.  <corners
2.          android:radius="50dp"/>
3.  <size
4.          android:height="100dp"
```

```
5.          android:width="100dp"/>
6.    <solid
7.          android:color="#FF4081"/>
8.    <stroke
9.          android:width="5dp"
10.         android:color="#3F51B5"
11.         android:dashWidth="20dp"
12.         android:dashGap="10dp"/>
```

stroke 案例效果如图 3-13 所示。

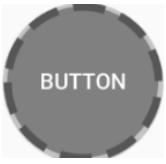

图 3-13　stroke 案例效果

（6）gradient：用于设置背景色的效果，一旦设置了这个属性，solid 中设置的背景颜色就不再生效。gradient 有 9 个属性，它们的作用如表 3-9 所示。当 android:type 的值不同时，有些属性不生效。

表 3-9　gradient 的属性的作用

属　　性	作　　用
android:type	指定渐变的类型，共有3种类型：linear（线性、默认）、radial（从中间往外扩散）、sweep（旋转扫一周）
android:startColor	设置渐变开始的颜色
android:endColor	设置渐变结束的颜色
android:centerColor	设置渐变中间的颜色
android:Angle	设置线性渐变的方向，默认从左向右。如果需要设置的话，则值应为整数并且要能被45整除，否则会报错。正数是逆时针，负数是顺时针。也就是说，angle=90是从下往上渐变，angle=−90是从上往下渐变，其他渐变无效果
android:centerX	在渐变类型为radial时，用于控制渐变圆圈中心点与左边框的距离
android:centerY	在渐变类型为radial时，用于控制渐变圆圈中心点与上边框的距离
android:gradientRadius	只有在渐变类型为radial时才生效，用于控制渐变圆圈的大小。在渐变类型为radial时，这个属性必须设置，否则会报错
android:useLevel	这个属性有两个值：true和false。当设置为true时，对应shape文件会被作为LevelListDrawable处理；一般设置为false。默认设置为false

gradient 案例：

