程/序/员/软/件/开/发/书/库

和外星人一起学
训练·应用

明日科技◎编著

Android

精彩编程200例

（全彩版）

精选 220 个实例，快学快用

- 实例丰富，涵盖广泛——精选 220 个实例，贴近实际开发应用
- 书网合一，同步学习——免费在线课程，PC 端和移动端同步学习
- 全彩印刷，重点突出——还原真实开发环境，让编程学习更简单
- 技术服务完善——明日学院官网、企业 QQ、论坛全程技术支持

数 字 电 子 书

吉林大学出版社

Android 精彩编程 200 例

明日科技 编著

明日科技

吉林大学出版社

内容简介

《Android 精彩编程 200 例》汇集了 Android 开发从基础到高级应用各个层面的 220 个实例及源代码，每一实例都按实例说明、关键技术、实现过程、扩展学习的顺序进行分析解读。全书主要内容有菜单与对话框、Android 常用控件应用、数据储存操作、第三方支持、手势交互、图像处理、图表处理、摄像头、动画、多媒体、语音识别、传感器、网络编程、与手机系统交互、位置服务与地图应用、物联网开发、游戏开发、Android 5.0+ 效果、商城常用、验证码合集、启动页合集、加载等待合集、抢红包服务、App 应用开发必备模块。配书光盘附赠了实例的完整源程序及部分讲解视频。

《Android 精彩编程 200 例》既适合 Android 程序员参考，也适合高校学生、软件开发培训学员及相关求职人员学习。

图书在版编目（CIP）数据

Android 精彩编程 200 例 / 明日科技编著 . – 长春 : 吉林大学出版社，2017.8

ISBN 978-7-5692-0453-7

Ⅰ . ① A… Ⅱ . ①明… Ⅲ . ①移动终端－应用程序－程序设计 Ⅳ . ① TN929.53

中国版本图书馆 CIP 数据核字 (2017) 第 178202 号

书　　名：Android 精彩编程 200 例
Android JINGCAI BIANCHENG 200 LI

作　　者：明日科技 编著
策划编辑：殷丽爽　　责任编辑：张宏亮
责任校对：殷丽爽　　装帧设计：明日科技
出版发行：吉林大学出版社
社　　址：长春市朝阳区明德路 501 号　　邮政编码：130021
发行电话：0431-89580026/0431-84978982
网　　址：http://www.jlup.com.cn　　电子邮箱：jdcbs@.jlu.edu.cn
印　　刷：吉广控股有限公司
开　　本：850mm×1100mm　1/16　　印　　张：34
（附 DVD 光盘 2 张）
字　　数：800 千字　　印　　数：1 ～ 10000
版　　次：2017 年 8 月第 1 版　　印　　次：2017 年 8 月第 1 次印刷
书　　号：ISBN 978-7-5692-0453-7
定　　价：89.80 元

版权所有　翻印必究

前　言

程序开发是一项复杂而富于创造性的工作，它不仅需要开发人员掌握各方面的知识，还需要具备丰富的开发经验及创造性的编程思维。丰富的开发经验可以迅速提升开发人员解决实际问题的能力，从而缩短开发时间，使编程工作更为高效。

为使开发人员获得更多的经验，明日科技 Android 开发团队精心设计了 200 余个经典实例，涵盖 Android 项目开发中的核心技术，以达到丰富经验、实战中学技术的目的。

本书内容

本书分为 24 章，共计 220 个实例，书中实例均为一线开发人员精心设计，囊括了开发中经常使用和需要解决的热点、难点问题。在讲解实例时，分别从实例说明、关键技术、实现过程、扩展学习四个板块进行讲解。

利用图文结合的方式，介绍实例的特点、功能和运行效果

详细介绍了实例的开发过程，指导读者开发出自己的实例程序

剖析实例应用的核心技术，了解设计思路

介绍相关技巧、经验总结，丰富编程体验

❋ 本书特点

实例丰富，涵盖广泛

本书精选了 220 个实例，涵盖了 Android 开发从基础到高级应用等多个层面的核心技术，以便读者积累丰富的开发经验。

关键技术实用、具体

书中所选实例均是项目开发中经常使用的技术，涵盖了编程中多个方面的各种应用，由一线工程师精心编选而成，可以帮助开发人员解读该技术的实现过程。读者在开发时所需的关键技术、技巧可以通过本书查找。

书网结合，同步学习

为方便读者更好地使用本书进行学习，本书提供了强大的网络支持和服务，通过扫描二维码，可以在移动端阅读数字电子书和其他扩展内容，也可以在 PC 端阅读本书的数字电子书等配套资源，还可以登录明日学院获得技术支持。

全彩印刷，重点突出

为还原真实开发环境，本书采用了全彩印刷，可以有效提升编程学习效率，减少阅读压力，使开发过程更加轻松有趣。

可操作性强

开发人员可以参照本书中的实例，开发出自己的实例，简单易学，便于积累经验。

技术服务完善

为了帮助读者尽快理解内容，读者可通过企业 QQ、论坛、信箱、客服电话、明日学院官方网站等多种渠道获取相关问题的答复与指导。

❋ 本书特别约定

实例使用方法

用户在学习本书的过程中，可以从光盘（本书配套 2 张 DVD 光盘）中复制实例源代码，修改实例的只读属性。有些实例需要使用相应的数据库或第三方资源，这些实例在使用前需要进行相应配置。

源码位置

实例在光盘中的存储格式为“光盘 \Code\ 章号 \ 实例序号”或“光盘 \Video\ 章号 \ 实例序号”。

部分实例只给出关键代码

由于篇幅限制，书中有些实例只给出了关键代码，完整代码请参考光盘实例程序。

明日科技

使用配套数字电子书

本书不仅内容丰富，图文并茂，还有与之配套的数字电子书。读者购买本书后，刮开在线学习码涂层，扫描封底的激活二维码，注册会员后即可查看本书的配套电子书及其他的学习资源。

① 刮开封底的在线学习码　② 扫描封底的激活二维码　③ 注册明日会员　④ 查看图书相关学习资源

移动端学习本书

图书中提供视频讲解、关键技术二维码，读者可以使用手机扫描二维码查看完整代码，有效利用碎片时间，提高学习效率。

技术服务

为方便解决学习中的疑难问题，我们提供了企业服务 QQ:4006751066（可容纳 10 万人），明日学院网站 (www.mingrisoft.com) 还提供了论坛服务和配套学习服务支持。

服务网站：www.mingrisoft.com

企业 QQ ：4006751066

服务信箱：mingrisoft@mingrisoft.com

客服电话：400 675 1066，0431-84978981/84978982

致读者

本书由明日科技 Android 项目开发团队策划并组织编写，主要编写人员有贾景波、李磊、赛奎春、王小科、王国辉、冯春龙、申小琦、张鑫、赵宁、周佳星、白宏健、杨柳、葛忠月、隋妍妍、赵颖、李春林、辛洪郁、刘杰、宋万勇、张宝华、杨丽、裴莹、刘媛媛、张云凯、吕玉翠、庞凤、孙巧辰、胡冬、梁英、周艳梅、房雪坤、江玉贞、高春艳。在编写本书的过程中，我们本着科学、严谨的态度，力求精益求精，但疏漏之处在所难免，敬请广大读者批评斧正。

感谢您阅读本书，希望本书能成为您编程路上的领航者。

祝读书快乐！

编　者

2017 年 8 月

明日科技

目　录

Content

第 1 章 菜单与对话框 1

视频讲解：3 小时 51 分

实例 001 微信右上角弹出的菜单 1
实例 002 明日学院抽屉式公告 2
实例 003 仿 QQ 侧滑菜单 4
实例 004 可以收放的旋转菜单 7
实例 005 仿 58 同城的退出动画 9
实例 006 明日学院消息通知 12
实例 007 退出当前账号对话框 14
实例 008 圆盘式工具箱菜单 16
实例 009 彩虹式菜单 18
实例 010 滑动删除聊天记录 20
实例 011 带图标的消息提示弹窗 23
实例 012 使用 PopWindow 实现选择菜单 25

第 2 章 Android 常用控件应用 27

视频讲解：20 分

实例 013 三种进度条样式 27
实例 014 仿支付密码输入框 29
实例 015 带字母检索的全国城市列表 32
实例 016 下拉顶部图片放大效果 34
实例 017 可以每日签到的日历表 36
实例 018 使用时间选择器设置提醒时间 38
实例 019 弹出智能提示信息的搜索框 40
实例 020 页面中垂直滚动的公告条 42
实例 021 外星人手机时钟 43
实例 022 加载大量文字时关键字变色 45
实例 023 返回顶部按钮 47

第 3 章 数据储存操作 50

视频讲解：42 分

实例 024 使用 SharePreferences 保存用户信息 50
实例 025 保存信息到 SD 卡 52
实例 026 记事本 55
实例 027 查看手机中的图片 57
实例 028 家庭账务记录 60
实例 029 课程表 62
实例 030 查看手机 SD 卡中所有文件 63
实例 031 创建与删除文件 66
实例 032 搜索应用程序 69
实例 033 文件的复制和粘贴 72

第 4 章 第三方支持 75

实例 034 使用 Bmob 第三方后台数据 75
实例 035 第三方登录 78
实例 036 极光推送 81
实例 037 第三方分享 83
实例 038 微信支付 85

明日科技

实例 039 支付宝支付 87
实例 040 银联支付 89

第 5 章 手势交互 92
视频讲解：15 分

实例 041 仿 QQ 删除消息提示 92
实例 042 九点锁屏 94
实例 043 看着我 96
实例 044 手势打电话 99
实例 045 手势翻书 100
实例 046 图片的放大 103
实例 047 长按碎屏效果 104
实例 048 抓不到我 107
实例 049 转动的地球 109

第 6 章 图像处理 112
视频讲解：1 小时 29 分

实例 050 浏览朋友圈图片 112
实例 051 仿大街网选择图片 114
实例 052 仿微博发表动态 117
实例 053 毛玻璃效果 119
实例 054 拼接图片 121
实例 055 扭曲变形效果 123
实例 056 为图片添加特殊效果 125
实例 057 图片圆角处理 127
实例 058 滑动浏览图片 130
实例 059 ViewPager 实现画廊效果 132

第 7 章 图表处理 136
视频讲解：30 分

实例 060 书籍销量柱状统计图 136
实例 061 家庭支出分析饼状图 139
实例 062 月支出麻点统计图 141
实例 063 心电图 143
实例 064 房价走势折线图 145
实例 065 人物能力分析雷达图 147

第 8 章 摄像头 150

实例 066 调用系统相机获取图像 150
实例 067 Camera 与 SurfaceView 153
实例 068 Camera2 与 TextureView 155
实例 069 为摄像机增加滤镜效果 159
实例 070 扫描与生成二维码 161
实例 071 人脸识别 163
实例 072 识别银行卡信息 166
实例 073 相机挂件 168

第 9 章 动画 171
视频讲解：42 分

实例 074 飘落的雪花 171
实例 075 种一朵花 173
实例 076 用刷子画外星人 176
实例 077 会旋转的明日 Logo 178
实例 078 会变色的花 180
实例 079 以动画的方式浏览照片 181
实例 080 拍立得打印照片的动画效果 184
实例 081 两只会飞的鸟 186
实例 082 仿小米手机卸载应用动画效果 188
实例 083 圆形扩散效果 190
实例 084 明日学院唱片式音乐播放器 191
实例 085 点赞效果 194
实例 086 仿支付宝咻一咻 195
实例 087 数字滚动效果 198

第 10 章 多媒体 200

视频讲解：38 分

实例 088 带音效的火山爆发效果 200
实例 089 显示歌词的音乐播放器 202
实例 090 录音机 206
实例 091 分贝测试仪 209
实例 092 调用本地播放器播放视频 211
实例 093 带弹幕的视频播放器 213
实例 094 手机屏幕录制 215
实例 095 仿微信录制视频 218
实例 096 播放视频对话框 220

第 11 章 语音识别 223

视频讲解：3 小时 48 分

实例 097 会聊天的机器人 223
实例 098 语音打电话 226
实例 099 语音电子书 228
实例 100 语音发短信 230
实例 101 语音唤醒 232
实例 102 仿微信输入 235

第 12 章 传感器 238

视频讲解：6 小时 18 分

实例 103 计步器 238
实例 104 仿微信摇一摇 241
实例 105 水平仪 243
实例 106 磁场指南针 246
实例 107 光线控制页面背景 248
实例 108 不要靠近听筒 249
实例 109 话筒吹雾小游戏 252
实例 110 弹力球碰撞 254
实例 111 使用陀螺仪控制全景视图 257
实例 112 摇晃手机更换图片 259

第 13 章 网络编程 262

视频讲解：4 小时 50 分

实例 113 多线程下载腾讯新闻客户端 262
实例 114 使用 WebView 访问明日学院 264
实例 115 明日学院课程详情页面 266
实例 116 获取网络菜谱 267
实例 117 下载网络图片 270
实例 118 调用系统下载文件 272
实例 119 天气预报 274
实例 120 国内新闻 276
实例 121 从本地获取并上传图片 278

第 14 章 与手机系统交互 282

视频讲解：2 小时 36 分

实例 122 吹蜡烛锁屏 282
实例 123 短信发送状态 285
实例 124 短信作弊器 288
实例 125 获取系统相关信息 290
实例 126 风车闹钟 291
实例 127 群发拜年短信 293
实例 128 设置黑名单 295
实例 129 设置免打扰 297
实例 130 一键截屏 299
实例 131 设置手机屏幕亮度 301
实例 132 通讯录 303
实例 133 网速监控 306
实例 134 悬浮窗 309
实例 135 监控电池状态 311
实例 136 手机应用管理 312

明日学院

实例 137 可以吸附在屏幕两侧的悬浮物.....315

第 15 章 位置服务与地图应用 . . . 319

实例 138 获取百度地图.....319
实例 139 使用百度地图定位自己的位置.....322
实例 140 百度地图公交查询.....324
实例 141 百度全景地图.....327
实例 142 下载百度离线地图.....329
实例 143 百度地图路线规划.....331
实例 144 高德 3D 地图.....333
实例 145 高德地图旋转方向定位.....336
实例 146 高德地图驾车导航.....339
实例 147 高德地图实时显示移动轨迹.....342
实例 148 高德地图实时天气.....344

第 16 章 物联网开发. 347

视频讲解：3 小时 58 分

实例 149 智能摄像头.....347
实例 150 智能家居.....349
实例 151 手表消息通知.....351
实例 152 手机与手表间的消息传递.....353
实例 153 手表动作通知.....357
实例 154 单对单蓝牙聊天.....360
实例 155 检测 Wi-Fi 信号强度.....362
实例 156 Wi-Fi 共享.....365
实例 157 周围 Wi-Fi 检测连接.....368
实例 158 触碰 NFC 标签打开设置的应用....370

第 17 章 游戏开发. 374

视频讲解：2 小时 10 分

实例 159 俄罗斯方块.....374
实例 160 疯狂单击小游戏.....376
实例 161 五子棋游戏.....379
实例 162 仿像素鸟游戏.....381
实例 163 仿别踩白块儿游戏.....384
实例 164 猜牌小游戏.....386
实例 165 拼图小游戏.....389
实例 166 仿关灯游戏.....392
实例 167 连连看游戏.....394
实例 168 2048 游戏.....396
实例 169 抓住大公鸡游戏.....398
实例 170 仿飞机大战.....400

第18章 5.0 + 效果(Material Design) . . . 403

实例 171 仿小米联系人折叠动画效果.....403
实例 172 主题切换.....405
实例 173 切换多个选项卡.....408
实例 174 浮动按钮和消息显示.....410
实例 175 过场动画.....413
实例 176 切换列表效果.....416
实例 177 弹出底部弹窗.....419
实例 178 根据图片调节风格.....421
实例 179 特殊提示信息效果.....423
实例 180 展开与收起菜单栏.....425

第 19 章 商城常用. 427

视频讲解：1 小时 40 分

实例 181 转盘大抽奖.....427
实例 182 刮刮乐大抽奖.....430
实例 183 随机抽奖.....432
实例 184 订餐类购物车.....435
实例 185 仿淘宝购物发表评价页.....438
实例 186 弹出小广告.....440
实例 187 广告闪屏页.....442

明日科技

实例 188 收货地址管理.....444
实例 189 使用瀑布流展示明日图书.....446
实例 190 带时间轴效果的物流信息.....448

第 20 章 验证码合集.....451
视频讲解：3 小时 10 分

实例 191 仿 12306 售票网验证码.....451
实例 192 成语验证码.....455
实例 193 文字过滤验证码.....457
实例 194 汉字转拼音验证码.....460
实例 195 汉字验证码.....463
实例 196 图片选择验证码.....466
实例 197 拖动形式的验证码.....469
实例 198 生成本地验证码.....471

第 21 章 启动页合集.....475
视频讲解：58 分

实例 199 闪屏页.....475
实例 200 新手引导.....477
实例 201 炫酷引导页.....479
实例 202 滑动隐藏引导页.....481
实例 203 滑动旋转引导页.....483

第 22 章 加载等待合集.....485

实例 204 加载等待条.....485
实例 205 小球加载等待.....488
实例 206 图形变换加载等待.....490
实例 207 加载等待弹窗.....491
实例 208 加载、错误和完成三种动画.....493

第 23 章 抢红包服务.....496

实例 209 微信抢红包.....496
实例 210 QQ 抢红包.....500

第 24 章 App 应用开发必备模块.....503
视频讲解：2 小时 43 分

实例 211 登录注册.....503
实例 212 视频登录.....505
实例 213 设置界面.....507
实例 214 上拉加载、下拉刷新.....510
实例 215 动态显示或隐藏标题栏.....512
实例 216 仿今日头条 TAB 效果.....515
实例 217 选择头像的注册.....517
实例 218 明日首页轮播图.....519
实例 219 微信底部选项卡.....521
实例 220 退出高德地图的提示.....524

附录 搭建 Android 开发环境.....526

明日科技

第1章 菜单与对话框

实例 001 微信右上角弹出的菜单

应用场景：顶部菜单

源码位置：光盘\Code\01\001
视频位置：光盘\Video\01\001

视频讲解

实例说明

目前，App 应用中 Menu 菜单的样式数不胜数，但是普通 Menu 的位置固定，无法将悬浮的 Menu 摆放在当前界面的任意位置。本实例将模仿微信实现一个单击按钮弹出菜单的控件，可以显示任意视图，并且可以悬浮在当前界面的任意位置。运行实例，将显示如图 1.1 所示的界面；单击右上角的“+”按钮，将弹出如图 1.2 所示的菜单。

图 1.1　初始界面

图 1.2　弹出菜单

关键技术

关键技术

本实例主要使用 PopupWindow 类及其构造方法、View.OnTouchListener 接口中的 OnTouch 事件，实现微信右上角弹出菜单的运行效果。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“WeiMenu”，公司域名为“mingrisoft. com”。

（2）导入资源文件：复制“光盘\Code\01\001\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）在“WeiMenu\app\src\main\res\layout”目录中创建名称为 menu.xml 的菜单布局文件，用于指定菜单的排列方式。

（4）打开 MainActivity 类，在该类中创建 OnMenu() 方法，该方法为菜单按钮的单击事件所调用的方法，用于实现菜单的显示。代码如下：

```
01  public void OnMenu(View view){
02      //获取自定义的菜单布局文件
03      View popupWindow_view = getLayoutInflater().inflate(R.layout.menu, null,false);
04      //创建PopupWindow实例,设置菜单宽度和高度为包裹其自身内容
05      popupWindow = new PopupWindow(popupWindow_view, ActionBar.LayoutParams.WRAP_CONTENT,
06              ActionBar.LayoutParams.WRAP_CONTENT, true);
07      //设置菜单显示在按钮的下面
08      popupWindow.showAsDropDown(findViewById(R.id.btn_menu),0,0);
09      //单击其他地方消失
10      popupWindow_view.setOnTouchListener(new View.OnTouchListener() {
11          @Override
12          public boolean onTouch(View v, MotionEvent event) {
13              //如果菜单存在并且为显示状态，就关闭菜单并初始化菜单
14              if (popupWindow != null && popupWindow.isShowing()) {
15                  popupWindow.dismiss();
16                  popupWindow = null;
17              }
18              return false;
19          }
20      });
21  }
```

扩展学习

在设置菜单显示的位置时，尝试使用 showAtLocation(View parent,int gravity,int x,int y) 方法，根据指定手机屏幕的 x 坐标与 y 坐标，控制菜单显示在屏幕中心的位置。

实例 002 明日学院抽屉式公告

应用场景：隐藏式消息与菜单

源码位置：光盘\Code\01\002

视频位置：光盘\Video\01\002

实例说明

在许多手机 App 应用中，单击一个按钮即可展开一个消息公告或者一个菜单列表，这样可以在有限的范围内放置更多的组件，需要时单击按钮或通过手指拖曳拉出，不需要时将其收起。本实例将实现一个横向拉出的隐藏式消息公告。运行实例，将显示如图 1.3 所示的界面；单击或向左滑动公告标

题栏，将显示具体的公告信息，如图 1.4 所示。

图 1.3　单击公告标题栏

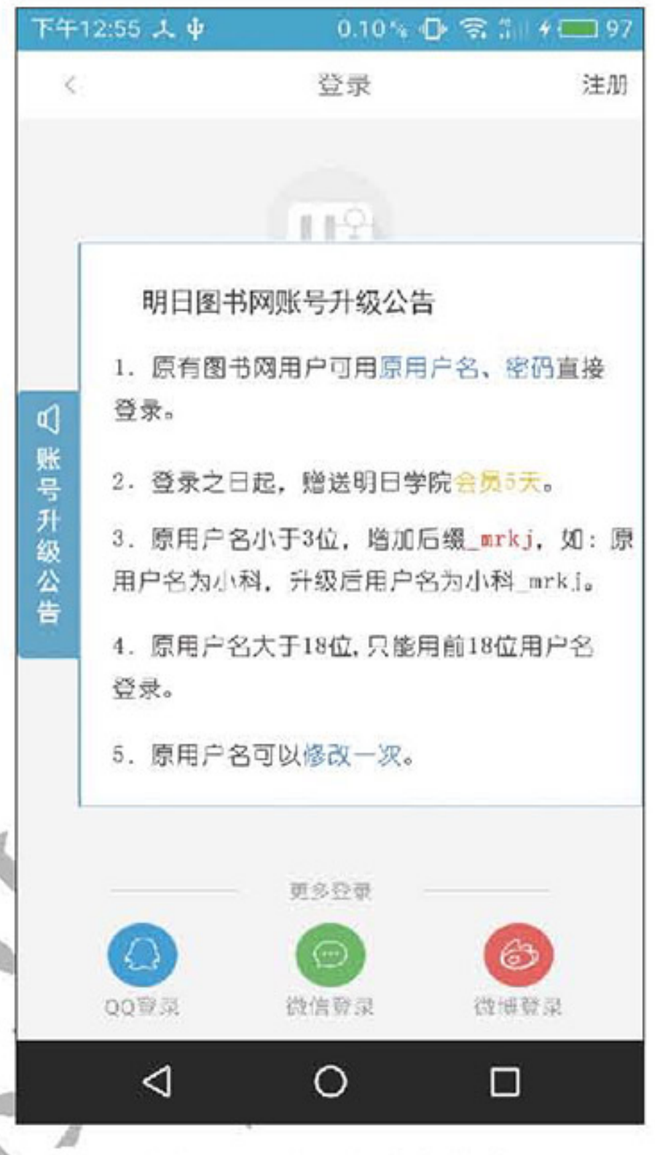


图 1.4　打开公告信息

关键技术

本实例主要使用 SlidingDrawer 控件以及该控件的 orientation 属性、handle 属性和 content 属性，实现抽屉式公告。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“AnnouncementDrawer”，公司域名为“mingrisoft.com”。

（2）导入资源文件：复制“光盘\Code\01\002\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）打开默认生成的 activity_main.xml 布局文件，首先在该文件中设置 SlidingDrawer（抽屉）控件，然后在该控件中添加用于引导的按钮与抽屉所显示的内容。代码如下：

```
<!--设置抽屉控件-->
<SlidingDrawer
    android:id="@+id/drawer1"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:layout_alignParentRight="true"
```

明日科技

```
07      android:layout_centerVertical="true"
08      android:content="@+id/myView"
09      android:handle="@+id/layout1"
10      android:orientation="horizontal">
11      <!--设置引导按钮-->
12      <LinearLayout
13          android:id="@id/layout1"
14          android:layout_width="35dp"
15          android:layout_height="match_parent"
16          android:gravity="center">
17          <ImageView
18              android:id="@+id/myImage1"
19              android:layout_width="wrap_content"
20              android:layout_height="wrap_content"
21              android:src="@drawable/open" />
22      </LinearLayout>
23      <!--设置抽屉内容-->
24      <ImageView
25          android:id="@+id/myView"
26          android:layout_width="wrap_content"
27          android:layout_height="wrap_content"
28          android:src="@drawable/iv_bg" />
29  </SlidingDrawer>
```

（4）打开 MainActivity 类，让该类继承自 Activity 类，并在样式资源文件中设置去除标题栏全屏样式。

扩展学习

本实例是以水平的方式打开消息公告抽屉，请尝试将 activity_main.xml 布局文件中的 android:orientation 属性值修改为“vertical”，实现垂直打开的抽屉，并且通过 GridView 网格视图，以便模拟一个纵向拉伸的抽屉菜单。

实例 003 仿 QQ 侧滑菜单

应用场景：隐藏式侧滑菜单

源码位置：光盘\Code\01\003

视频位置：光盘\Video\01\003

实例说明

视频讲解

侧滑菜单是 App 应用中常见的一种菜单形式，通过手指从左向右滑动时可以将隐藏的菜单显示在屏幕的左侧，当手指从右向左滑动时，将隐藏所使用的菜单。运行实例，效果如图 1.5 所示；手指从左向右滑动，将显示隐藏在左侧的菜单，如图 1.6 所示。

明日科技

图 1.5　手指向右滑动

图 1.6　显示隐藏的菜单

关键技术

关键技术

本实例主要使用自定义 QQMenu 类中的构造方法、View 类中的 onMeasure() 方法、onLayout() 方法，实现仿 QQ 侧滑菜单的效果。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“QQMenu”，公司域名为“mingrisoft. com”。

（2）导入资源文件：复制“光盘\Code\01\003\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）在“app\res\layout”目录中创建名称为“left_menu.xml”的布局文件，用于显示向右滑动时所显示的菜单内容。（代码位置：app\res\layout\left_menu.xml）。

（4）首先在“app\src\main\java”目录中创建 view 包，然后在该包中创建名称为“QQMenu”的 Java 类，该类继承自 HorizontalScrollView，并创建自定义控件的构造方法，最后在构造方法中获取当前手机屏幕的尺寸。代码如下：

```
01      public QQMenu(Context context, AttributeSet attrs) {   //构造方法
02          super(context, attrs);
03          //获取窗口管理器服务
04          WindowManager wm = (WindowManager) context.getSystemService(Context.WINDOW_SERVICE);
05          //创建显示尺寸对象
06          DisplayMetrics outMetrics = new DisplayMetrics();
07          //获取当前屏幕的宽和高
08          wm.getDefaultDisplay().getMetrics(outMetrics);
```

明日科技

```
        //为屏幕宽度赋值
        mScreenWidth = outMetrics.widthPixels;
        //将50dp边距转为像素值(px)
        mMenuRightPadding = (int) TypedValue.applyDimension(
                TypedValue.COMPLEX_UNIT_DIP, 50,
                context.getResources().getDisplayMetrics());
    }
```

重写 onMeasure() 方法，在该方法中设置滚动视图与子视图的宽和高。代码如下：

```
//设置滚动视图与子视图的宽和高
@Override
protected void onMeasure(int widthMeasureSpec, int heightMeasureSpec) {
    if (!call) {                                                    //用于控制只设置一次尺寸
        mScrollView = (LinearLayout) getChildAt(0);                 //获取滚动视图中的子布局
        mMenu = (ViewGroup) mScrollView.getChildAt(0);              //获取菜单区域
        mContent = (ViewGroup) mScrollView.getChildAt(1);           //获取主显示区域
        //设置菜单宽度
        mMenuWidth = mMenu.getLayoutParams().width = mScreenWidth
                - mMenuRightPadding;
        //设置主显示区域宽度
        mContent.getLayoutParams().width = mScreenWidth;
        call = true;
    }
    super.onMeasure(widthMeasureSpec, heightMeasureSpec);
}
```

重写 onLayout() 方法，在该方法中设置偏移量实现菜单隐藏。代码如下：

```
//设置偏移量让菜单隐藏
@Override
protected void onLayout(boolean changed, int l, int t, int r, int b) {
    super.onLayout(changed, l, t, r, b);
    if (changed) {
        this.scrollTo(mMenuWidth, 0);    //滚动条向右移动，主显示区域向左移动
    }
}
```

重写 onTouchEvent() 方法，在该方法中判断手指抬起时隐藏或显示菜单。

（5）在默认生成的 activity_main.xml 布局文件中添加自定义滚动条控件。（代码位置：app\res\layout\activty_main.xml）。

扩展学习

本实例中通过自定义的横向滚动视图，实现了横向滑动隐藏或显示的菜单，请尝试在 QQMenu 类的构造方法中添加“this.setHorizontalScrollBarEnabled(false);”代码，实现隐藏默认显示的横向滚动条。

实例 004　可以收放的旋转菜单

应用场景：收缩菜单

源码位置：光盘\Code\01\004

实例说明

通过一个按钮弹出多个子按钮，既美观又节省屏幕的空间。本实例将实现一个类似的可收放并旋转的菜单。运行实例，效果如图 1.7 所示；单击屏幕中的主按钮，将旋转弹出扇形子菜单，如图 1.8 所示；单击其中某个子按钮，该按钮在放大后消失，其他子按钮缩小后消失，如图 1.9 所示。

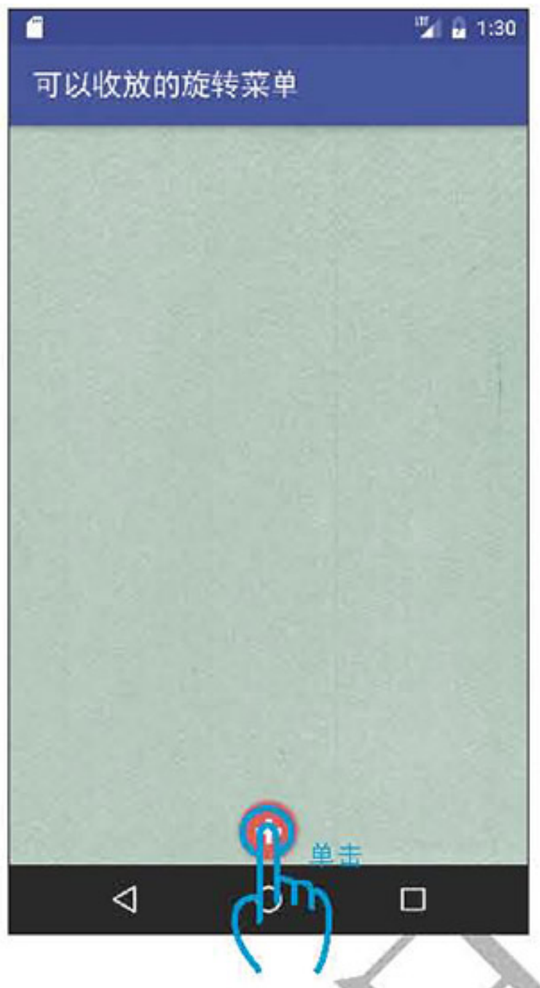


图 1.7　单击主按钮

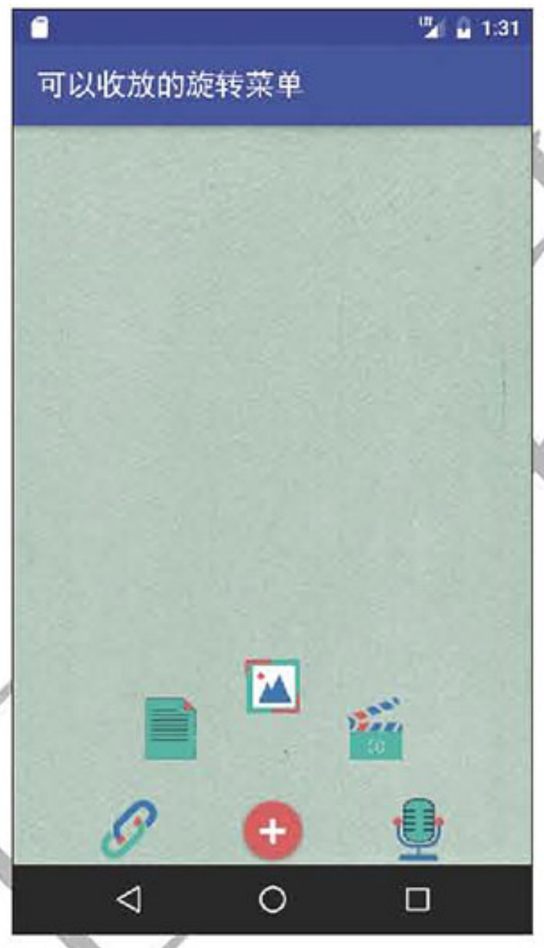


图 1.8　显示扇形子菜单

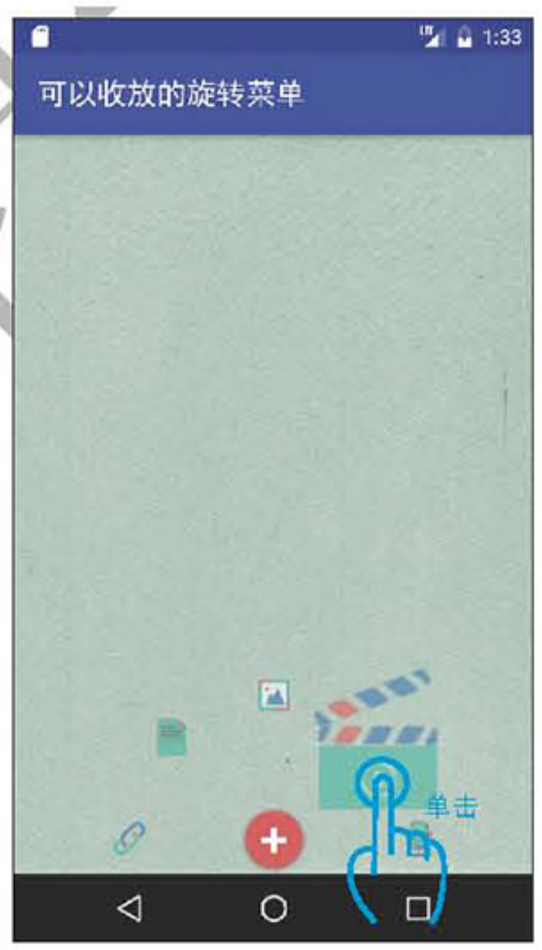


图 1.9　单击子菜单

关键技术

本实例主要使用 Resources.Theme 类中的 obtainStyledAttributes() 方法、TypedArray 类中的 getDimension() 方法和自定义 shrinkMenu() 方法。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“SatelliteMenu”，公司域名为“mingrisoft.com”。

（2）导入资源文件：复制“光盘\Code\01\004\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）在“SatelliteMenu\app\src\main\java”目录中创建 view 包，在该包中创建 SatelliteMenu 类，该类继承自 ViewGroup，首先通过 SatelliteMenu() 构造方法获取屏幕的宽度与高度，用于计算按钮显示的坐标位置，然后获取自定义属性的值，最后获取按钮半径的属性值。代码如下：

```
01  public SatelliteMenu(Context context, AttributeSet attrs, int defStyleAttr) {
02      super(context, attrs, defStyleAttr);
03      //获取屏幕宽和高
04      WindowManager wm = (WindowManager) getContext()
05              .getSystemService(Context.WINDOW_SERVICE);
06      pm_width = wm.getDefaultDisplay().getWidth();
07      pm_height = wm.getDefaultDisplay().getHeight();
08
09      //获取自定义属性的值
10      TypedArray a = context.getTheme().obtainStyledAttributes(attrs,
11              R.styleable.ArcMenu, defStyleAttr, 0);
12      //获取按钮半径属性值
13      mRadius = (int) a.getDimension(R.styleable.ArcMenu_radius, TypedValue
14              .applyDimension(TypedValue.COMPLEX_UNIT_DIP, 100,
15                      getResources().getDisplayMetrics()));
16      a.recycle();                                          //回收资源
17  }
```

定义一个 buttonLayout() 方法来设置主按钮在布局中显示的位置。代码如下：

```
01  private void buttonLayout() {
02      mButton = getChildAt(0);                              //获取菜单主按钮
03      //设置按钮的单击事件
04      mButton.setOnClickListener(new OnClickListener() {
05          @Override
06          public void onClick(View v) {
07              rotatemButton(v, 0f, 360f, 500);              //调用主按钮旋转的方法
08              shrinkMenu(500);                              //调用收缩子按钮动画的方法
09          }
10      });
11      //初始化按钮左边距与上边距
12      int l = 0;
13      int t = 0;
14      //获取主按钮的宽度与高度
15      int width = mButton.getMeasuredWidth();
16      int height = mButton.getMeasuredHeight();
17      //判断主按钮位置
18      l = pm_width/2-width/2;
19      t = getMeasuredHeight() - height;
20      mButton.layout(l, t, l + width, t + width);           //设置主按钮在父容器的位置
21  }
```

（4）通过 childLayout() 方法来实现子按钮在布局中显示的位置。

（5）通过 shrinkMenu() 方法来实现子按钮收缩的动画过程。

（6）通过 childButtonClickAnim() 方法来实现子按钮的单击动画效果。代码如下：

```
01  private void childButtonClickAnim(int pos) {
02      for (int i = 0; i < getChildCount() - 1; i++)
03      {
04          View childView = getChildAt(i + 1);
05          if (i == pos)                          //当前单击的子按钮
06          {   //启动子按钮变大并消失动画
07              childView.startAnimation(scaleBigAnim(200));
08          } else
09          {
10              //启动其他没有被单击的按钮变小并消失动画
11              childView.startAnimation(scaleSmallAnim(200));
12          }
13          childView.setClickable(false);     //子按钮不可单击
14          childView.setFocusable(false);     //子按钮不可调焦
15      }
16  }
```

扩展学习

本实例已经实现了菜单旋转收放的功能，请尝试在 SatelliteMenu 类中添加 setOnSatelliteMenuItem ClickListener() 方法，并在 MainActivity 类中调用该方法实现单击子按钮后显示子菜单按钮标记信息的功能。

实例 005　仿 58 同城的退出动画

应用场景：对话框

源码位置：光盘\Code\01\005
视频位置：光盘\Video\01\005

实例说明

对话框对于 App 应用是必不可少的一个控件，本实例将实现一个带动画效果的对话框。运行实例，效果如图 1.10 所示；单击手机或模拟器的返回按钮，提示退出的对话框将从顶部移动至中间，如图 1.11 所示；单击对话框中的“取消”按钮后，对话框将从中间移动至底部并消失，如图 1.12 所示。

关键技术

本实例主要使用 KeyEvent.Callback 静态接口中的 onKeyDown() 方法、AlertDialog.Builder 类中的 AlertDialog.Builder() 构造方法和 Window 类，实现带动画效果的对话框。