```
1.  <gradient
2.      android:startColor="@color/white"
```

```
3.      android:endColor="@color/black"
4.      android:centerColor="#3F51B5"
5.      android:type="linear"
6.      android:useLevel="false"/>
```

gradient 案例效果如图 3-14 所示。

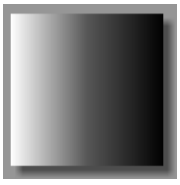

图 3-14　gradient 案例效果

3.2.8　selector

1. selector 简介

selector（选择器）在 Android 中通常用作组件的背景，这样可以省去用代码实现组件在不同状态下的不同背景颜色或不同图片的变换，使用十分方便。

2. 在 Android Studio 中添加 selector 的方法

（1）在 Project 视图中右击 res 文件夹，依次单击“New”→“Android Resource File”选项，新建文件，如图 3-15 所示。

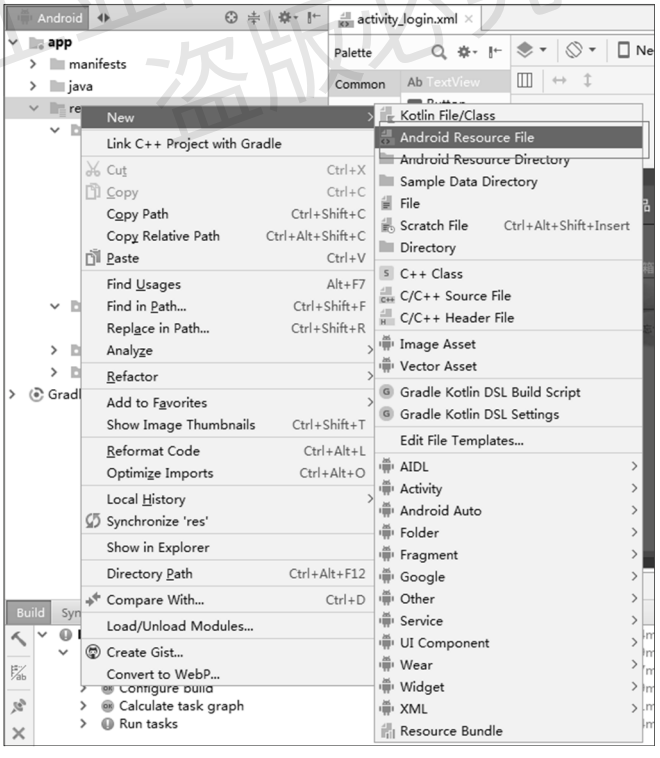

图 3-15　新建文件

（2）在打开的“New Resource File”对话框中，除了要通过“File Name”为新文件命名，还需要将“Resource type”修改为“Drawable”，将“Root element”修改为“selector”，如图 3-16 所示。

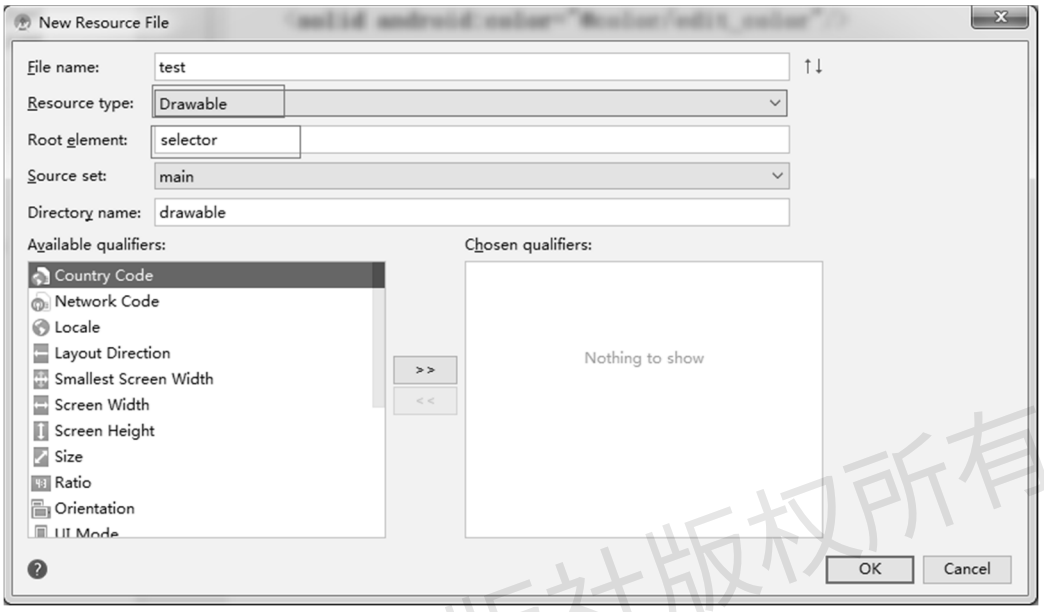

图 3-16 “New Resource File”对话框

（3）单击“OK”按钮，在项目程序的 res/drawable 文件夹中添加一个名为 test.xml 的 selector 文件。

（4）打开 test.xml 文件，在该文件中添加相应的属性即可。

3. selector 常用属性（见表 3-10）

表 3-10 selector 的属性的作用

属　性	作　用
android:color="hex_color"	设置颜色值：#RGB、$ARGB、#RRGGBB、#AARRGGBB
android:drawable="@［package:］drawable/drawable_resource"	设置图片资源
android:state_pressed=［"true"｜"false"］	设置是否触摸
android:state_focused=［"true"｜"false"］	设置是否获取焦点
android:state_hovered=［"true"｜"false"］	设置光标是否经过
android:state_selected=［"true"｜"false"］	设置是否选中
android:state_checkable=［"true"｜"false"］	设置是否可勾选
android:state_checked=［"true"｜"false"］	设置是否勾选
android:state_enabled=［"true"｜"false"］	设置是否可用

续表

属　　性	作　　用
android:state_activated=［"true"	"false"］	设置是否激活
android:state_window_focused=［"true"|"false"］	设置所在窗口是否获取焦点

4. 使用 selector 的方法

方法一：在组件中配置 android:listSelector="@drawable/xxx"，或者添加属性 android:background="@drawable/xxx"。

方法二：Drawable drawable = getResources().getDrawable(R.drawable.xxx);

组件.setSelector(drawable);

3.3　热 身 任 务

本节热身任务为微信中的“我”。

1. 任务说明

完成如图 3-17 所示的布局效果。

2. 操作步骤

（1）新建项目。

（2）将项目图片复制到项目的 res/mipmap 文件夹中。在 weixin.xml 布局文件中添加组件，产生如图 3-18 所示的效果。微信中的“我”的 Component Tree 如图 3-19 所示。

图 3-17　微信中的“我”效果图

图 3-18　初始布局

图 3-19 微信中的“我”的 Component Tree

（3）修改 activity_main.xml 文件中相应组件的属性，修改完成后的 activity_main.xml 文件的代码及相应功能说明如下。