图 1.10 单击返回按钮

图 1.11 顶部进入对话框

图 1.12 底部退出对话框

实现过程

（1）打开 Android Studio，新建一个项目，命名为“AnimDialog”，公司域名为“mingrisoft. com”。

（2）导入资源文件：复制“光盘\Code\01\005\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）在“app\res\layout”目录中创建一个名称为“dialog_layout.xml”的布局文件，该布局文件主要用于布局对话框中的动画显示内容。（代码位置：app\res\layout\dialog_layout.xml）。

（4）在“app\res\layout”目录中创建 anim 文件夹，在该文件夹中，创建名称为“dialog_enter.xml”的动画资源文件，用于实现对话框进入时的动画。代码如下：

```
<!--从上部进入到中间-->
<translate
    android:duration="200"
    android:fromYDelta="-100%">
</translate>
```

创建名称为“dialog_exit.xml”的动画资源文件，作为对话框退出时的动画文件。代码如下：

```
<!--从中间到底部退出-->
<translate
    android:fromYDelta="0"
```

```
04      android:toYDelta="100%"
05      android:duration="200"/>
```

（5）打开“app\res\values”目录中的“styles.xml”样式资源文件，在该文件中设置对话框的动画样式。代码如下：

```
01  <style name="mystyle" parent="android:Animation">
02      <!--进入时的动画-->
03      <item name="@android:windowEnterAnimation">@anim/dialog_enter</item>
04      <!--退出时的动画-->
05      <item name="@android:windowExitAnimation">@anim/dialog_exit</item>
06  </style>
```

（6）打开 MainActivity 类，在该类中重写 onKeyDown() 方法，该方法中判断是否单击“返回”按钮，如果单击则显示带动画效果的对话框。代码如下：

```
01  @Override
02  public boolean onKeyDown(int keyCode, KeyEvent event) {
03      //判断是否单击返回按钮
04      if (keyCode == KeyEvent.KEYCODE_BACK) {
05          //创建对话框实例
06          dlg = new AlertDialog.Builder(this).create();
07          dlg.show();                                          //显示对话框
08          Window window = dlg.getWindow();                     //获取对话框窗口
09          window.setGravity(Gravity.CENTER);                   //此处设置对话框显示在屏幕中心位置
10          window.setWindowAnimations(R.style.mystyle);         //添加动画
11          window.setContentView(R.layout.dialog_layout);       //设置对话框的布局文件
12          //获取对话框确定按钮
13          button_determine = (Button) window.findViewById(R.id.btn_determine);
14          //获取对话框取消按钮
15          button_cancel = (Button) window.findViewById(R.id.btn_cancel);
16          initEvent();                                         //调用初始化事件方法
17      }
18      return super.onKeyDown(keyCode, event);
19  }
```

扩展学习

本实例实现了单击返回按钮后，对话框从顶部进入、底部退出并消失的效果。请尝试修改 anim 文件夹中的动画文件，实现对话框从屏幕左侧进入、右侧退出并消失的动画效果。

实例 006 明日学院消息通知

应用场景：消息通知

源码位置：光盘\Code\01\006

视频位置：光盘\Video\01\006

实例说明

在使用手机时，当有未接来电或是新短消息时，手机会给出相应的提示信息，这些提示信息通常会显示到手机屏幕的状态栏上。本实例将实现一个模拟明日学院的消息通知。运行实例，效果如图 1.13 所示；打开 App 应用程序，通知将自动显示在状态栏中，按住状态栏并向下滑动，直到出现如图 1.14 所示的通知窗口，单击第一项，如图 1.15 所示。

图 1.13 明日学院的通知

图 1.14 显示通知列表

图 1.15 通知的详细内容

关键技术

本实例主要使用 Activity 类中的 getSystemService() 方法、NotificationCompat.Builder 类，实现明日学院消息通知的运行效果。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“MessageNotification”，公司域名为“mingrisoft.com”。

（2）导入资源文件：复制“光盘\Code\01\006\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）在 java 目录下的 com.mingrisoft.messagenotification 包中创建名称为 MessageActivity 的 Activity 界面，用于显示通知的详细内容。

（4）打开“app\res\layout”目录中的“activity_message.xml”布局文件，在该文件中删除内边距代码并设置背景图片。

（5）打开 MainActivity 类，在该类中获取通知管理器并创建通知对象。代码如下：

```
//获取通知管理器，用于发送通知
NotificationManager notificationManager =
        (NotificationManager) getSystemService(NOTIFICATION_SERVICE);
NotificationCompat.Builder notification = new
        NotificationCompat.Builder(MainActivity.this);    //创建一个Notification对象
```

设置通知的相关参数与通知的发送时间。代码如下：

```
notification.setAutoCancel(true);
//设置显示在状态栏的通知提示信息
notification.setTicker("Android课程第一季上线啦！");
//设置通知的小图标
notification.setSmallIcon(R.mipmap.ic_launcher);
//设置下拉列表中的大图标
notification.setLargeIcon(BitmapFactory.decodeResource(getResources(),R.mipmap.ic_launcher));
//设置通知内容的标题
notification.setContentTitle("Android入门第一季！");
//设置通知内容
notification.setContentText("点击查看详情！");
//设置发送时间
notification.setWhen(System.currentTimeMillis());
```

设置从通知栏跳转至通知的详细内容界面，并进行通知的发送。代码如下：

```
//创建一个启动其他Activity的Intent
Intent intent = new Intent(MainActivity.this,
        MessageActivity.class);
PendingIntent pi = PendingIntent.getActivity(
        MainActivity.this, 0, intent, 0);
//设置通知栏单击跳转
notification.setContentIntent(pi);
//发送通知
notificationManager.notify(NOTIFYID, notification.build());
```

扩展学习

本实例实现了一个明日学院的消息通知。请尝试在 MainActivity 类的 onCreate() 方法中实现接收通知时的震动与铃声。

明日科技

实例 007 退出当前账号对话框

应用场景：自定义对话框样式

源码位置：光盘\Code\01\007
视频位置：光盘\Video\01\007

实例说明

视频讲解

对话框在 App 应用中是一种常见的控件，但是系统默认的对话框控件外观与样式比较固定，本实例将通过自定义的方式实现退出 QQ 当前账号对话框的显示。运行实例，效果如图 1.16 所示，单击“退出当前账号”按钮，将弹出模拟 QQ 退出当前账号的确认对话框，如图 1.17 所示。

图 1.16 单击“退出当前账号”按钮

图 1.17 显示确认退出对话框

关键技术

关键技术

本实例主要使用 LayoutInflater 类中的 inflate() 方法和 Dialog 类中的 setContentView() 方法，实现模拟 QQ 退出当前账号对话框的显示。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“IosDialog”，公司域名为“mingrisoft. com”。

（2）导入资源文件：复制“光盘\Code\01\007\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）在“app\res\layout”目录中创建名称为“mdialoglayout.xml”的布局文件，用于布局对话框所显示的内容。（代码位置：app\res\layout\mdialoglayout.xml）

（4）首先在“app\src\main\java”目录中创建 view 包，然后在该包中创建名称为“Mdialog”的 Java 类，该类继承自 Dialog，并创建自定义控件的构造方法，在构造方法中获取相关组件，并设置对话框使用的布局文件。代码如下：

```
01  //自定义构造方法
02  public Mdialog(Context context) {
03      super(context, R.style.mdialog);
04      View view = LayoutInflater.from(getContext()).
05              inflate(R.layout.mdialoglayout, null);                    //通过LayoutInflater获取布局
06      tv = (TextView) view.findViewById(R.id.title);                    //获取显示标题的文本框控件
07      button_cancel = (Button) view.findViewById(R.id.btn_cancel);      //获取取消按钮
08      button_exit = (Button) view.findViewById(R.id.btn_exit);          //获取确认退出按钮
09      setContentView(view);                                             //设置显示的视图
10  }
```

创建用于显示标题文字的 setTv() 方法，以及对话框中“取消”按钮与“确定退出”按钮的监听方法。代码如下：

```
01  //设置显示的标题文字
02  public void setTv(String content) {
03      tv.setText(content);
04  }
05  //“取消”按钮的监听方法
06  public void setOnCancelListener(View.OnClickListener listener){
07      button_cancel.setOnClickListener(listener);
08  }
09
10  //“退出”按钮的监听方法
11  public void setOnExitListener(View.OnClickListener listener){
12      button_exit.setOnClickListener(listener);
13  }
```

（5）在 MainActivity 类中创建 OnExitNumber() 方法，该方法用于实现对话框的显示与关闭功能。代码如下：

```
01  public void OnExitNumber(View view) {
02      final Mdialog mdialog = new Mdialog(this);                        //实例化自定义对话框
03      //对话框中“退出”按钮的监听事件
04      mdialog.setOnExitListener(new View.OnClickListener() {
05          @Override
06          public void onClick(View v) {
07              //如果对话框处于显示状态
08              if (mdialog.isShowing()) {
09                  mdialog.dismiss();                                    //关闭对话框
10                  finish();                                             //关闭当前界面
11              }
12          }
13      });
14      //对话框中取消按钮的监听事件
15      mdialog.setOnCancelListener(new View.OnClickListener() {
16          @Override
17          public void onClick(View v) {
```

明日科技

```
18              if (mdialog != null && mdialog.isShowing()) {
19                  mdialog.dismiss();                              //关闭对话框
20              }
21          }
22      });
23      mdialog.show();                                             //显示对话框
24  }
```

扩展学习

本实例将对话框作为“退出当前账号”按钮的事件来执行。请尝试应用该技术实现模拟 App 应用的提示升级时所显示的对话框。

实例 008　圆盘式工具箱菜单

应用场景：旋转菜单

源码位置：光盘\Code\01\008

实例说明

圆盘式工具箱菜单在 App 应用中是比较常见的菜单样式。本实例将实现一个圆盘式工具箱菜单。运行实例，效果如图 1.18 所示；工具图标围绕手机中心摆放，根据手指滑动的方向菜单将进行旋转，如图 1.19 所示。

图 1.18　旋转前菜单

图 1.19　旋转后菜单

关键技术

关键技术

本实例主要使用自定义 CircleMenuLayout 类中的 getAngle() 方法、getQuadrant() 方法和 setMenuItemIconsAndTexts() 方法，实现圆盘式工具箱菜单的功能。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“CircleMenu”，公司域名为“mingrisoft.com”。

（2）导入资源文件：复制“光盘\Code\01\008\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）在“CircleMenu\app\src\main\java”目录中创建 view 包，在该包中创建 CircleMenuLayout 类，该类继承自 ViewGroup。在 dispatchTouchEvent() 方法中，首先判断手指按下时的坐标与角度，代码如下：

```
01      case MotionEvent.ACTION_DOWN:                          //手指按下事件
02          mLastX = x;                                        //记录x坐标
03          mLastY = y;                                        //记录y坐标
04          mDownTime = System.currentTimeMillis();            //记录按下的时间
05          mTmpAngle = 0;                                     //记录按下到抬起时旋转的角度
06          if (isFling) {                                     //如果当前已经在快速滚动
07              removeCallbacks(mFlingRunnable);               //移除快速滚动的回调
08              isFling = false;                               //标记为没有快速滚动
09              return true;
10          }
11          break;
```

然后判断手指移动时的坐标与角度，代码如下：

```
01  case MotionEvent.ACTION_MOVE:                              //手指移动事件
02      float start = getAngle(mLastX, mLastY);                //获得开始的角度
03      float end = getAngle(x, y);                            //获得当前的角度
04      //如果在第一象限或第四象限，则直接用当前的角度减去开始的角度，角度值是正值
05      if (getQuadrant(x, y) == 1 || getQuadrant(x, y) == 4) {
06          mStartAngle += end - start;                        //布局时的开始角度
07          mTmpAngle += end - start;                          //按下到抬起时旋转的角度
08      } else
09      //如果在第二象限或第三象限，则用开始的角度减去当前的角度，角度值是负值
10      {
11          mStartAngle += start - end;
12          mTmpAngle += start - end;
13      }
14      //重新布局
15      requestLayout();
16      mLastX = x;                                            //记录x坐标
17      mLastY = y;                                            //记录y坐标
18      break;
```

最后判断手指抬起时，菜单是否自动旋转，代码如下：

```
01  case MotionEvent.ACTION_UP:                                //手指抬起事件
02      //计算每秒移动的角度
03      float anglePerSecond = mTmpAngle * 1000
04              / (System.currentTimeMillis() - mDownTime);
```

明日科技

```
05      //如果达到该值则被快速移动
06      if (Math.abs(anglePerSecond) > mFlingableValue && !isFling)   {
07          //post一个任务，去自动旋转
08          post(mFlingRunnable = new AutoFlingRunnable(anglePerSecond));
09          return true;
10      }
11      //如果当前旋转角度超过NOCLICK_VALUE，则屏蔽单击事件
12      if (Math.abs(mTmpAngle) > NOCLICK_VALUE) {
13          return true;
14      }
15      break;
```

扩展学习

本实例已经实现了圆盘式工具箱菜单，尝试通过在 MainActivity 类中调用 setOnMenuItemClickListener() 方法为圆盘式工具箱菜单添加单击事件，实现单击子按钮显示对应按钮名称的功能。

实例 009　彩虹式菜单

应用场景：功能菜单

源码位置：光盘\Code\01\009

视频位置：光盘\Video\01\009

视频讲解

实例说明

彩虹菜单是通过动画效果对布局进行旋转进入的彩虹式菜单。本实例将通过 Animation 类与 RotateAnimation 类来实现彩虹菜单。运行实例，效果如图 1.20 所示；打开应用后显示默认的三道彩虹菜单效果；单击第一道彩虹将隐藏上面的两道彩虹菜单，如图 1.21 所示；隐藏后再次单击，将弹出第二道彩虹菜单，如图 1.22 所示，以此类推。

图 1.20　显示三道彩虹菜单

图 1.21　隐藏两道彩虹菜单

图 1.22　显示第二道彩虹菜单

关键技术

本实例主要使用 Animation 类中的 setFillAfter() 方法、setDuration() 方法、setStart Offset() 方法，RotateAnimation 类中的 RotateAnimation() 构造方法，实现彩虹菜单的功能。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“RainbowMenu”，公司域名为“mingrisoft. com”。

（2）导入资源文件：复制“光盘\Code\01\009\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）在“RainbowMenu\app\src\main\java\com\mingrisoft\rainbowmenu”目录中创建 MyAnimation 类，通过创建 animationIN() 方法，实现彩虹菜单进入的动画效果。代码如下：

```
01  public static void animationIN(ViewGroup viewGroup, int duration){
02      for(int i = 0; i < viewGroup.getChildCount(); i++ ){          //遍历布局中的按钮控件
03          viewGroup.getChildAt(i).setVisibility(View.VISIBLE);      //设置显示
04          viewGroup.getChildAt(i).setFocusable(true);               //获得焦点
05          viewGroup.getChildAt(i).setClickable(true);               //可以单击
06      }
07      Animation animation;                                          //动画对象
08      /**
09       * 旋转动画
10       */
11      animation = new RotateAnimation(-180, 0, Animation.RELATIVE_TO_SELF,
12              0.5f, Animation.RELATIVE_TO_SELF, 1.0f);              //创建旋转动画
13      animation.setFillAfter(true);                                 //停留在动画结束位置
14      animation.setDuration(duration);                              //动画过程的时间
15      viewGroup.startAnimation(animation);                          //启动动画
16  }
```

通过创建 animationOUT() 方法，实现彩虹菜单退出的动画效果。代码如下：

```
01  public static void animationOUT(final ViewGroup viewGroup, int duration , int startOffSet){
02      Animation animation;                                          //动画对象
03      animation = new RotateAnimation(0, -180, Animation.RELATIVE_TO_SELF, 0.5f,
04              Animation.RELATIVE_TO_SELF, 1.0f);                    //创建旋转动画
05      animation.setFillAfter(true);                                 //停留在动画结束位置
06      animation.setDuration(duration);                              //动画过程的时间
07      animation.setStartOffset(startOffSet);                        //设置多长时间后启动
08      //设置动画监听事件
09      animation.setAnimationListener(new Animation.AnimationListener() {
10          @Override
11          public void onAnimationStart(Animation animation) {
```

```
12              //TODO Auto-generated method stub
13          }
14              @Override
15          public void onAnimationRepeat(Animation animation) {
16              //TODO Auto-generated method stub
17          }
18          /**
19           * 动画结束后的事件处理
20           */
21          @Override
22              public void onAnimationEnd(Animation animation) {
23                  for(int i = 0; i < viewGroup.getChildCount(); i++ ){
24                      viewGroup.getChildAt(i).setVisibility(View.GONE);    //设置隐藏菜单中的按钮
25                      viewGroup.getChildAt(i).setFocusable(false);         //失去焦点
26                      viewGroup.getChildAt(i).setClickable(false);         //不可单击
27                  }
28              }
29      });
30      viewGroup.startAnimation(animation);                                 //启动退出动画
31  }
```

（4）打开 MainActivity 类，在该类中设置触发彩虹菜单显示与隐藏的按钮事件。（代码位置：RainbowMenu\app\src\main\java\com\mingrisoft\rainbowmenu\MainActivity.java）

扩展学习

本实例已经通过 Animation 类与 RotateAnimation 类实现了彩虹菜单的功能，尝试实现彩虹菜单中子按钮的单击事件。首先让 MainActivity 类实现 View.OnClickListener 接口，然后在 onCreate() 方法中获取所有的按钮控件并注册单击事件。最后在 onClick() 方法中判断单击的是哪个按钮，并执行对应的操作。

实例 010 滑动删除聊天记录

应用场景：删除菜单

源码位置：光盘\Code\01\010
视频位置：光盘\Video\01\010

实例说明

在使用手机 QQ 时，可以在聊天记录列表的子项中滑动，“删除”按钮将显示在该子项的右侧。单击“删除”按钮将在菜单列表中删除该子项。本实例将实现滑动删除菜单的功能。运行实例，将显示如图 1.23 所示的默认菜单列表，向左滑动需要删除的菜单子项时，将显示“删除”按钮，如图 1.24 所示；单击“删除”按钮，即可删除指定的菜单项，如图 1.25 所示。

图 1.23　显示默认的菜单列表

图 1.24　滑动后显示“删除”按钮

图 1.25　删除列表子项

关键技术

本实例主要使用自定义 SlidingButtonView 类中的 changeScrollx() 方法、RecyclerView.ViewHolder 类中的 getLayoutPosition() 方法和 RecyclerView.Adapter 类中的 notifyItemRemoved() 方法，实现滑动删除菜单的功能。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“SlideDeleteMenu”，公司域名为“mingrisoft.com”。

（2）导入资源文件：复制“光盘\Code\01\010\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）打开 Gradle Scripts 目录中的 build.gradle（Module：app）文件，在 dependencies 节点中添加 recyclerview 控件的依赖，代码如下：

```
compile 'com.android.support:recyclerview-v7:24.2.1'
```

（4）在“SlideDeleteMenu\app\src\main\java”目录中创建 utils 包，在该包中创建 Utils 类。该类中的 getScreenWidth() 方法用于获取手机屏幕的宽度。

（5）在“SlideDeleteMenu\app\src\main\java”目录中创建 view 包，在该包中创建名称为 SlidingButtonView.java 的类，该类用于实现自定义滑动“删除”按钮的控件。主要通过自定义 changeScrollx() 方法实现根据滑动的距离判断是否显示“删除”按钮的功能。代码如下：

```
01  public void changeScrollx(){
02      if(getScrollX() >= (lScrollWidth/2)){          //触摸滑动的距离大于删除按钮宽度的一半
03          this.smoothScrollTo(lScrollWidth, 0);      //显示删除按钮
04      }else{
```

```
            this.smoothScrollTo(0, 0);                          //隐藏删除按钮
        }
    }
```