```
<?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"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/linearLayout"
    android:layout_width="match_parent"//设置组件的宽度
    android:layout_height="match_parent"//设置组件的高度
    android:orientation="vertical"//设置线性布局为垂直线性布局
    android:background="#ebebeb"//设置背景颜色
    tools:context=".MainActivity">
    <android.support.constraint.ConstraintLayout
        android:layout_width="match_parent"
        android:layout_height="71dp"
        android:background="#ffffff">
        <ImageView
```

```
16.                 android:id="@+id/imageView2"
17.                 android:layout_width="49dp"
18.                 android:layout_height="55dp"
19.                 android:layout_marginBottom="8dp"//设置组件与下边组件的距离
20.                 android:layout_marginEnd="8dp"//设置组件与右边组件的距离
21.                 android:layout_marginStart="15dp"//设置组件与左边组件的距离
22.                 android:layout_marginTop="8dp" //设置组件与上边组件的距离
23.                 app:layout_constraintBottom_toBottomOf="parent"//设置与父窗体底端对齐
24.                 app:layout_constraintEnd_toEndOf="parent"
25.                 app:layout_constraintHorizontal_bias="0.024"//水平方向偏移量
26.                 app:layout_constraintStart_toStartOf="parent"
27.                 app:layout_constraintTop_toTopOf="parent"
28.                 app:srcCompat="@mipmap/userphoto" />//设置 ImageView 的图像源为 userphoto
29.             <TextView
30.                 android:id="@+id/textView"
31.                 android:layout_width="wrap_content"
32.                 android:layout_height="wrap_content"
33.                 android:layout_marginLeft="10dp"
34.                 android:text="妲己"//设置文本内容为“妲己”
35.                 app:layout_constraintLeft_toRightOf="@id/imageView2"
36.                 app:layout_constraintTop_toTopOf="@id/imageView2" />
37.             <TextView
38.                 android:id="@+id/textView2"
39.                 android:layout_width="wrap_content"
40.                 android:layout_height="wrap_content"
41.                 android:layout_marginLeft="10dp"
42.                 android:text="微信号：oh my god"
43.                 app:layout_constraintBottom_toBottomOf="@id/imageView2"
44.                 app:layout_constraintLeft_toRightOf="@id/imageView2"/>
45.             <ImageView
46.                 android:id="@+id/imageView3"
47.                 android:layout_width="56dp"
48.                 android:layout_height="32dp"
49.                 android:layout_marginBottom="8dp"
50.                 android:layout_marginEnd="8dp"
51.                 android:layout_marginStart="8dp"
52.                 android:layout_marginTop="8dp"
53.                 app:layout_constraintBottom_toBottomOf="parent"
54.                 app:layout_constraintEnd_toEndOf="parent"
55.                 app:layout_constraintHorizontal_bias="1.0"
56.                 app:layout_constraintStart_toStartOf="parent"
57.                 app:layout_constraintTop_toTopOf="parent"
58.                 app:layout_constraintVertical_bias="0.478"//垂直方向偏移量
59.                 app:srcCompat="@mipmap/erwanma" />
60.         </android.support.constraint.ConstraintLayout>
61.         <LinearLayout
62.             android:layout_width="match_parent"
```