（6）在“SlideDeleteMenu\app\src\main\java”目录中创建 adapter 包，在该包中创建 Adapter 类，该类继承自“RecyclerView.Adapter<Adapter.MyViewHolder>”，用于实现显示列表菜单的适配器。通过 onBindViewHolder() 方法，设置列表菜单中子项所显示的内容。代码如下：

```
public void onBindViewHolder(final MyViewHolder holder, int position) {
    //设置图标
    holder.img.setBackgroundResource(listIcon.get(position));
    //设置名称
    holder.name.setText(listName.get(position));
    //设置信息
    holder.info.setText(listInfo.get(position));
    //设置内容布局的宽为屏幕宽度
    holder.layout_content.getLayoutParams().width = Utils.getScreenWidth(lContext);
    //删除按钮的事件，单击后删除整行
    holder.btn_Delete.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            int n = holder.getLayoutPosition();          //获取要删除行的位置
            removeData(n);                               //删除列表中的子项
        }
    });
}
```

编写 removeData() 方法，实现删除列表菜单中的子项。代码如下：

```
public void removeData(int position) {
    listIcon.remove(position);                           //删除子项中显示图标
    listName.remove(position);                           //删除名字
    listInfo.remove(position);                           //删除信息
    notifyItemRemoved(position);                         //删除列表
}
```

（7）打开 MainActivity 类，在该类中设置适配器与列表菜单的显示。（代码位置：SlideDeleteMenu\app\src\main\java\com\mingrisoft\slidedeletemenu\MainActivity.java）。

扩展学习

本实例已经通过 RecyclerView.Adapter 类与 HorizontalScrollView 类来实现滑动删除菜单的功能，尝试实现单击“删除”按钮时弹出确认删除的对话框。打开 Adapter 类在删除按钮的单击事件 onClick() 方法中，通过 AlertDialog.Builder 创建确认删除对话框，并实现单击对话框中的“确定”按钮删除列表中子项的功能。

明日科技

实例 011　带图标的消息提示弹窗

应用场景：信息提示

源码位置：光盘\Code\01\011

实例说明

消息弹窗的运行效果会影响 App 应用的用户体验。本实例将实现一个带图标的消息提示弹窗。运行实例，在出现的界面中单击“提示”按钮后，弹出一个带有图标的 Toast 提示，而且背景为炫目的粉色，如图 1.26 所示。

图 1.26　自定义 Toast 的效果

关键技术

本实例主要使用 Toast 类的 setView() 方法来设置 Toast 的显示效果。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“CustomToast”，公司域名为“mingrisoft. com”。

（2）导入资源文件：复制“光盘\Code\01\011\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）创建 MyToast 类，在该类中设置创建 MyToast 对象的方法。代码如下：

```
01  /**
02   * 创建MyToast对象
03   */
04  public MyToast build(){
05      if (null == mine){
06          mine = new MyToast(mContext);                    //创建对象
```

```
    }
    if (isShowIcon){
    //隐藏图标
        mine.icon.setVisibility(View.VISIBLE);
        if (null != icon){                                    //判断是否显示图标
            mine.icon.setImageBitmap(icon);                   //设置图标
        }else {
    //设置背景图片
            mine.icon.setBackgroundResource(iconID);
        }
    }
    if (!message.isEmpty()){                                  //判断内容是否为空
        mine.message.setText(message);
    }else {
        mine.message.setText("");
    }
    //设置背景
    mine.view.setBackground(new BackgroundDrawable(backgroundColor,mContext));
    mine.toast.setDuration(duration);                         //设置时长
    mine.toast.setView(mine.view);                            //添加自定义效果
    mine.toast.setGravity(gravity,offsetX,offsetY);           //设置偏移量
    return mine;
}
```

在 MainActivity 类中调用 MyToast 类，并设置 MyToast 的显示内容和效果。关键代码如下：

```
//创建Toast
myToast = new MyToast.Builder(this)
        .setMessage("自定义Toast效果！")                      //设置提示文字
        .setBackgroundColor(0xe9ff4587)                       //设置背景颜色
        .setGravity(Gravity.CENTER)                           //设置Toast位置
        .showIcon(true)                                       //是否显示图标
        .build();                                             //创建Toast
```

设置按钮的单击事件并设置按钮防抖动。代码如下：

```
//按钮的单击事件
RxView.clicks(hintToast)
        .throttleFirst(2, TimeUnit.SECONDS)
        .subscribe(aVoid -> myToast.show());
```

扩展学习

请尝试为自定义的 Toast 增加更改字体颜色的自定义方法。

实例 012　使用 PopWindow 实现选择菜单

应用场景：界面效果

源码位置：光盘\Code\01\012

实例说明

本实例使用 PopWindow 来实现选择菜单的效果，运行实例，将显示如图 1.27 所示的界面，单击任意一个菜单项，会弹出相应的动画效果，如图 1.28 所示。

图 1.27　界面的默认效果

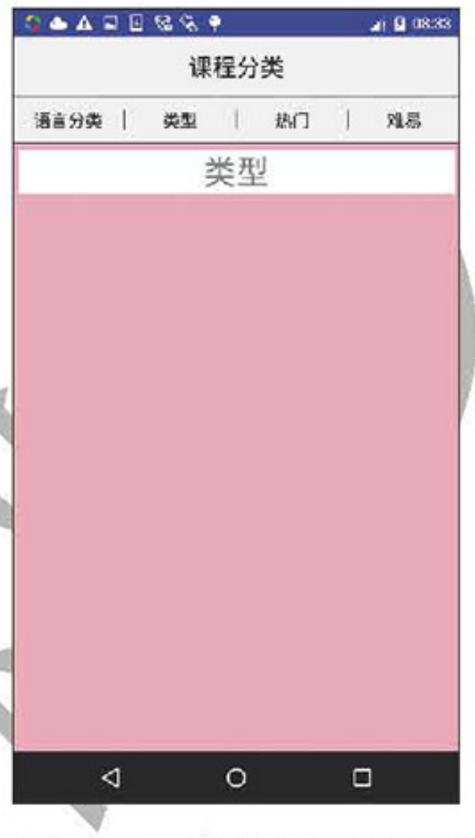


图 1.28　菜单打开的效果

关键技术

关键技术

本实例主要使用 PopWindow 来实现选择弹窗的效果，使用 setAnimationStyle() 方法设置弹窗的动画效果，使用 setBackgroundDrawable() 方法设置弹窗的背景，使用 setOutsideTouchable() 方法设置单击弹窗外部是否关闭弹窗。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“DownMenu”，公司域名为“mingrisoft.com”。

（2）导入资源文件：复制“光盘\Code\01\012\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）在 MainActivity 类的 onCreate() 方法中，初始化菜单项，并设置菜单的单击事件。代码如下：

```
01  //初始化菜单项
02  findViewById(R.id.select_1).setOnClickListener(this);
03  findViewById(R.id.select_2).setOnClickListener(this);
04  findViewById(R.id.select_3).setOnClickListener(this);
05  findViewById(R.id.select_4).setOnClickListener(this);
06  tag_line = (ImageView) findViewById(R.id.tag_line);
```

创建 setMenu() 方法，用来设置 PopWindow 的显示和隐藏。代码如下：

```
/**
 * 设置菜单
 */
private void setMenu(String message) {
    if (null == popupWindow){
        View view = getLayoutInflater().inflate(R.layout.pop_layout, null);
        txt = (TextView) view.findViewById(R.id.text);
        //创建弹窗
        popupWindow = new PopupWindow(view, WindowManager.LayoutParams.MATCH_PARENT,
                WindowManager.LayoutParams.MATCH_PARENT);
        popupWindow.setAnimationStyle(R.style.PopupWindowAnimation); //设置动画效果
        popupWindow.setOutsideTouchable(false);                      //设置单击弹窗外部不关闭弹窗
        popupWindow.setFocusable(false);                             //设置不获取焦点
    }
    if (isShow){
        if (index == oldIndex){
            popupWindow.dismiss();                                   //关闭弹窗
            isShow = false;
            return;
        }
    }
    txt.setText(message);
    popupWindow.showAsDropDown(tag_line);                            //设置显示弹窗
    isShow = true;
    oldIndex = index;
}
```

扩展学习

当前的 PopWindow 的背景效果是完全透明的，请给 PopWindow 设置一个背景颜色。

明日科技

第2章 Android 常用控件应用

实例 013 三种进度条样式

应用场景：加载进度

源码位置：光盘\Code\02\013

实例说明

在 App 应用加载网络数据或资源的时候，经常可以看到各种各样的加载进度条，本实例为用户提供多种加载进度条的样式。运行实例，将显示如图 2.1 所示的界面，单击“动画进度条”按钮，将加载动画进度条，如图 2.2 所示；单击“颜色进度条”按钮，将加载颜色进度条，如图 2.3 所示；单击“图片进度条”按钮，将加载图片进度条，如图 2.4 所示。

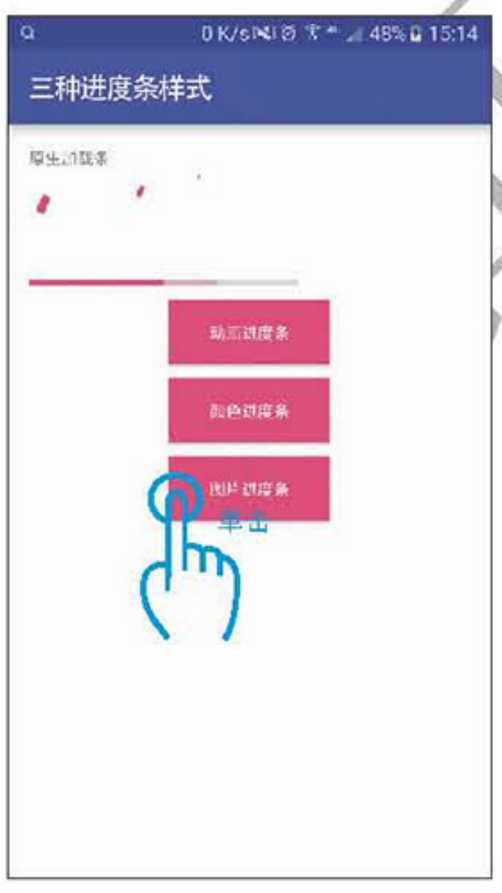


图 2.1 运行效果

图 2.2 动画进度条效果

图 2.3 颜色进度条效果

图 2.4 图片进度条效果

关键技术

本实例主要使用 AlertDialog 类中的 Builder() 方法、Dialog 类中的 setContentView() 方法、View 类中的 setOnClickListener() 方法和自定义方法 showRoundProcessDialog()，实现加载多种进度条。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“ProgressBar”，公司域名为“mingrisoft.com”。

（2）导入资源文件：复制“光盘\Code\02\013\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）在“app\res\layout”目录下默认生成的 activity_main.xml 布局文件中添加原生进度条。代码如下：

```
<TextView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="原生进度条"/>

<LinearLayout
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:orientation="horizontal">
    <!--原生圆形进度条-->
    <ProgressBar
        style="?android:attr/progressBarStyleLarge"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"/>

    <ProgressBar
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"></ProgressBar>

    <ProgressBar
        style="?android:attr/progressBarStyleSmall"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"/>

</LinearLayout>
<!--原生线型进度条-->
<ProgressBar
    android:id="@+id/progressBar5"
    style="?android:attr/progressBarStyleHorizontal"
    android:layout_width="200dp"
    android:layout_height="wrap_content"
    android:max="100"
    android:progress="50"
    android:secondaryProgress="70"></ProgressBar>
```

明日科技

（4）在“app\java\com\mingrisoft\progressbar”文件夹下默认生成的 MainActivity 类中，创建 inti() 方法初始化按钮。代码如下：

```
01  private void init() {
02      //初始化控件
03      anim = (Button) findViewById(R.id.anim);
04      color = (Button) findViewById(R.id.color);
05      icon = (Button) findViewById(R.id.icon);
06  }
```

在 MainActivity 类中，创建 setClickListener() 方法，为按钮绑定单击事件。代码如下：

```
01  private void setClickListener() {
02      //绑定单击事件
03      anim.setOnClickListener(this);
04      color.setOnClickListener(this);
05      icon.setOnClickListener(this);
06  }
```

在 MainActivity 类中创建 showRoundProcessDialog() 方法，用于根据布局显示不同样式的进度条。代码如下：

```
01  public void showRoundProcessDialog(Context mContext, int layout) {
02      //初始化进度条
03      mDialog = new AlertDialog.Builder(mContext).create();
04      //显示进度条
05      mDialog.show();
06      //注意此处要放在show之后，否则会报异常
07      mDialog.setContentView(layout);
08  }
```

扩展学习

请尝试为进度条添加显示时间，让进度条定时消失。

实例 014　仿支付密码输入框

应用场景：App 支付页

源码位置：光盘\Code\02\014

实例说明

在支付宝、微信等 App 应用中的支付页面里，有一些样式不同而功能相同的支付密码页面。本实例模仿该功能通过自定义输入框样式，实现支付密码输入页面。运行实例，将显示如图 2.5 所示的界面，单击数字键盘，输入密码，如图 2.6 所示；单击数字键盘输入 6 位密码后，弹出输入的密码，如图 2.7 所示。

明日科技

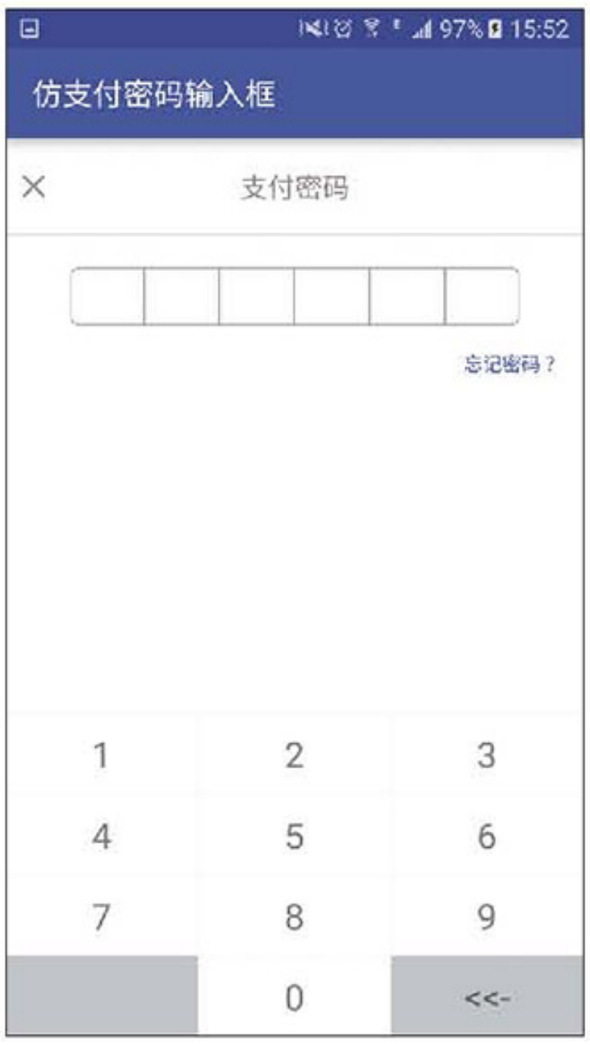


图 2.5　初始化效果

图 2.6　输入密码

图 2.7　弹出输入的密码

关键技术

本实例主要使用了 Adapter 类中的 onItemClick() 方法，TextView 类中的 addTextChangedListener() 方法，View 类中的 onClick() 方法、setBackgroundResource() 方法，TextView 类中的 setEnabled() 方法，实现支付密码输入页面。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“ZFPlay”，公司域名为“mingrisoft.com”。

（2）导入资源文件：复制“光盘\Code\02\014\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）在“app\java\com\mingrisoft\zfplay”文件夹下创建自定义 PasswordView 控件类，用于实现输入支付密码页。

（4）在“app\res\layout”目录下默认生成的 activity_main.xml 布局文件中添加自定义 PasswordView 控件。代码如下：

```
<!--密码输入页-->
<com.mingrisoft.zfplay.PasswordView
    android:id="@+id/pwd_view"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_alignParentBottom="true" />
```

（5）在“app\java\com\mingrisoft\zfplay”文件夹下创建 OnPasswordInputFinish 类，用于自定义接口。代码如下：

```
01  //自定义接口
02  public interface OnPasswordInputFinish {
03      //添加密码输入完成接口
04      void inputFinish();
05      //取消支付接口
06      void outfo();
07      //忘记密码接口
08      void forgetPwd();
09  }
```

（6）在自定义 PasswordView 控件中为自定义键盘绑定单击事件。代码如下：

```
01  //为键盘按键添加单击事件
02  gridView.setOnItemClickListener(new AdapterView.OnItemClickListener {
03      @Override
04      public void onItemClick(AdapterView<?> parent, View view,
05          int position, long id) {
06          //单击0~9按钮
07          if (position < 11 && position != 9) {
08              //判断输入位置，要小心数组越界
09              if (currentIndex >= -1 && currentIndex < 5) {
10                  tvList[++currentIndex].setText(valueList.get(positi
11                          .get("name"));
12              }
13          } else {
14              //单击退格键
15              if (position == 11) {
16                  //判断是否删除完毕，要小心数组越界
17                  if (currentIndex - 1 >= -1) {
18                      tvList[currentIndex--].setText("");
19                  }
20              }
21          }
22      }
23  });
```

扩展学习

尝试在自定义 MainActivity 类中修改输入完密码后弹出输入的内容。

实例 015 带字母检索的全国城市列表

应用场景：城市选择

源码位置：光盘\Code\02\015

实例说明

在项目中经常会用到快速选择城市的功能，但是在定位不准确的情况下，就需要手动选择城市。本实例实现了按字母分类快速选择城市，单击城市名拼音的首字母，可切换到该字母所在的分类中，从而达到快速选择城市的目的。运行实例，将显示如图 2.8 所示的界面，向上滑动，可查看城市，如图 2.9 所示；单击右侧字母跳转到相应的字母对应的城市分类，如 2.10 所示。

图 2.8 向上滑动

图 2.9 单击右侧字母

图 2.10 切换到对应的城市分类

关键技术

关键技术

本实例主要使用 Canvas 类中的 drawText() 方法，ListView 类中的 onScroll() 方法、setSelection() 方法，Handler 类中的 removeCallbacks() 方法、postDelayed() 方法，实现检索城市列表功能。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“AZSort”，公司域名为“mingrisoft.com”。

（2）自定义控件，创建 MySideBar 类，让其继承 View 类，用于设置右侧显示字母。

（3）在“app\res\layout”目录下默认生成的 activity_main.xml 布局文件中，首先设置 ListVeiw 控件用于显示城市列表，然后设置自定义控件放到右侧。

（4）在“app\java\com\mingrisoft\azsort”文件夹下默认生成的 MainActivity 类中使用 setContentView() 方法实现在主视图布局中显示，并让 MainActivity 类继承 MySideBar.OnTouchingLetterChangeListener 接口，用于为右侧字母控件添加触摸方法。

使用 onScroll() 方法监听城市列表的滚动事件，实现设置列表上方固定的显示文字。代码如下：

```
//当ListView滚动时设置城市列表上方固定文字
mainList.setOnScrollListener(new OnScrollListener() {
    public void onScrollStateChanged(AbsListView view, int scrollState) {
    }
    //滑动的时候触发
    public void onScroll(AbsListView view, int firstVisibleItem,
          int visibleItemCount, int totalItemCount) {
       if(letterCharList.get(firstVisibleItem) >= 0){
          //设置显示文字
          tv01.setText(title[letterCharList.get(firstVisibleItem)]);
          lastFirstVisibleItem = firstVisibleItem;
       }else{
          if(lastFirstVisibleItem > firstVisibleItem){
             //设置显示文字
             tv01.setText(title[letterCharList.get(lastFirstVisibleItem) - 1]);
          }
       }
    }
});
```