```
63.             android:layout_height="40dp"
64.             android:layout_marginTop="30dp"
65.             android:background="#ffffff"
66.             android:gravity="center_vertical"//设置组件内部元素垂直居中对齐
67.             android:orientation="horizontal">
68.             <ImageView
69.                 android:id="@+id/imageView4"
70.                 android:layout_width="20dp"
71.                 android:layout_height="20dp"
72.                 android:layout_weight="1"
73.                 app:srcCompat="@mipmap/icon1" />
74.             <TextView
75.                 android:id="@+id/textView3"
76.                 android:layout_width="wrap_content"
77.                 android:layout_height="wrap_content"
78.                 android:layout_weight="4"
79.                 android:text="钱包" />
80.         </LinearLayout>
81.         <LinearLayout
82.             android:layout_width="match_parent"
83.             android:layout_height="40dp"
84.             android:layout_marginTop="30dp"
85.             android:background="#ffffff"
86.             android:gravity="center_vertical"
87.             android:orientation="horizontal">
88.             <ImageView
89.                 android:id="@+id/imageView5"
90.                 android:layout_width="20dp"
91.                 android:layout_height="20dp"
92.                 android:layout_weight="1"
93.                 app:srcCompat="@mipmap/icon2" />
94.             <TextView
95.                 android:id="@+id/textView4"
96.                 android:layout_width="wrap_content"
97.                 android:layout_height="wrap_content"
98.                 android:layout_weight="4"
99.                 android:text="收藏 " />
100.        </LinearLayout>
101.        <LinearLayout
102.            android:layout_width="match_parent"
103.            android:layout_height="40dp"
104.            android:layout_marginTop="2dp"
105.            android:background="#ffffff"
106.            android:gravity="center_vertical"
107.            android:orientation="horizontal">
108.            <ImageView
109.                android:id="@+id/imageView6"
```

```
110.                android:layout_width="20dp"
111.                android:layout_height="20dp"
112.                android:layout_weight="1"
113.                app:srcCompat="@mipmap/icon3" />
114.            <TextView
115.                android:id="@+id/textView5"
116.                android:layout_width="wrap_content"
117.                android:layout_height="wrap_content"
118.                android:layout_weight="4"
119.                android:text="相册" />
120.        </LinearLayout>
121.        <LinearLayout
122.            android:layout_width="match_parent"
123.            android:layout_height="40dp"
124.            android:layout_marginTop="2dp"
125.            android:background="#ffffff"
126.            android:gravity="center_vertical"
127.            android:orientation="horizontal">
128.            <ImageView
129.                android:id="@+id/imageView7"
130.                android:layout_width="20dp"
131.                android:layout_height="20dp"
132.                android:layout_weight="1"
133.                app:srcCompat="@mipmap/icon4" />
134.            <TextView
135.                android:id="@+id/textView6"
136.                android:layout_width="wrap_content"
137.                android:layout_height="wrap_content"
138.                android:layout_weight="4"
139.                android:text="卡包" />
140.        </LinearLayout>
141.        <LinearLayout
142.            android:layout_width="match_parent"
143.            android:layout_height="40dp"
144.            android:layout_marginTop="2dp"
145.            android:background="#ffffff"
146.            android:gravity="center_vertical"
147.            android:orientation="horizontal">
148.            <ImageView
149.                android:id="@+id/imageView8"
150.                android:layout_width="20dp"
151.                android:layout_height="20dp"
152.                android:layout_weight="1"
153.                app:srcCompat="@mipmap/icon5" />
154.            <TextView
155.                android:id="@+id/textView7"
156.                android:layout_width="wrap_content"
```