实现右侧触摸监听接口，用于显示提示窗口，以及 1.5 秒后自动关闭提示窗口。代码如下：

```
//触摸监听
public void onTouchingLetterChanged(int s) {
    //切换到list指定的item位置
    mainList.setSelection(letterPositionList.get(s));
    //设置提示的字母
    overlay.setText(title[s]);
    //显示提示窗
    overlay.setVisibility(View.VISIBLE);
    //移除以前开启的线程
    handler.removeCallbacks(overlayThread);
    //1.5秒后执行overlayThread线程，隐藏选择字母提示
    handler.postDelayed(overlayThread, 1500);
}
//关闭提示信息线程
private class OverlayThread implements Runnable {
    @Override
    public void run() {
        //隐藏提示信息
        overlay.setVisibility(View.GONE);
    }
}
```

明日科技

扩展学习

在上面的实例中，我们将右侧选择字母的位置显示到中间位置，尝试修改提示窗口样式。

实例 016	下拉顶部图片放大效果 应用场景：个人中心页背景	源码位置：光盘\Code\02\016

实例说明

在微博 App 应用的“我的模块”中，下拉可以看到头像背景图片放大。本实例模仿该功能，实现下拉顶部图片放大。运行实例，初始效果如图 2.11 所示；向下滑动后图片放大，如图 2.12 所示。

图 2.11　向下滑动

图 2.12　顶部图片放大

关键技术

本实例主要使用 ScrollView 类中的 setOverScrollMode() 方法、ViewGroup 类中的 getChildAt() 方法、ScrollView 类中的 onTouchEvent() 方法、View 类中的 setLayoutParams() 方法，实现头像背景下拉放大效果。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“ImageFDScrollView”，公司域名为“mingrisoft.com”。

（2）导入资源文件：复制“光盘\Code\02\016\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）在“app\java\com\mingrisoft\imagefdscrollview”文件夹下创建 HeadZoomScrollview 控件类，让其继承 Scrollview 控件，用于放大顶部图片。

在自定义 HeadZoomScrollview 控件类中获取要放大的子控件。代码如下：

```
01  //获得默认第一个view
02  if (getChildAt(0) != null && getChildAt(0) instanceof ViewGroup && zoomView == null) {
03      ViewGroup vg = (ViewGroup) getChildAt(0);
04      if (vg.getChildCount() > 0) {
05          zoomView = vg.getChildAt(0);
06      }
07  }
```

在自定义 HeadZoomScrollview 控件中设置控件放大。代码如下：

```
01  //放大view
02  private void setZoom(float s) {
03      float scaleTimes = (float) ((zoomViewWidth+s)/(zoomViewWidth*1.0));
04      //如超过最大放大倍数，直接返回
05      if (scaleTimes > mScaleTimes) return;
06      ViewGroup.LayoutParams layoutParams = zoomView.getLayoutParams();
07      layoutParams.width = (int) (zoomViewWidth + s);
08      layoutParams.height = (int)(zoomViewHeight*((zoomViewWidth+s)/zoomViewWidth));
09      //设置控件水平居中
10      ((MarginLayoutParams) layoutParams).setMargins(-(layoutParams.width - zoomViewWidth) / 2, 0, 0, 0);
11      zoomView.setLayoutParams(layoutParams);
12  }
```

（4）在“app\res\layout”目录下默认生成的 activity_main.xml 布局文件中，添加自定义 HeadZoomScrollview 控件。代码如下：

```
01  <!--滑动放大控件-->
02  <com.mingrisoft.imagefdscrollview.HeadZoomScrollView
03      android:id="@+id/dzsv"
04      android:layout_width="match_parent"
05      android:layout_height="match_parent"
06      android:fillViewport="true"
07      android:scrollbars="none"
08      >
09  </com.mingrisoft.imagefdscrollview.HeadZoomScrollView>
```

扩展学习

在自定义 HeadZoomScrollview 控件中，尝试修改设置，使第二张图片滑动放大。

实例 017 可以每日签到的日历表

应用场景：日历签到功能

源码位置：光盘\Code\02\017

实例说明

OA 类型的 App 应用、论坛网站等都有签到功能。本实例通过在日历中进行标注，实现日历签到功能。运行实例，效果如图2.13 所示；单击“签到”按钮，会在日历中当天的位置添加“签到”印记，如图 2.14 所示；单击向左箭头或者向右滑动会切换到前一个月份，如图 2.15 所示。

图 2.13　单击“签到”按钮

图 2.14　添加签到印记

图 2.15　切换到前一个月份

关键技术

关键技术

本实例主要使用 SignCalendar 类中的 showCalendar() 方法、setCalendarDayBgColor() 方法，DBManager 类中的 add() 方法，View 类中的 setBackgroundResource() 方法，TextView 类中的 setEnabled() 方法，实现日历签到功能。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“CalendarsDemo”，公司域名为“mingrisoft.com”。

（2）导入资源文件：复制“光盘\Code\02\017\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）在“app\java\com\mingrisoft\calendarsdemo”文件夹下自定义控件，创建 SignCalendar 类继承 ViewFlippe，用于显示日历。

（4）在“app\res\layout”目录下默认生成的 activity_main.xml 布局文件中，添加自定义控件用于显示日历。

（5）在“app\java\com\mingrisoft\calendarsdemo”文件夹下默认生成的 MainActivity 类中，获取当前日期，设置需要的格式，代码如下：

```
01  //日期格式：2016-12-03
02  SimpleDateFormat formatter = new SimpleDateFormat("yyyy-MM-dd");
03  //获取当前时间
04  Date curDate = new Date(System.currentTimeMillis());
05  //设置时间格式为指定格式
06  date1 = formatter.format(curDate);
```

使用 add() 方法添加签到日期，并使用 query() 方法实现在签到日期上显示签到图案。代码如下：

```
01  //添加签到日期
02  public void add(String date) {
03      ArrayList<sqlit> persons = new ArrayList<sqlit>();
04      sqlit person1 = new sqlit(date, "true");
05      persons.add(person1);
06      dbManager.add(persons);
07  }
08  //在签到日期上显示签到图案
09  public void query() {
10      List<sqlit> persons = dbManager.query();
11      for (sqlit person : persons) {
12          list.add(person.date);
13          if (date1.equals(person.getDate())) {
14              isinput = true;
15          }
16      }
17      calendar.addMarks(list, 0);
18  }
```

监听当前月份，设置日历上方显示文字（即当前的年月）。代码如下：

```
01  //监听当前月份
02  calendar.setOnCalendarDateChangedListener(new SignCalendar.OnCalendarDateChangedListener() {
03      public void onCalendarDateChanged(int year, int month) {
04          popupwindow_calendar_month
05                  .setText(year + "年" + month + "月");
06      }
07  });
```

扩展学习

修改本实例，尝试监听日历上每一天的单击事件，并在单击事件中添加签到图片。

明日科技

实例 018 使用时间选择器设置提醒时间

应用场景：选择时间

源码位置：光盘\Code\02\018

实例说明

在需要设置提醒时间的 App 应用中，经常可以看到时间选择器的使用。本实例将通过系统自带的时间选择器设置提醒时间。运行实例，效果如图 2.16 所示；单击闹钟图标，弹出时间选择器，如图 2.17 所示；选择时间后，单击“确认”按钮，确认设置时间，显示到控件上，如图 2.18 所示。

图 2.16 单击闹钟图标

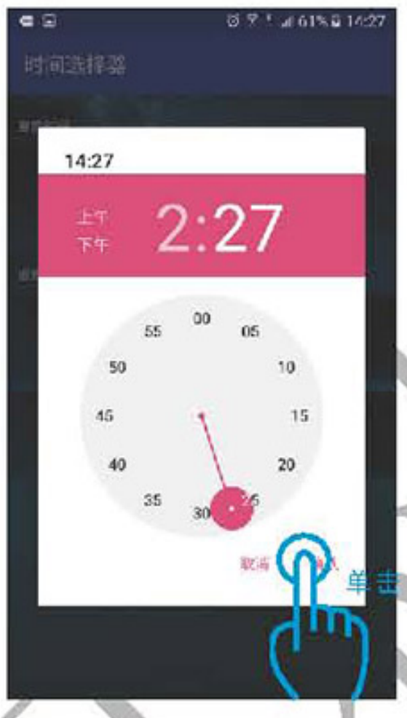


图 2.17 单击“确认”按钮

图 2.18 显示设置时间

关键技术

关键技术

本实例主要使用 SimpleDateFormat 类中的 SimpleDateFormat() 方法、Date 类中的 Date() 方法、TextView 类中的 setText() 方法、TimePickerDialog 类中的 TimePickerDialog() 方法，实现使用时间选择器设置提醒时间的功能。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“TimeDialog”，公司域名为“mingrisoft.com”。

（2）导入资源文件：复制“光盘\Code\02\018\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）在“app\java\com\mingrisoft\timedialog”文件夹下默认生成的 MainActivity 类中，按照自定义格式显示当前时间。代码如下：

```
01  //显示当前日期控件
02  TextView text = (TextView) findViewById(R.id.text);
03  //设置文本格式
04  SimpleDateFormat formatter = new SimpleDateFormat("yyyy年MM月dd日    HH:mm:ss");
05  //获取当前系统时间
```

```
Date curDate = new Date(System.currentTimeMillis()); //获取当前时间
//转换时间为指定格式
String str = formatter.format(curDate);
//设置显示文字
text.setText(str);
```

创建时间弹窗回调函数，设置选择后的时间显示到控件。代码如下：

```
//时间选择弹窗回调函数
TimePickerDialog.OnTimeSetListener timeListener =
        new TimePickerDialog.OnTimeSetListener() {
            @Override
            public void onTimeSet(TimePicker timerPicker,
                                  int hourOfDay, int minute) {
                //初始化显示设置时间控件
                TextView textV = (TextView) findViewById(R.id.textV);
                //小时
                String hourOfDays;
                //分
                String minutes;
                if (minute < 10) {
                    //为了美观，小于10补0
                    minutes = "0" + minute;
                } else {
                    minutes = minute + "";
                }
                if (hourOfDay < 10) {
                    //为了美观，小于10补0
                    hourOfDays = "0" + hourOfDay;
                } else {
                    hourOfDays = "" + hourOfDay;
                }
                //设置控件显示选择时间
                textV.setText(hourOfDays + ":" +
                        minutes);
            }
        };
```

创建时间选择弹窗，代码如下：

```
//时间选择弹窗
dialog = new TimePickerDialog(this, timeListener,
        calendar.get(Calendar.HOUR_OF_DAY),
        calendar.get(Calendar.MINUTE),
        //是否为24小时制
        false);
```

明日科技

扩展学习

在 MainActivity 类中，尝试通过修改 SimpleDateFormat() 方法，修改自己想要的时间格式，如“yyyyMMdd”。

实例 019	弹出智能提示信息的搜索框 应用场景：搜索框	源码位置：光盘\Code\02\019

实例说明

在具有搜索功能的 App 应用中，为了方便大家搜索，在输入内容时经常会有提示性的内容。本实例主要使用 AutoCompleteTextView 控件实现智能提示信息的功能。运行实例，将显示如图 2.19 所示的界面，单击搜索输入框，输入“书”，弹出相关提示信息，如图 2.20 所示；单击提示信息中的某一项，内容自动添加到搜索框内，如图 2.21 所示。

图 2.19　单击搜索输入框

图 2.20　单击提示信息

图 2.21　信息自动添加到搜索框

关键技术

关键技术

本实例主要使用自定义 MyAdapter 类中的 getView() 方法、View 类中的 inflate() 方法、TextView 类中的 setText() 方法、ArrayFilter 类中的 performFiltering() 方法、ArrayFilter 类中的 publishResults() 方法，实现智能提示信息的功能。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“AutoCompleteTextViewDemo”，公司域名为“mingrisoft.com”。

（2）导入资源文件：复制“光盘\Code\02\019\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）在“app\java\com\mingrisoft\autocompletetextviewdemo”文件夹下新建 MyAdapter 类，用于自定义适配器，实现加载提示信息到布局。

（4）在“app\res\layout”目录下默认生成的 activity_main.xml 布局文件中，添加 AutoCompleteTextView 控件。代码如下：

```
<!--输入提示控件-->
<AutoCompleteTextView
    android:id="@+id/autocomplete_country"
    android:layout_weight="9.5"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_centerVertical="true"
    android:background="@null"
    android:scrollbars="none"
    android:completionThreshold="1"
    android:textColor="#FFFFFF"
    android:lines="1"
    android:padding="10dp"
    />
```

（5）在“app\java\com\mingrisoft\autocompletetextviewdemo”文件夹下，创建 Book 实体类用于数据管理。

（6）在“app\java\com\mingrisoft\autocompletetextviewdemo”文件夹下，默认生成的 MainActivity 类中，使用自定义方法 initData()，模拟装载提示数据到集合。

初始化 MyAdapter 自定义适配器类，加载数据到 AutoCompleteTextVeiw 控件中。代码如下：

```
private void initView() {
    //找到相应的控件
    AutoCompleteTextView autoCompleteTextView
            = (AutoCompleteTextView) findViewById(R.id.autocomplete_country);
    //配置Adaptor
    bookAdapter = new MyAdapter(booknames,this);
    //填充数据到控件
    autoCompleteTextView.setAdapter(bookAdapter);
}
```

扩展学习

尝试在 MainActivity 类中修改提示信息的内容。

明日科技

实例 020 页面中垂直滚动的公告条

应用场景：公告条

源码位置：光盘\Code\02\020

实例说明

在明日学院 App 中可以看到首页上的垂直滚动新闻公告条。本实例模仿该新闻公告条，实现垂直滚动的公告条。运行实例，将显示如图 2.22 所示的界面；新闻公告条中的信息自动切换，如图 2.23 所示；单击新闻公告将弹出提示框显示新闻公告信息，如图 2.24 所示。

图 2.22 文字自动滚动

图 2.23 单击新闻公告

图 2.24 弹出公告信息

关键技术

关键技术

本实例主要使用 VerTextView 类中的 setListener() 方法、Toast 类中的 makeText() 方法、Paint 类中的 getTextBounds() 方法、Handler 类中的 sendEmptyMessageDelayed() 方法、Canvas 类中的 drawText() 方法。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“CZGDNew”，公司域名为“mingrisoft.com”。

（2）导入资源文件：复制“光盘\Code\02\020\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）在“app\java\com\mingrisoft\czgdnew”文件夹下，创建自定义控件 VerTextView 类，让其继承 View，用于显示滚动新闻公告。

（4）在“app\res\layout”目录下默认生成的 activity_main.xml 布局文件中，添加自定义 VerTextView 控件。

（5）在“app\java\com\mingrisoft\czgdnew”文件夹下默认生成的 MainActivity 类中，为自定义新闻

公告控件绑定触摸事件，实现弹出新闻公告内容。代码如下：

```
01  //绑定单击事件
02  tv_ver.setListener(new VerTextView.onTouchListener() {
03      @Override
04      public void touchListener(String s) {
05          //弹出提示信息
06          Toast.makeText(MainActivity.this, s, Toast.LENGTH_LONG).show();
07      }
08  });
```

扩展学习

尝试在自定义新闻公告控件中添加自己想要显示的公告内容。

实例 021　外星人手机时钟

应用场景：模拟时钟

源码位置：光盘\Code\02\021

实例说明

时钟是人们生活中不可或缺的一部分，通过时钟可以知晓时间。随着科技的进步，现在多数人都是通过手机来知晓时间的。本实例为自定义一款自己喜欢的时钟样式——外星人手机时钟。运行实例，初始效果如图 2.25 所示，一分钟后时钟的界面效果如图 2.26 所示。

图 2.25　秒针跳动

图 2.26　一分钟后时钟效果

关键技术

本实例主要使用 Paint 类中的 setTypeface() 方法、View 类中的 onMeasure() 方法、Canvas 类中的 scale() 方法、Drawable 类中的 setBounds() 方法、Canvas 类中的 rotate() 方法、Handler 类中的 postAtTime() 方法，实现一款自定义的手机时钟。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“Clock”，公司域名为“mingrisoft.com”。

（2）导入资源文件：复制“光盘\Code\02\021\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）在“app\java\com\mingrisoft\clock”文件夹下，自定义 MyClock 控件，用于显示钟表控件。

（4）在“app\res\layout”目录下默认生成的 activity_main.xml 布局文件中，添加自定义 MyClock 控件。代码如下：

```
<!--钟表控件-->
<com.mingrisoft.clock.MyClock
    android:id="@+id/analog_appwidget"
    style="@style/AppWidget"
    android:layout_width="match_parent"
    android:layout_height="match_parent"/>
```

（5）在自定义 MyClock 控件中，利用线程更新时间。代码如下：

```
//更新线程
mTicker = new Runnable() {
    public void run() {
        //判断是否退出线程
        if (mTickerStopped)
            //退出线程
            return;
        //事件改变方法
        onTimeChanged();
        //刷新控件
        invalidate();
        //获取当前系统时间
        long now = SystemClock.uptimeMillis();
        //计算下次需要更新的时间间隔
        long next = now + (1000 - now % 1000);
        //递归执行，达到秒针一直在动的效果
        mHandler.postAtTime(mTicker, next);
    }
};
mTicker.run();
```

明日科技

扩展学习

尝试在自定义 MyClock 控件中添加星期和日期的显示。

实例 022　加载大量文字时关键字变色

应用场景：显示大量文字

源码位置：光盘\Code\02\022
视频位置：光盘\Video\02\022

实例说明

在 App 开发中常常会碰到加载大量文字的情况，大多数时候，服务端发送给客户端的是带标签的文字，可以使用 WebView 控件加载并直接显示样式，但有的情况是直接返回大量文字。本实例就是在加载大量文字时为了方便用户观察，实现关键字变色。运行实例，将显示如图 2.27 所示的没有添加关键字变色的效果；添加关键字变色的效果如图 2.28 所示。

图 2.27　没有添加关键字变色效果

图 2.28　添加关键字变色效果

关键技术

本实例主要使用 TextView 类中的 setText() 方法、SpannableString 类中的 SpannableString () 方法、Pattern 类中的 compile() 方法，实现多个关键字变色的功能。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“KeyText”，公司域名为“mingrisoft.com”。
（2）导入资源文件：复制“光盘\Code\02\022\src”中的所有文件夹，粘贴到“app\res”目录下，

单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）在“app\res\layout”目录下默认生成的 activity_main.xml 布局文件中，添加 TextView 控件显示文本内容。代码如下：

```
<!--显示文本-->
<TextView
    android:id="@+id/tv"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:textSize="20dp"
    android:text="Hello World!" />
```

（4）在“app\java\com\mingrisoft\sjzlistview”文件夹下默认生成的 MainActivity 类中，创建要显示的文本内容。

（5）在“app\java\com\mingrisoft\keytext”文件夹下创建 KeywordUtil 类，并且新建 matcherSearchTitle() 方法，主要用于设置关键字变色。代码如下：

```
/**
 * 多个关键字高亮变色
 *
 * @param color
 *            变化的色值
 * @param text
 *            文字
 * @param keyword
 *            文字中的关键字数组
 * @return 处理后的文本内容
 */
public static SpannableString matcherSearchTitle(int color, String text,
                                                  String[] keyword) {
    //转换要处理的文本内容为SpannableString对象
    SpannableString s = new SpannableString(text);
    //循环关键字
    for (int i = 0; i < keyword.length; i++) {
        //获取匹配规则
        Pattern p = Pattern.compile(keyword[i]);
        //进行匹配，返回Matcher对象
        Matcher m = p.matcher(s);
        //循环匹配结果
        while (m.find()) {
            //获取匹配开始位置
            int start = m.start();
            //获取匹配结束位置
            int end = m.end();
            //设置字符中相应位置变色
            s.setSpan(new ForegroundColorSpan(color), start, end,
```

明日科技

```
30                      Spanned.SPAN_EXCLUSIVE_EXCLUSIVE);
31              }
32          }
33          return s;
34      }
```

扩展学习

尝试设置关键字显示为黄色。

实例 023　返回顶部按钮

应用场景：回到界面的顶部

源码位置：光盘\Code\02\023

实例说明

在浏览多条数据时，如果浏览到数据的底部，想返回到数据的顶部，只需要单击返回顶部按钮。本实例将实现在页面中添加“返回顶部”按钮。运行实例，当手指向下滑动时，显示“返回顶部”按钮，如图 2.29 所示，单击“返回顶部”按钮，将回到页面顶部，如图 2.30 所示。

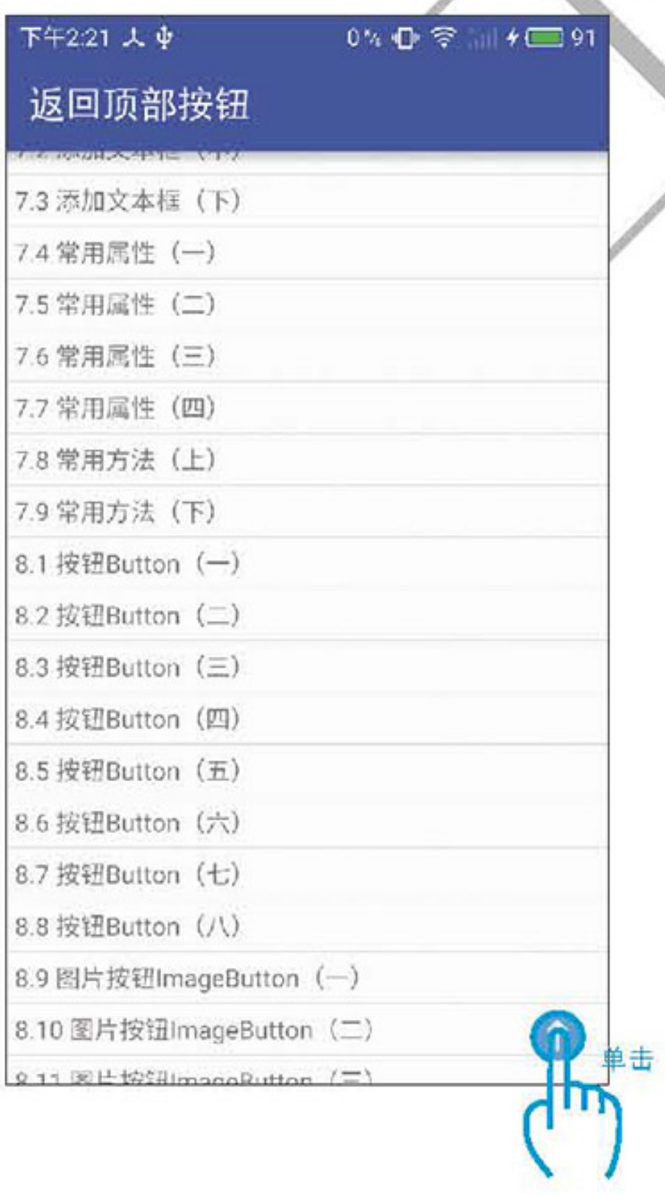


图 2.29　浏览数据底部

图 2.30　返回数据顶部

关键技术

本实例主要使用 ListView 类中的 smoothScrollToPosition() 方法和 View 类中的 bringToFront() 方法，实现设置 listview 回到顶部以及返回顶部按钮的显示与隐藏。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“TopButton”，公司域名为“mingrisoft.com”。

（2）导入资源文件：复制“光盘\Code\02\023\src”中的图片资源到“app\res”文件夹中的 mipmap 文件夹。

（3）在“app\res\values”文件夹中来创建名为 arrays.xml 的文件，该文件用于储存列表中的数据。

（4）创建 ListAdapter 类，用于显示 ListView 的数据。

（5）创建 GestureListener 类，主要用于手势的判断以及“返回顶部”按钮的显示与隐藏。主要代码如下：

```
@Override
public boolean onFling(MotionEvent e1, MotionEvent e2, float velocityX, float velocityY) {
    /*e1：第一个ACTION_DOWN MotionEvent
      e2：最后一个ACTION_MOVE MotionEvent
      velocityX：x轴上的移动速度（像素/秒）
      velocityY：y轴上的移动速度（像素/秒）*/
   if(null==e1){e1=downEvent;}
    if(e1==null||e2==null){return false;}
    float beginY = e1.getY();                                    //开始位置的y轴坐标
    float endY = e2.getY();                                      //结束位置的y轴坐标
    if (beginY - endY > distance && Math.abs(velocityY) > velocity) {        //上滑
        up(button);                                              //指定按钮隐藏
    } else if (endY - beginY > distance && Math.abs(velocityY) > velocity) { //下滑
        down(button);                                            //指定按钮显示
    }
    return false;
}
```

（6）在 MainActivity 类中，将 values 文件夹下的 arrays.xml 文件中的数据添加到适配器中并显示出来，关键代码如下：

```
TypedArray ar =getResources().obtainTypedArray(R.array.listTitle); //获取数据数组
int len = ar.length();                                           //获取数组的长度
List list  = new ArrayList<>();                                  //初始化List
for (int i = 0; i < len; i++)                                    //for循环添加数据
   list.add(i,ar.getString(i));                                  //向ListView中添加数据
adapter.addData(list);                                           //将数据传给adapter
```

（7）单击置顶按钮时，执行 ListView 滑动到顶部，代码如下：

```
imageBtn.setOnClickListener(new View.OnClickListener() { //返回顶部按钮的单击事件
    @Override
```

明日科技

```
03        public void onClick(View v) {
04            listView.smoothScrollToPosition(0);                    //使ListView返回顶部
05        }
06    });
```

扩展学习

本实例使用 View 类中的 bringToFront() 方法来实现按钮的隐藏与显示，请尝试使用其他方法来实现该效果。

明日科技

第3章 数据储存操作

实例 024 使用 SharePreferences 保存用户信息

应用场景：App 登录时保存用户信息，免登录功能

源码位置：光盘\Code\03\024

视频位置：光盘\Video\03\024

实例说明

在 App 应用中，用户登录成功后经常需要保存用户的信息，便于在应用中调用。例如再次进入登录页将显示上次登录的名称，或者是用于自动登录。本实例主要使用 SharedPreferences 类实现保存用户信息。运行实例，初始效果如图 3.1 所示；输入账号名和密码，单击“保存用户信息”按钮，弹出“信息保存成功！”提示，如图 3.2 所示；信息保存成功后，单击“显示保存信息”按钮，会弹出显示保存的用户信息，如图 3.3 所示。

图 3.1 初始效果

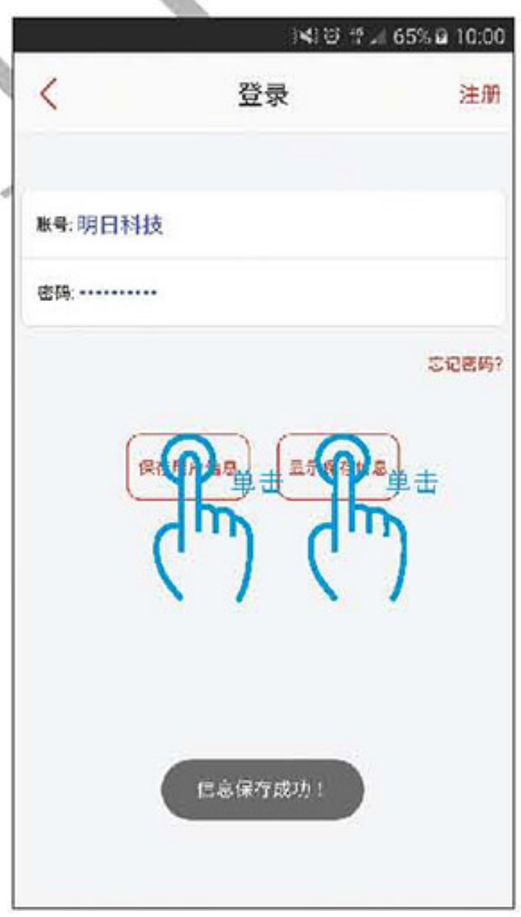


图 3.2 保存用户信息

图 3.3 显示保存的用户信息

关键技术

本实例主要使用 Toast 类中的 makeText() 方法，TextUtils 类中的 isEmpty() 方法，ContextWrapper 类中的 getSharedPref erences() 方法，SharedPreferences 类中的 putString() 方法、getString() 方法，实现保存与显示用户信息的功能。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“SharedPreferences”，公司域名为“mingrisoft.com”。

（2）导入资源文件：复制“光盘\Code\03\024\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）在“app\java\com\mingrisoft\sharedpreferences”目录中默认生成的 MainActivity 类中，初始化 EditText 控件，获取控件上输入的登录信息，并判断控件上内容是否为空。代码如下：

```
01  //获取控件上输入的内容
02  username = et_username.getText().toString().trim();
03  password = et_password.getText().toString().trim();
04  if (TextUtils.isEmpty(username) || TextUtils.isEmpty(password)) {
05      Toast.makeText(this, "账号或密码不可为空", Toast.LENGTH_SHORT).show();
06  }
```

实例化 SharedPreferences 对象，并通过该对象保存用户信息，保存成功后弹出提示。代码如下：

```
01  //实例化SharedPreferences对象
02  mySharedPreferences= getSharedPreferences("myuserinfo",
03          MainActivity.MODE_PRIVATE);
04  //实例化SharedPreferences.Editor对象（第二步）
05  SharedPreferences.Editor editor = mySharedPreferences.edit();
06  //用putString的方法保存数据
07  editor.putString("username", username);
08  editor.putString("pwd", password);
09  //提交当前数据
10  editor.commit();
11  Toast.makeText(this, "信息保存成功！", Toast.LENGTH_SHORT).show();
```

实例化 SharedPreferences 对象，并通过该对象获取保存的用户信息，弹出提示框显示保存的用户信息。代码如下：

```
01  //实例化SharedPreferences对象
02  mySharedPreferences = getSharedPreferences("myuserinfo", MainActivity.MODE_PRIVATE);
03  //使用getString方法获得value，注意第2个参数是value的默认值
04  String username = mySharedPreferences.getString("username", "没有保存的数据");
05  String pwd = mySharedPreferences.getString("pwd", "没有保存的数据");
06  AlertDialog.Builder builder = new AlertDialog.Builder(
07          MainActivity.this);
08  builder.setTitle("显示保存的用户信息");
```

```
09  builder.setMessage("账号："+username+"\n"+"密码："+pwd);
10  builder.setCancelable(false);
11  builder.setPositiveButton("知道了！", new DialogInterface.OnClickListener() {
12      public void onClick(DialogInterface dialog, int which) {
13          dialog.cancel();
14      }
15  });
16  builder.create().show();
```

扩展学习

尝试实现免输入账号功能，在保存用户信息后，再次进入应用自动添加账号和密码信息。

实例 025　保存信息到 SD 卡

应用场景：文件储存数据

源码位置：光盘\Code\03\025

实例说明

在 App 应用中，经常会以文档形式保存信息到 SD 卡中。本实例主要使用 File 类创建并保存文件信息。实例运行，初始效果如图 3.4 所示；输入内容后，单击“内容写入 SD 卡文档”按钮，弹出“写入文件成功”提示信息，如图 3.5 所示；内容保存成功后，单击右上角按钮，弹出提示框显示文件的内容，如图 3.6 所示。

图 3.4　初始效果

图 3.5　内容写入 SD 卡文档

图 3.6　显示 SD 卡文档信息

关键技术

本实例主要使用 File 类中的 File() 方法、Toast 类中的 makeText() 方法、FileOutputStream 类中的 FileOutputStream() 方法、FileInputStream 类中的 FileInputStream() 方法、AlertDialog 类中的 Builder() 方法。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“FileInfo”，公司域名为“mingrisoft.com”。

（2）导入资源文件：复制“光盘\Code\03\025\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）在“app\java\com\mingrisoft\fileinfo”目录中默认生成的 MainActivity 类中，使用 File 类，在 SD 卡根目录创建 filedemo.txt 文件。代码如下：

```
01  //初始化文件 在根目录创建目录为filedemo的txt文件
02  file = new File(Environment.getExternalStorageDirectory(), FILENAME);
```

保存输入的信息到 SD 卡上创建的文件中。代码如下：

```
01      //写入文件到SD卡
02      private void writesd() {
03          if (Environment.getExternalStorageState().equals(
04                  Environment.MEDIA_MOUNTED)) {
05              try {
06                  //写入文件流
07                  FileOutputStream fos = new FileOutputStream(file);
08                  fos.write(et1.getText().toString().getBytes());
09                  //关闭写入流
10                  fos.close();
11                  //弹出提示信息
12                  Toast.makeText(MainActivity.this, "写入文件成功",
13                          Toast.LENGTH_LONG).show();
14              } catch (Exception e) {
15                  //弹出提示信息
16                  Toast.makeText(MainActivity.this, "写入文件失败",
17                          Toast.LENGTH_SHORT).show();
18              }
19          } else {
20              //此时SD卡不存在或者不能进行读写操作
21              Toast.makeText(MainActivity.this,
22                      "此时SD卡不存在或者不能进行读写操作", Toast.LENGTH_SHORT).show();
23          }
24      }
```

读取 SD 卡上所创建文件的内容，并弹出提示框显示。代码如下：

```
01  //读取SD卡上的文件
02  private void readsd() {
03      if (Environment.getExternalStorageState().equals(
04              Environment.MEDIA_MOUNTED)) {
05          try {
06              FileInputStream inputStream = new FileInputStream(file);
07              byte[] b = new byte[inputStream.available()];
08              //读取文件流
09              inputStream.read(b);
10              String nr = new String(b);
11              if (!TextUtils.isEmpty(nr)) {
12                  AlertDialog.Builder builder = new AlertDialog.Builder(
13                          MainActivity.this);
14                  builder.setTitle("读取文档信息");
15                  builder.setMessage("文档内容：" + nr);
16                  builder.setCancelable(false);
17                  builder.setPositiveButton("知道了！", new DialogInterface.OnClickListener() {
18                      public void onClick(DialogInterface dialog, int which) {
19                          dialog.cancel();
20                      }
21                  });
22                  builder.create().show();
23              } else {
24                  //弹出提示信息
25                  Toast.makeText(MainActivity.this, "文档内容为空,请输入要保存信息！",
26                          Toast.LENGTH_SHORT).show();
27              }
28
29          } catch (Exception e) {
30              //弹出提示信息
31              Toast.makeText(MainActivity.this, "读取失败",
32                      Toast.LENGTH_SHORT).show();
33          }
34      } else {
35          //此时SD卡不存在或者不能进行读写操作
36          Toast.makeText(MainActivity.this,
37                  "此时SD卡不存在或者不能进行读写操作", Toast.LENGTH_SHORT).show();
38      }
39  }
```

扩展学习

尝试修改在 SD 卡上保存文件的名称。

实例 026	记事本 应用场景：记事、便签	源码位置：光盘\Code\03\026

实例说明

手机中自带的应用，例如便签、备忘录或者记事本等都是比较实用的工具，便于记录并查看信息。本实例将实现一个具有记事本功能的手机应用。运行实例，初始效果如图 3.7 所示；单击“+”按钮，进入新增记事页面，如图 3.8 所示；输入要记录的内容，然后单击右上角的“√”按钮，保存记录的内容，并自动跳转到首页显示记录内容，如图 3.9 所示。

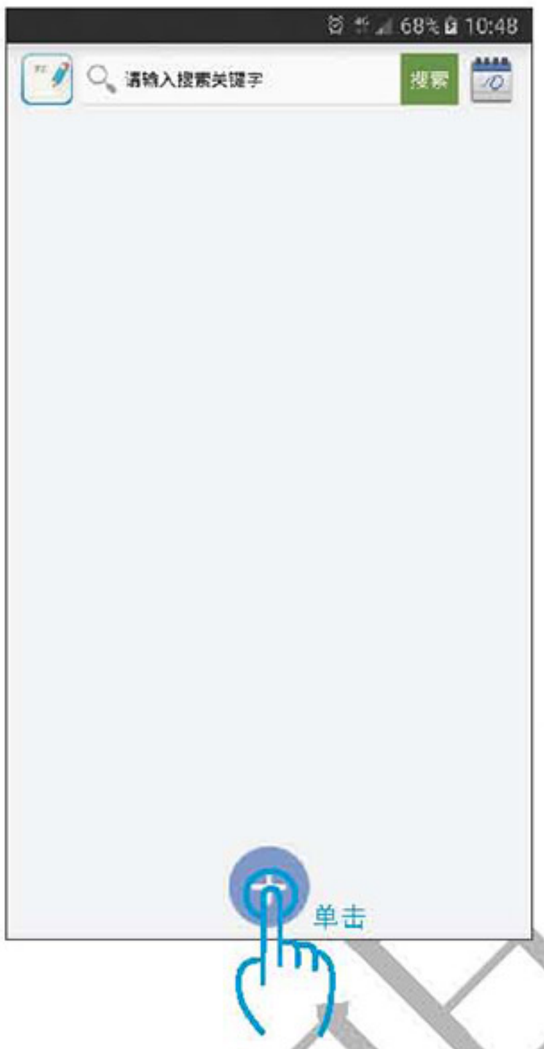


图 3.7　单击“+”按钮

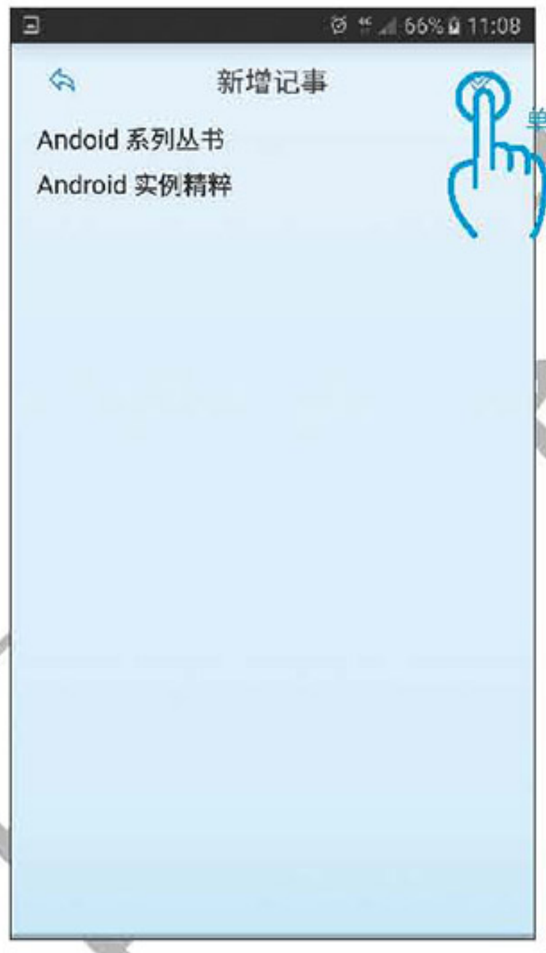


图 3.8　单击“对号”按钮

图 3.9　列表显示保存信息

关键技术

本实例主要使用 SQLiteDatabase 类中的 execSQL() 方法、Toast 类中的 makeText() 方法、Date 类中的 Date() 方法、SQLiteDatabase 类中的 openOrCreateDatabase() 方法，实现记事本的功能。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“Notes”，公司域名为“mingrisoft.com”。

（2）导入资源文件：复制“光盘\Code\03\026\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）在“app\java\com\mingrisoft\notes”目录中创建 db 文件夹，并在其中创建自定义数据库工具类 DatabaseOperation，用于保存记事。