```
157.                android:layout_height="wrap_content"
158.                android:layout_weight="4"
159.                android:text="表情" />
160.        </LinearLayout>
161.        <LinearLayout
162.            android:layout_width="match_parent"
163.            android:layout_height="40dp"
164.            android:layout_marginTop="30dp"
165.            android:background="#ffffff"
166.            android:gravity="center_vertical"
167.            android:orientation="horizontal">
168.            <ImageView
169.                android:id="@+id/imageView9"
170.                android:layout_width="20dp"
171.                android:layout_height="20dp"
172.                android:layout_weight="1"
173.                app:srcCompat="@mipmap/icon6" />
174.            <TextView
175.                android:id="@+id/textView8"
176.                android:layout_width="wrap_content"
177.                android:layout_height="wrap_content"
178.                android:layout_weight="4"
179.                android:text="设置" />
180.        </LinearLayout>
181. </LinearLayout>
```

思考

还有其他方法可以实现与微信中的“我”相同的布局吗？

上述方法中的代码还可以简化吗？

3.4 实现“登录”模块的布局

1. 知识点

- LinearLayout 的使用方法。
- Android 图片不同分辨率的适配。
- EditText 的使用方法。
- shape 的使用方法。
- selector 的使用方法。
- res/values 文件夹下各类资源文件的使用方法。

2. 工作任务

制作“良心食品”App 登录模块的 UI 布局，实现如图 3-20 所示的 UI 效果。

图 3-20 “登录”界面

3. 操作流程

（1）新建项目，项目名称为 HealthFood。

（2）将项目用到的所有图片分类复制到 Drawable 和 mipmap 两个文件夹中（注意要根据 Android 图片不同分辨率适配原则分别存放图片）。

（3）右击项目中的 res/layout/activity_main.xml 文件，依次单击“refactor”→“Rename”选项，将文件名改为 activity_login.xml。

（4）在 activity_login.xml 文件中添加组件，为了避免在一个项目中出现 ID 重名问题，建议添加组件时在每个组件的默认 ID 名前面都添加一个“Login_”，完成后的登录界面 UI 效果图如图 3-21 所示，其 Component Tree 如图 3-22 所示。

图 3-21 完成后的“登录”界面 UI 效果图

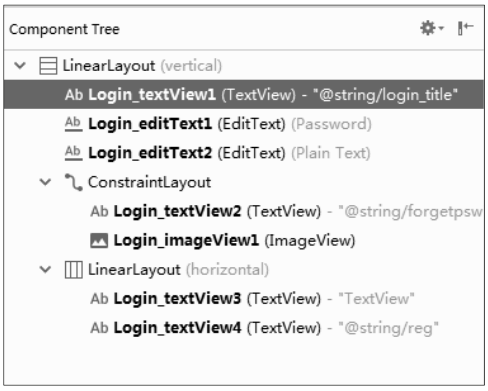

图 3-22 “登录”界面的 Component Tree

（5）选中 Component Tree 中最外层的 LinearLayout 组件，通过修改其 background 属性将图片 login_bg.jpg 设置为整个布局的背景。

（6）打开项目 res/values 文件夹下的 strings.xml 文件，修改该文件的 app_name 属性，并添加 login_title 等多个字符串属性，具体代码如下。

```
1.  <?xml version="1.0" encoding="utf-8"?>
2.  <resources>
3.      <string name="app_name">良心食品</string>//
4.      <string name="login_title">登入</string>
5.      <string name="username">手机号/邮箱</string>
6.      <string name="password">密 码</string>
7.      <string name="forgetpsw">忘记密码？点击找回</string>
8.      <string name="nouser">还没有账号？ </string>
9.      <string name="reg">注册</string>
10. </resources>
```

第 3 行代码用于修改 App 标题。该行代码默认通过 AndroidManifest.xml 文件中的 android:label="@string/app_name"语句实现对 App 标题的修改。

（7）通过属性面板将组件的 text 属性设置为在步骤（6）中添加的字符串。例如，将 Login_textView1 组件的 text 属性设置为@string/login_title，完成修改后的布局效果如图 3-23 所示。

图 3-23 修改组件字符信息

（8）右击项目中的 res/values 文件夹，依次单击“New”→“XML”→“Values XML File”选项，新建 dimens.xml 文件，并在该文件中添加 login_textsize 属性，具体代码如下。

```
<resources>
    <!-- Default screen margins, per the Android Design guidelines. -->
        <dimen name="login_textsize">30sp</dimen>//此尺寸定义用于设置“登入”的字体大小
        <dimen name="edit_dimens">10dp</dimen>//此尺寸定义用于设置文本编辑框圆角的大小
</resources>
```

（9）通过属性面板将在步骤（8）中添加的尺寸应用于 TextView 的 TextSize 属性，实现将“登入”文字大小修改为 30sp 的功能，效果如图 3-24 所示。

图 3-24　修改组件尺寸信息

（10）打开项目 res/values 文件夹下的 colors.xml 文件，并在该文件中添加 edit_color 等属性，具体代码如下。

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <color name="edit_color">#40ffffff</color>//此颜色定义用于设置编辑框中文字的颜色
    <color name="hint_textColor">#65ffffff</color>//此颜色定义用于设置编辑框中提示文字的颜色
    <color name="font_color">#ffffff</color>//此颜色定义用于设置白色文字
</resources>
```