（4）在“app\java\com\mingrisoft\notes”目录中默认生成的 MainActivity 类中，查询数据库显示列表信息。代码如下：

```
//读取保存信息
private void readsd() {
    if (Environment.getExternalStorageState().equals(
            Environment.MEDIA_MOUNTED)) {
        try {
            FileInputStream inputStream = new FileInputStream(file);
            byte[] b = new byte[inputStream.available()];
            //读取文件流
            inputStream.read(b);
            String nr = new String(b);
            if (!TextUtils.isEmpty(nr)) {
                AlertDialog.Builder builder = new AlertDialog.Builder(
                        MainActivity.this);
SQLBean bean = new SQLBean();                    //创建数据库实体类
//保存日记信息id到实体类
bean.set_id(" " + cursor.getInt(cursor.getColumnIndex("_id")));
bean.setContext(cursor.getString(cursor
      .getColumnIndex("context")));              //保存日记内容到实体类
//保存日记标题到实体类
bean.setTitle(cursor.getString(cursor.getColumnIndex("title")));
//保存日记记录时间到实体类
bean.setTime(cursor.getString(cursor.getColumnIndex("time")));
//保存日记是否设置提醒时间到实体类
bean.setDatatype(cursor.getString(cursor
      .getColumnIndex("datatype")));
//保存日记提醒时间到实体类
bean.setDatatime(cursor.getString(cursor
      .getColumnIndex("datatime")));
//保存日记是否设置了日记锁到实体类
bean.setLocktype(cursor.getString(cursor
      .getColumnIndex("locktype")));
//保存日记锁密码到实体类
            bean.setLock(cursor.getString(cursor.getColumnIndex("lock")));
            list.add(bean);                              //把保存日记信息实体类保存到日记信息集合里
        }
        //倒序显示数据
        Collections.reverse(list);
        adapter = new MainAdapter(list, this);       //装载日记信息到首页
        lv_notes.setAdapter(adapter);                //为日记列表设置日记信息适配器
    }
    dop.close_db();                                  //关闭数据库
    if (list.size()==0){
        lv_notes.setVisibility(View.GONE);
```

```
44      }else {
45          lv_notes.setVisibility(View.VISIBLE);
46      }
47  }
```

设置记事列表长按的事件，代码如下：

```
01  //记事列表长按监听器
02  class ItemLongClickEvent implements OnItemLongClickListener {
03      @Override
04      public boolean onItemLongClick(AdapterView<?> parent, View view,
05                                     int position, long id) {
06          //初始化日记id保存控件
07          tv_note_id = (TextView) view.findViewById(R.id.tv_note_id);
08          //初始化是否添加日记锁保存控件
09          tv_locktype = (TextView) view.findViewById(R.id.tv_locktype);
10          //初始化日记锁密码保存信息
11          tv_lock = (TextView) view.findViewById(R.id.tv_lock);
12          //获取控件上是否设置日记锁信息
13          String locktype = tv_locktype.getText().toString();
14          //获取控件上日记密码信息
15          String lock = tv_lock.getText().toString();
16          //获取控件上id信息转换成int类型
17          int item_id = Integer.parseInt(tv_note_id.getText().toString());
18          //调用弹出选择操作框的方法
19          simpleList(item_id, locktype, lock);
20          return true;
21      }
22  }
```

扩展学习

尝试为单个记事添加密码锁，输入密码进入查看页。

实例 027　查看手机中的图片

应用场景：浏览手机中图片

源码位置：光盘\Code\03\027

实例说明

在 App 应用中，经常会调用手机中的图片，以实现一些特殊效果。本实例将实现获取手机中包含图片的文件夹并显示。运行实例，获取手机中包含图片的文件夹的效果如图 3.10 所示；单击某一个图片文件夹，进入图片列表显示页，查看该文件夹中的图片，如图 3.11 所示；向上滑动，就可以查看其余图片，如图 3.12 所示。

明日科技

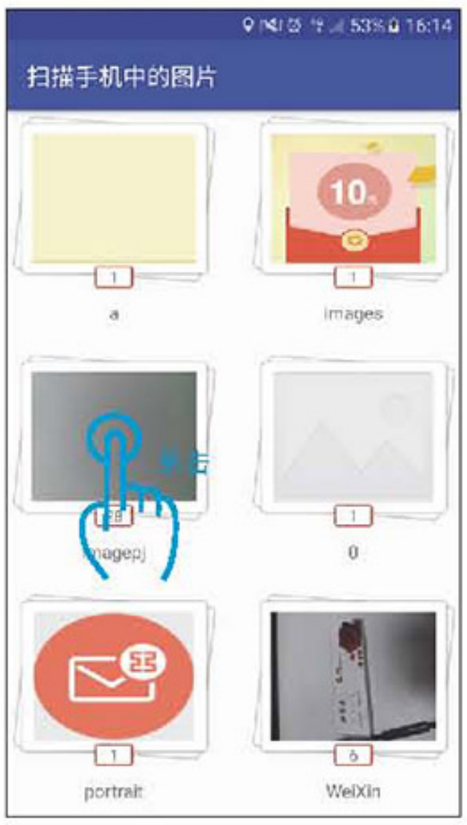


图 3.10　初始效果

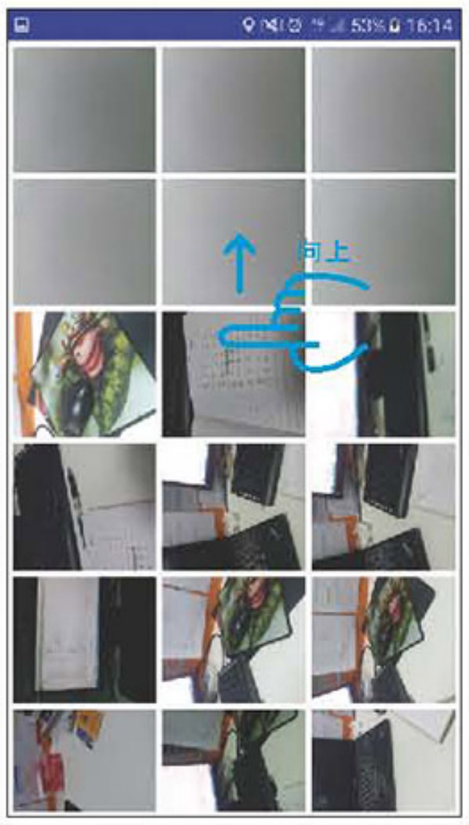


图 3.11　查看手机图片

图 3.12　查看其余图片

关键技术

本实例主要使用 ContentResolver 类中的 query() 方法，HashMap 类中的 put() 方法、get() 方法、Handler 类中的 sendEmptyMessage() 方法，ProgressDialog 类中的 show() 方法，实现查看手机图片的功能。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“ContentProvider”，公司域名为“mingrisoft.com”。

（2）导入资源文件：复制“光盘\Code\03\027\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）在“app\java\com\mingrisoft\contentprovider”目录中默认生成的 MainActivity 类中，利用 ContentProvider 类查看手机中的图片，此操作需要在子线程中运行。代码如下：

```
01  new Thread(new Runnable() {
02      @Override
03      public void run() {
04          Uri mImageUri = MediaStore.Images.Media.EXTERNAL_CONTENT_URI;
05          ContentResolver mContentResolver = MainActivity.this.getContentResolver();
06          //只查询jpeg和png的图片
07          Cursor mCursor = mContentResolver.query(mImageUri, null,
08                  MediaStore.Images.Media.MIME_TYPE + "=? or "
09                          + MediaStore.Images.Media.MIME_TYPE + "=?",
10                  new String[] { "image/jpeg", "image/png" },
11                  MediaStore.Images.Media.DATE_MODIFIED);
12
13          while (mCursor.moveToNext()) {
14              //获取图片的路径
```

```
            String path = mCursor.getString(mCursor
                .getColumnIndex(MediaStore.Images.Media.DATA));
            //获取该图片的父路径名
            String parentName = new File(path).getParentFile().getName();
            //根据父路径名将图片放入到mGruopMap中
            if (!mGruopMap.containsKey(parentName)) {
                List<String> chileList = new ArrayList<String>();
                chileList.add(path);
                mGruopMap.put(parentName, chileList);
            } else {
                mGruopMap.get(parentName).add(path);
            }
        }
        mCursor.close();
    }
}
```

查询完图片后发送消息，通知更新数据。代码如下：

```
//通知Handler扫描图片完成
mHandler.sendEmptyMessage(SCAN_OK);
```

接收消息更新显示数据。代码如下：

```
//用于接收消息更新页面
private Handler mHandler = new Handler(){
    @Override
    public void handleMessage(Message msg) {
        super.handleMessage(msg);
        switch (msg.what) {
            case SCAN_OK:
            //关闭进度条
            mProgressDialog.dismiss();
            //初始化适配器
            adapter = new GroupAdapter(MainActivity.this, list = subGroupOfImage(mGruopMap), mGroupGridView);
            //绑定适配器
            mGroupGridView.setAdapter(adapter);
            break;
        }
    }
};
```

扩展学习

尝试在 ShowImageActivity 类中为图片列表添加单击事件，实现单击后放大显示单张图片。

明日科技

实例 028 家庭账务记录

应用场景：记账功能

源码位置：光盘\Code\03\028

实例说明

利用手机 App 实现记录家庭账务，可有效掌控家庭财务收支情况。本实例主要使用第三方开源的 jxl.jar 包完成对 Excel 文档的操作，包括可以在 SD 卡中查看 Excel 文档，也可以在项目中单击“导出账单”查看记录。运行实例，效果如图 3.13 所示；输入数据后单击“写入账单”按钮，写入成功如图 3.14 所示；单击“导出账单”，查看账单列表，如图 3.15 所示。

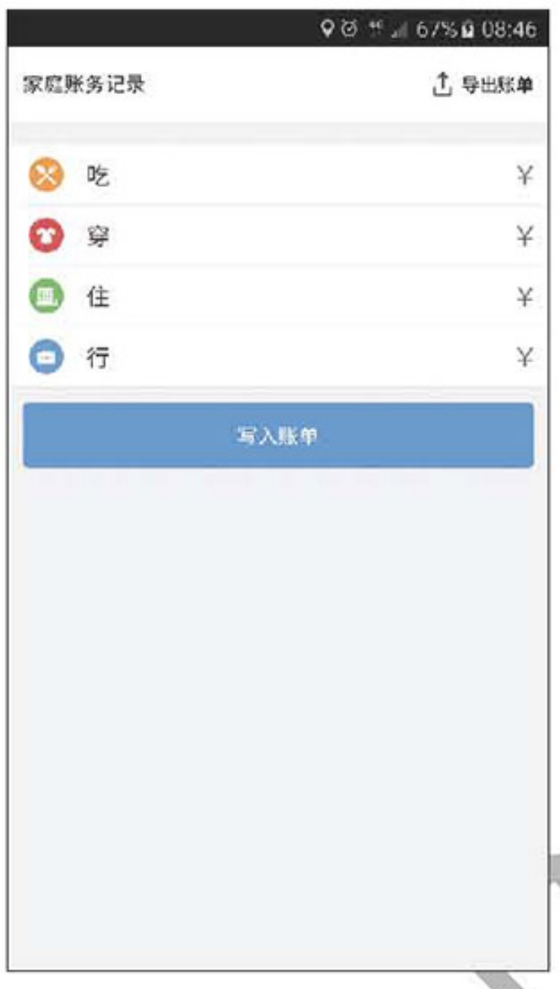


图 3.13 运行效果

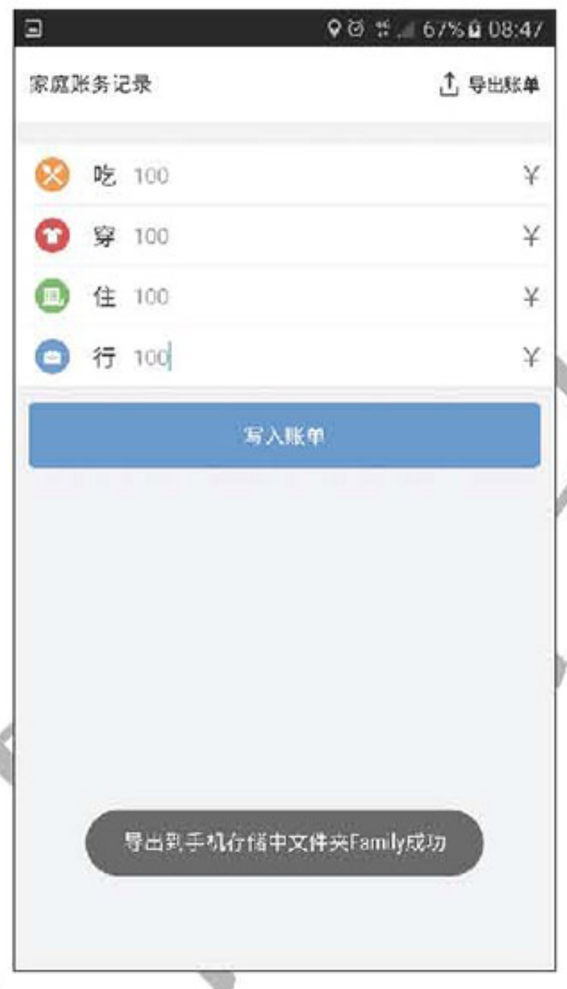


图 3.14 单击“写入账单”按钮

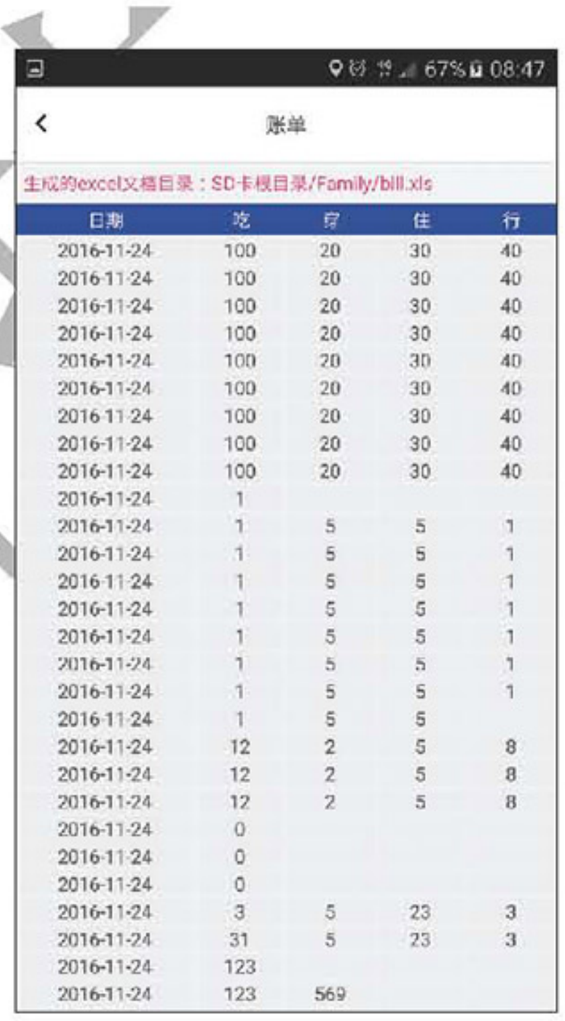


图 3.15 显示保存的账单信息

关键技术

本实例主要使用 SQLiteDatabase 类中的 execSQL() 方法、SQLiteDatabase 类中的 insert() 方法、SQLiteDatabase 类中的 query() 方法、SQLiteDatabase 类中的 rawQuery() 方法、Label 类中的 Label() 方法，实现记录及查看家庭账务记录功能。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“ASExcel”，公司域名为“mingrisoft.com”。

（2）导入资源文件：复制“光盘\Code\03\028\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）导入资源文件：复制“光盘\Code\03\028\src\libs\”文件夹下的 jxl.jar 包，粘贴到“app\ibs”目录下，该文件为开源的 Jar 包用于生成 Excel 表格。

（4）在“app\java\com\mingrisoft\asexcel\db”文件夹下创建 DBHelper 类，让其继承 SQLiteOpenHelper 类，作为自定义创建数据库的工具类。

（5）在“app\java\com\mingrisoft\asexcel”文件夹下默认生成的 MainActivity 类中，保存信息到数据库。代码如下：

```
//获取输入信息添加到信息数组
saveData = new String[]{
        new SimpleDateFormat("yyyy-MM-dd").format(new Date()),
        mFoodEdt.getText().toString().trim(),
        mArticlesEdt.getText().toString().trim(),
        mTrafficEdt.getText().toString().trim(),
        mTravelEdt.getText().toString().trim()};
if (canSave(saveData)) {
    //ContentValues 和HashTable类似都是一种存储的机制，但是两者最大的区别就在于：
    //contenvalues只能存储基本类型的数据，像string、int之类的数据
    //创建ContentValues储存数据
    ContentValues values = new ContentValues();
    values.put("time",
            new SimpleDateFormat("yyyy-MM-dd").format(new Date()));
    values.put("food", mFoodEdt.getText().toString());
    values.put("use", mArticlesEdt.getText().toString());
    values.put("traffic", mTrafficEdt.getText().toString());
    values.put("travel", mTravelEdt.getText().toString());
    //添加数据到数据库中
    long insert = mDbHelper.insert("family_bill", values);
    //判断数据添加是否成功
    if (insert > 0) {
        //创建Excel文件，保存数据
        initData();
    }
} else {
    Toast.makeText(this, "请填写任意一项内容", Toast.LENGTH_SHORT).show();
}
```

（6）在“app\java\com\mingrisoft\asexcel\familybill”文件夹下创建自定义适配器 BillAdapter 类，让其继承 BaseAdapter 类，用于装载数据，并加载数据库的内容。

（7）在“app\java\com\mingrisoft\asexcel”文件夹下创建 UserInfoActivity 类，用于显示数据库中的信息，并获取数据绑定适配器。代码如下：

```
//获取数据添加到数组中
ArrayList<BillObject> billList = (ArrayList<BillObject>) ExcelUtils
        .read2DB(new File(getSDPath() + "/Family/bill.xls"), this);
//绑定适配器
bill_listview.setAdapter(new BillAdapter(this, billList));
```

明日科技

扩展学习

尝试在 Excel 文档中多删除一项内容，删除完成后查看文档，查看是否删除数据成功。

实例 029	课程表 应用场景：应用课程信息	源码位置：光盘\Code\03\029

实例说明

为方便工作学习，应用市场中出现了越来越多方便大家的应用，比如课程表。本实例主要用于记录及查看课程。运行实例，效果如图 3.16 所示；单击要添加课程的位置，进入课程录入页面，如图 3.17 所示；输入课程信息后，单击“保存”按钮，返回课程表页面，如图 3.18 所示。

图 3.16　运行效果

图 3.17　单击“保存”按钮

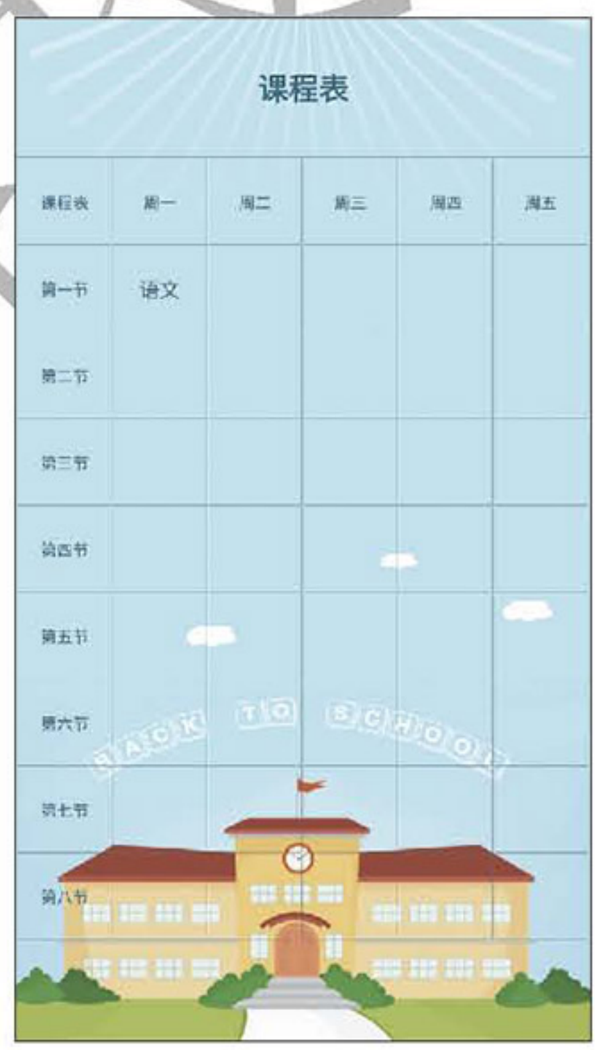


图 3.18　显示课程表

关键技术

关键技术

本实例主要使用 SQLiteDatabase 类中的 execSQL() 方法、Canvas 类中的 drawLine() 方法、Toast 类中的 makeText() 方法、TextView 类中的 setText() 方法、SQLiteDatabase 类中的 openOrCreateDatabase() 方法，实现记录及查看课程功能。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“MySchedule”，公司域名为“mingrisoft.com”。