（11）在 Project 视图中右击 src 文件夹，依次单击“New”→“Android Resource File”选项，新建一个名字为 edit_login_shape_t.xml 的 shape 文件，并在该文件中添加相应属性。此处的 shape 文件用于设置 Login_editText1 及 Login_editText2 两个文本编辑框获取焦点后的背景效果，具体代码如下。

```
<?xml version="1.0" encoding="utf-8"?>
<shape xmlns:android="http://schemas.android.com/apk/res/android" >
    //设置 shape 圆角为 Dimens 中定义的 edit_dimens
    <corners android:radius="@dimen/edit_dimens"></corners>
    <solid android:color="@color/hint_textColor"/>//设置 shape 描边颜色为 Colors 中定义的 hint_textColor
</shape>
```

（12）重复步骤（11）的操作，新建一个名字为 edit_login_shape_f.xml 的 shape 文件，并在该文件中添加相应属性。此处的 shape 文件用于设置 Login_editText1 及 Login_editText2 两

个文本编辑框失去焦点后的背景效果，具体代码如下。

```
<?xml version="1.0" encoding="utf-8"?>
<shape xmlns:android="http://schemas.android.com/apk/res/android" >
    <corners android:radius="@dimen/edit_dimens" ></corners>
    <solid android:color="@color/edit_color"/>
</shape>
```

（13）在 Project 视图中右击 src 文件夹，依次单击“New”→“Android Resource File”选项，新建一个名为 edituser_selector 的 selector 文件，并在该文件中添加相应属性。此处的 selector 文件主要用于完成当 Login_editText1 及 Login_editText2 两个文本编辑框获取焦点或失去焦点时自动切换背景的功能，具体代码如下。

```
<?xml version="1.0" encoding="utf-8"?>
<selector xmlns:android="http://schemas.android.com/apk/res/android" >
    <item android:state_focused="true"
        //将组件获取焦点后的图片设置为 edit_login_shape_t.xml
        android:drawable="@drawable/edit_login_shape_t "></item>
     <item android:state_focused="false"
        //将组件失去焦点后的图片设置为 edit_login_shape_f.xml
        android:drawable="@drawable/edit_login_shape_f "></item>
</selector>
```

（14）打开项目 res/values 文件夹下的 style.xml 文件，并在该文件中添加 login_et 样式，具体代码如下。

```
<style name="login_et">
    <item name="android:layout_width">match_parent</item>
    //设置组件的宽度与父窗体对齐
    <item name="android:layout_height">45dp</item>
    //设置组件的高度为 45dp
    <item name="android:textColor">#ffffff</item>
    //设置文字颜色为白色
    <item name="android:background">@drawable/edituser_selector</item>
    //设置背景为 edituser_selector
    <item name="android:layout_marginLeft">30dp</item>
    //设置与左边元素边缘的距离为 30dp
    <item name="android:layout_marginRight">30dp</item>
    //设置与右边元素边缘的距离为 30dp
    <item name="android:maxLength">20</item>
    //设置最多可输入 20 个字符
    <item name="android:textColorHint">@color/hint_textColor</item>
    //设置提示文字的颜色为颜色资源文件中定义的 hint_textColor
    <item name="android:textSize">@dimen/login_textsize</item>
    //设置文字大小为尺寸资源文件中定义的 login_textsize
    <item name="android:layout_gravity">center_horizontal</item>
    //定义组件相对于父窗体水平居中对齐
    <item name="android:drawablePadding">8dp</item>
    //设置 text 与 drawable（图片）的间隔为 8dp
    <item name="android:singleLine">true</item>
    //设置为单行文本显示
```

```
26.     </style>
```

（15）分别在 Login_editText1 及 Login_editText2 两个文本编辑框组件中将 style 属性设置为@style/lgoin_et，实现对这两个文本编辑框样式的添加。

（16）分别在 Login_editText1 及 Login_editText2 两个文本编辑框组件中将 drawableLeft 属性设置为@drawable/user 和@drawable/pass，实现对这两个文本编辑框左内置图的添加。

（17）分别设置其他组件的文字颜色、间距等属性，最终完成布局，具体代码如下。

```
1.  <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
2.      xmlns:app="http://schemas.android.com/apk/res-auto"
3.      xmlns:tools="http://schemas.android.com/tools"
4.      android:layout_width="match_parent"
5.      android:layout_height="match_parent"
6.      android:background="@drawable/login_bg"
7.      android:orientation="vertical"
8.      tools:context=".Login">
9.      <TextView
10.         android:id="@+id/Login_textView1"
11.         android:layout_width="match_parent"
12.         android:layout_height="wrap_content"
13.         android:layout_marginLeft="30dp"
14.         android:layout_marginTop="160dp"
15.         android:text="@string/login_title"
16.         android:textColor="@color/font_color"
17.         android:textSize="@dimen/login_textsize" />
18.     <EditText
19.         android:id="@+id/Login_editText1"
20.         style="@style/Login_et"
21.         android:layout_marginTop="30dp"
22.         android:drawableLeft="@mipmap/user"//图片放在 mipmap 下，所以用@mipmap/方式引用
23.         android:ems="10"
24.         android:hint="@string/username"
25.         android:inputType="textPassword" />
26.     <EditText
27.         android:id="@+id/Login_editText2"
28.         style="@style/login_et"
29.         android:layout_marginTop="30dp"
30.         android:drawableLeft="@mipmap/pass"
31.         android:ems="10"
32.         android:hint="@string/password"
33.         android:inputType="textPersonName" />
34.     <android.support.constraint.ConstraintLayout
35.         android:layout_width="match_parent"
36.         android:layout_height="40dp"
37.         android:layout_marginTop="10dp">
38.         <TextView
39.             android:id="@+id/Login_textView2"
40.             android:layout_width="wrap_content"
41.             android:layout_height="wrap_content"
```

```
42.                 android:layout_marginEnd="8dp"
43.                 android:layout_marginStart="8dp"
44.                 android:text="@string/forgetpsw"
45.                 android:textColor="@color/font_color"
46.                 app:layout_constraintBottom_toBottomOf="parent"
47.                 app:layout_constraintEnd_toEndOf="parent"
48.                 app:layout_constraintHorizontal_bias="0.316"
49.                 app:layout_constraintStart_toStartOf="parent"
50.                 app:layout_constraintTop_toTopOf="parent"
51.                 app:layout_constraintVertical_bias="0.51" />
52.             <ImageView
53.                 android:id="@+id/Login_imageView1"
54.                 android:layout_width="wrap_content"
55.                 android:layout_height="wrap_content"
56.                 android:layout_marginBottom="8dp"
57.                 android:layout_marginEnd="8dp"
58.                 android:layout_marginStart="8dp"
59.                 android:layout_marginTop="8dp"
60.                 android:src="@mipmap/login"
61.                 app:layout_constraintBottom_toBottomOf="parent"
62.                 app:layout_constraintEnd_toEndOf="parent"
63.                 app:layout_constraintHorizontal_bias="0.916"
64.                 app:layout_constraintStart_toStartOf="parent"
65.                 app:layout_constraintTop_toTopOf="parent"
66.                 app:layout_constraintVertical_bias="0.523" />
67.         </android.support.constraint.ConstraintLayout>
68.         <LinearLayout
69.             android:layout_width="match_parent"
70.             android:layout_height="match_parent"
71.             android:gravity="bottom|center_horizontal"
72.             android:orientation="horizontal"
73.             android:paddingBottom="12dp">
74.             <TextView
75.                 android:id="@+id/Login_textView3"
76.                 android:layout_width="wrap_content"
77.                 android:layout_height="wrap_content"
78.                 android:text="@string/nouser"
79.                 android:textColor="#bbffffff"/>
80.             <TextView
81.                 android:id="@+id/Login_textView4"
82.                 android:layout_width="wrap_content"
83.                 android:layout_height="wrap_content"
84.                 android:text="@string/reg"
85.                 android:textColor="@color/font_color" />
86.         </LinearLayout>
87. </LinearLayout>
```