（2）导入资源文件：复制“光盘\Code\03\029\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）在“app\java\com\mingrisoft\myschedule”文件夹下创建 LineGridView 类，让其继承 GridView 类，用于自定义颜色列表，显示课程信息。

（4）在“app\java\com\mingrisoft\myschedule”文件夹下默认生成的 MainActivity 类中，创建数据表。代码如下：

```
try {
    //创建表
    db.execSQL("create table schedule ("
          + "sid integer PRIMARY KEY autoincrement, "
          + "date  text, "
          + "one text, "
          + "two text, "
          + "three text, "
          + "four text, "+"five text,"+"six text,"+"seven text,"+"eight text);");
    String sql="create table course(lid integer PRIMARY KEY autoincrement,coursename text,
          teacher text,location text,email text);";
    db.execSQL(sql);
```

（5）在“app\java\com\mingrisoft\myschedule”文件夹下创建 AseAdapter 类，让其继承 BaseAdapter 类，作为自定义适配器，用于装载课程信息到列表。

（6）在“app\java\com\mingrisoft\myschedule”文件夹下创建 AseDialog 类，让其继承 Dialog 类，用于自定义弹窗，写入课程信息。

扩展学习

在课程显示列表页，为课程添加老师邮箱。

实例 030　查看手机 SD 卡中所有文件

应用场景：查看 SD 卡内文件

源码位置：光盘\Code\03\030

实例说明

文件查看器可以获取 SD 卡根目录中的所有文件信息，并且可以根据文件类型使用系统中的应用查看图片、音乐、视频、txt 文本等资源。本实例将通过 File 文件类来实现查看手机内存卡所有文件的功能。运行实例，将自动获取 SD 卡中的所有文件，如图 3.19 所示；单击要查看的目标文件，选择文件的打开方式，如图 3.20 所示；单击文件的打开方式，打开指定的文件，如图 3.21 所示。

明日科技

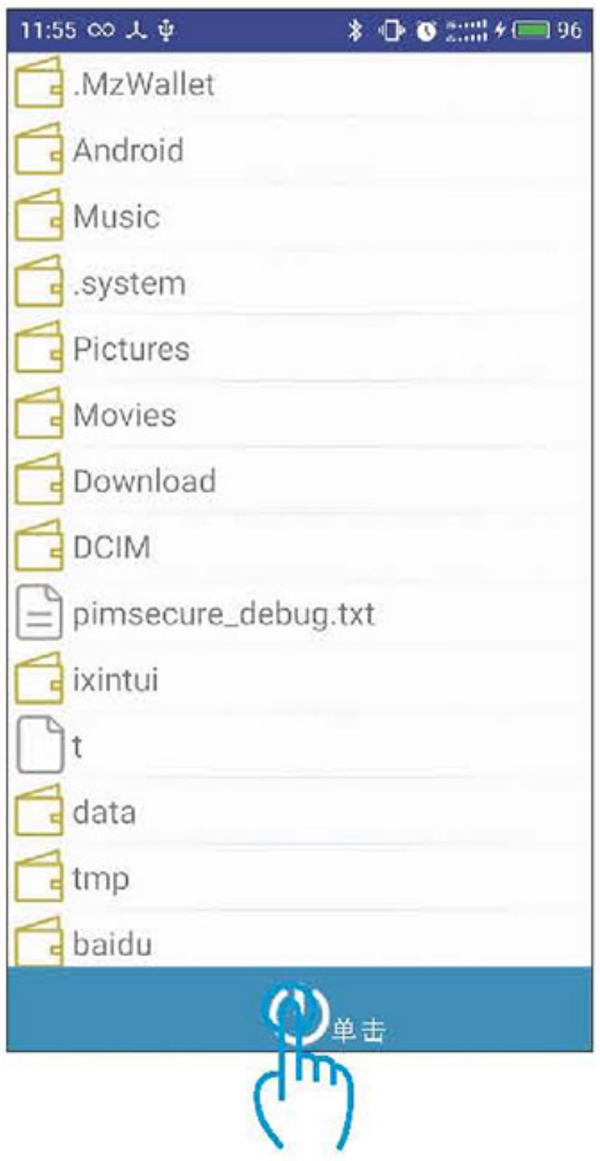


图 3.19　显示应用列表

图 3.20　打开指定的应用程序

图 3.21　使用应用程序打开指定文件

关键技术

关键技术

本实例主要使用 File 类及 Environment 类中的 getExternalStorageDirectory() 方法、ListActivity 类中的 setListAdapter() 方法，实现文件查看器的功能。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“FileView”，公司域名为“mingrisoft. com”。

（2）导入资源文件：复制“光盘\Code\03\030\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）打开 MainActivity 类，让该类继承 ListActivity 类，并实现 View.OnClickListener 接口，创建全局变量保存 SD 卡的根目录。代码如下：

```
01  //SD卡根目录
02  private String lSDCard = Environment.getExternalStorageDirectory().toString();
```

通过 initFileListInfo() 方法实现根据指定的目录，获取目录中的所有文件。代码如下：

```
01  private void initFileListInfo(String filePath) {
02      isAddBackUp = false;                          //将添加返回按键标识设置为false
03      lCurrentFilePath = filePath;                  //当前路径
04      lFileName = new ArrayList<String>();          //创建保存显示文件名称的列表
```

```
05      lFilePaths = new ArrayList<String>();  //创建保存显示的文件路径的列表
06      File mFile = new File(filePath);
07      //遍历该路径下的所有文件和文件夹
08      File[] mFiles = mFile.listFiles();
09      initAddBackUp(filePath, lSDCard);
10      //将所有文件信息添加到集合中
11      for (File mCurrentFile : mFiles) {
12          lFileName.add(mCurrentFile.getName());
13          lFilePaths.add(mCurrentFile.getPath());
14      }
15      //适配显示的文件名称与路径
16      setListAdapter(new FileAdapter(MainActivity.this, lFileName, lFilePaths));
17  }
```

（4）在“FileView\app\src\main\res”目录下创建 xml 包，在该包中创建 file_paths.xml 文件，用于指定文件路径。代码如下：

```
01  <paths>
02      <external-path path="Android/data/com.mingrisoft.fileview/" name="files_root" />
03      <external-path path="." name="external_storage_root" />
04  </paths>
```

（5）打开“FileView\app\src\main\AndroidManifest.xml”文件，在该文件中添加 provider 标签，用于添加 Uri 权限以及加载 file_paths.xml 文件。代码如下：

```
01  <provider
02      android:name="android.support.v4.content.FileProvider"
03      android:authorities="com.mingrisoft.fileview"
04      android:grantUriPermissions="true"
05      android:exported="false">
06      <meta-data
07          android:name="android.support.FILE_PROVIDER_PATHS"
08          android:resource="@xml/file_paths" />
09  </provider>
```

（6）在 MainActivity 类中通过 openFile() 方法实现根据打开文件的类型，获取可以打开该文件的相关应用。代码如下：

```
01  private void openFile(File file) {
02      //获取文件类型匹配
03      MimeTypeMap mime = MimeTypeMap.getSingleton();
04      //获取当前文件的名称
05      String ext = file.getName().substring(file.getName().lastIndexOf(".") + 1);
06      //根据文件名称匹配MIME类型
07      String type = mime.getMimeTypeFromExtension(ext);
```

```
08      try {
09          Intent intent = new Intent();
10          //设置Intent的Action属性
11          intent.setAction(Intent.ACTION_VIEW);
12          //判断系统版本
13          if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.N) {
14              //设置Intent标志
15              intent.setFlags(Intent.FLAG_GRANT_READ_URI_PERMISSION);
16              Uri contentUri = FileProvider.getUriForFile(this,
17                      "com.mingrisoft.fileview", file);
18              //设置数据与类型
19              intent.setDataAndType(contentUri, type);
20          } else {
21              intent.setDataAndType(Uri.fromFile(file), type);
22          }
23          //启动跳转
24          startActivityForResult(intent,ACTIVITY_VIEW_ATTACHMENT);
25      } catch (ActivityNotFoundException anfe) {
26          Toast.makeText(this, "没有可以打开当前文件的应用！", Toast.LENGTH_LONG).show();
27      }
28  }
```

扩展学习

本实例通过 File 文件类实现了文件查看的功能，请尝试将当前文件的根目录显示在文件列表的上方。首先在 activity_main.xml 布局文件中添加用于显示文件根目录的 TextView 控件，然后在 MainActivity 类中获取该控件，并在 initFileListInfo() 方法中实现当前文件根目录的显示。

实例 031 创建与删除文件

应用场景：SD 卡内文件及文件夹的创建与删除

源码位置：光盘\Code\03\031

实例说明

创建与删除文件应用主要实现在手机的 SD 卡中创建（删除）文件夹或文本文件，并且刷新当前目录中的文件列表。本实例将通过 File 文件类实现文件的创建与删除。运行实例，在出现的界面中，单击“创建”按钮将弹出选择创建文件夹或文件的对话框，输入文件夹名称，如图 3.22 所示；单击“创建”按钮，将创建对应的文件或文件夹，同时列表自动刷新，如图 3.23 所示；长按要删除的文件夹或文件，将显示如图 3.24 所示的确认删除对话框，单击“确定”按钮，将删除该文件夹或文件。

图 3.22　显示创建对话框

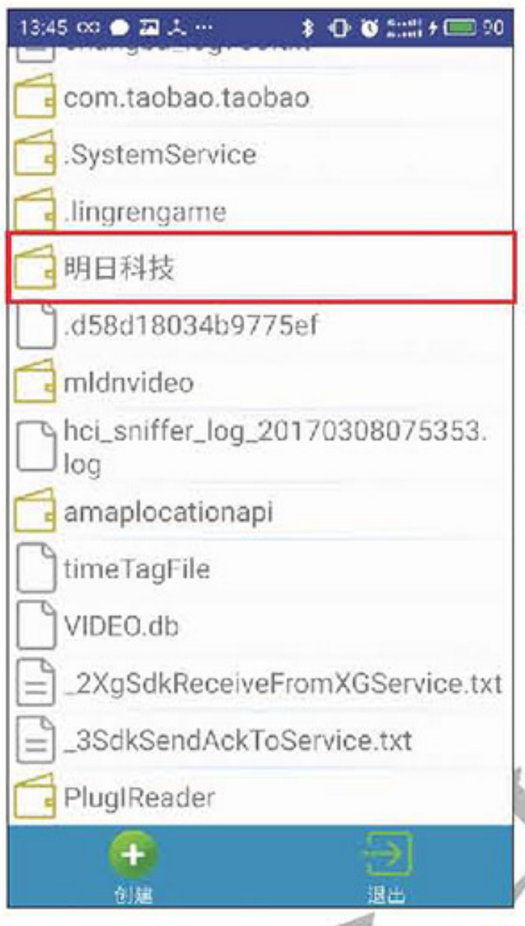


图 3.23　显示创建后的文件夹

图 3.24　显示删除文件提示对话框

关键技术

本实例主要使用 RadioGroup.OnCheckedChangeListener 接口中的 onCheckedChanged() 方法、File 类中的 createNewFile() 方法、AlertDialog.Builder 类，实现文件的创建与删除功能。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“FileCreationAndDeletion”，公司域名为“mingrisoft. com”。

（2）导入资源文件：复制“光盘\Code\03\031\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）打开 MainActivity 类，让该类继承自 ListActivity 并实现 View.OnClickListener 和 AdapterView.OnItemLongClickListener 接口。然后创建 createFolder() 方法，并在该方法中实现创建文件的确认对话框，用于文件夹或文本文件的创建。代码如下：

```
01  AlertDialog.Builder lBuilder = new AlertDialog.Builder(MainActivity.this)
02          .setView(linearLayout)
03          .setPositiveButton("创建", new DialogInterface.OnClickListener() {
04              public void onClick(DialogInterface dialog, int which) {
05                  //获取用户输入的名称
06                  lNewFolderName = ((EditText) linearLayout.
07                          findViewById(R.id.new_filename)).getText().toString();
08                  if (isChecked == 1) {
09                      try {
10                          //设置文件路径："当前路径"+ "/" + "文件名称" + ".txt"
```

```
                    lCreateFile = new File(lCurrentFilePath +
                            java.io.File.separator + lNewFolderName + ".txt");
                    lCreateFile.createNewFile();          //创建文本文件
                    //刷新当前目录文件列表
                    initFileListInfo(lCurrentFilePath);
                } catch (IOException e) {
                    Toast.makeText(MainActivity.this, "文件名拼接出错!",
                            Toast.LENGTH_SHORT).show();
                }
            } else if (isChecked == 2) {
                lCreateFile = new File(lCurrentFilePath + java.io.File.
                        separator + lNewFolderName);
                if (!lCreateFile.exists() && !lCreateFile.isDirectory() &&
                        lNewFolderName.length() != 0) {
                    if (lCreateFile.mkdirs()) {
                        //刷新当前目录文件列表
                        initFileList中的Info(lCurrentFilePath);
                    } else {
                        Toast.makeText(MainActivity.this, "创建失败, " +
                                "可能是系统权限不够！", Toast.LENGTH_SHORT).show();
                    }
                } else {
                    Toast.makeText(MainActivity.this, "文件名为空或是重名",
                            Toast.LENGTH_SHORT).show();
                }
            }
        }).setNeutralButton("取消", null);
lBuilder.show();
```

创建 deleteDialog() 方法实现确认删除的对话框。代码如下：

```
private void deleteDialog(final File file) {
    new AlertDialog.Builder(MainActivity.this)
            .setTitle("提示!")
            .setMessage("您确定要删除吗?")
            .setPositiveButton("确定", new DialogInterface.OnClickListener() {
                public void onClick(DialogInterface dialog, int which) {
                    if (file.isFile()) {
                        //如果是文本文件则直接删除
                        file.delete();
                    } else {
                        //如果是文件夹则用deleteFolder()方法删除
                        deleteFolder(file);
                    }
                    //重新遍历该文件的父目录
```

```
                    initFileListInfo(file.getParent());
                }
            })
            .setNegativeButton("取消", null).show();
}
```

创建 deleteFolder() 方法实现删除文件夹的功能。代码如下：

```
public void deleteFolder(File folder) {
    File[] fileArray = folder.listFiles();                    //创建文件夹内文件数组
    if (fileArray.length == 0) {                              //如果没有文件
        //空文件夹则直接删除
        folder.delete();
    } else {
        //遍历该目录
        for (File currentFile : fileArray) {
            if (currentFile.exists() && currentFile.isFile()) {
                //文件则直接删除
                currentFile.delete();
            } else {
                //递归删除
                deleteFolder(currentFile);
            }
        }
        folder.delete();
    }
}
```

扩展学习

本实例通过 File 文件类实现了文件的创建与删除功能，请尝试实现单击“返回”按钮，弹出确认退出对话框功能。打开 MainActivity 类，在该类中调用 onKeyDown() 方法，在该方法中实现单击“返回”按钮时，通过对话框确认退出应用的功能。

实例 032　搜索应用程序

应用场景：名称搜索

源码位置：光盘\Code\03\032

实例说明

搜索应用程序是通过输入应用程序的名称或其中的一个字符，实现应用程序的搜索功能。本实例将通过 SearchView.OnQueryTextListener 接口来实现应用程序的搜索功能。运行实例，将显示手机中所有的应用程序，如图 3.25 所示，单击右上角“搜索”按钮，将开启搜索功能，如图 3.26 所示；输入需

明日科技

要搜索应用中的字符，将显示搜索到的含有该字符的应用名称，如图 3.27 所示。

图 3.25　显示手机中所有应用

图 3.26　开启搜索功能

图 3.27　搜索应用名称

关键技术

本实例主要使用 SearchView.OnQueryTextListener 接口中的 onQueryTextSubmit() 方法、onQueryTextChange() 方法，String 类中的 contains() 方法，实现搜索应用程序的功能。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“SearchApplication”，公司域名为“mingrisoft. com”。

（2）导入资源文件：复制“光盘\Code\03\032\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）在“SearchApplication\app\src\main\java”目录中创建 util 包，该包用于保存本项目中所使用到的工具类文件，该包中包括两个类文件和一个接口文件，其名称及作用如表 3.1 所示。

表 3.1　util 包中的 3 个文件

名　　称	描　　述
AppInfo 类	用于保存应用程序的具体信息
Utils 类	该类为开发工具类，包含获取应用程序具体信息的集合、时间转化函数、打开应用方法、卸载应用方法等
IUninstall 接口	用于实现应用程序的卸载接口

在 Utils 类中创建 getSearchResult() 方法实现获取搜索的结果。代码如下：

```
01  public static List<AppInfo> getSearchResult(List<AppInfo> list, String keyword) {
02      List<AppInfo> searchResultList = new ArrayList<AppInfo>();    //返回集合
03      for (int i = 0; i < list.size(); i++) {                        //循环遍历
04          AppInfo app = list.get(i);                                 //获取单个的实体类
05          //用关键字和应用信息实体类比较
06          if (app.appName.toLowerCase().contains(keyword.toLowerCase())) {
07              searchResultList.add(app);                             //将搜索结果添加到结果集
08          }
09      }
10      return searchResultList;
11  }
```

（4）在 SearchApplication\app\src\main\java 目录中创建 adapter 包，在该包中创建 MyAdapter 类。该类继承自 BaseAdapter，是显示应用程序信息列表的适配器。

（5）打开 MainActivity 类，让该类继承 Activity 类，并实现 AdapterView.OnItemClickListener、Iuninstall 和 SearchView.OnQueryTextListener 接口。通过 onQueryTextSubmit() 方法实现单击搜索按钮后执行搜索任务。代码如下：

```
01  public boolean onQueryTextSubmit(String query) {
02      //提交关键字
03      Utils.toast(this, "您查询的关键字是 : " + query.trim());
04      Utils.KEY = query.trim();                                      //删除空白字符
05      list = Utils.getSearchResult(allList, query);                  //根据关键字生成结果
06      update_sort();                                                 //重新排序
07      return true;
08  }
```

通过 onQueryTextChange() 方法实现修改需要搜索的应用名称时，立即显示相匹配的应用程序。代码如下：

```
01  public boolean onQueryTextChange(String newText) {
02      Utils.KEY = newText.trim();                                    //删除空白字符
03      list = Utils.getSearchResult(allList, newText.trim());         //根据关键字生成结果
04      update_sort();                                                 //重新排序并更新
05      return true;
06  }
```

扩展学习

本实例通过 SearchView.OnQueryTextListener 接口实现了应用程序的搜索功能，请尝试长按列表中的应用名称，实现跳转到当前应用的信息界面。首先为 MainActivity 类添加 AdapterView.OnItemLongClickListener 接口，然后在 onCreate() 方法中设置 ListView 的长按事件，最后重写 onItemLongClick() 方法，实现长按列表中应用名称后跳转到当前应用设置的信息界面。

实例 033 文件的复制和粘贴

应用场景：SD 卡内文件的复制与粘贴

源码位置：光盘\Code\03\033

实例说明

在手机的 SD 卡中，可以实现各种文件的复制与粘贴，并且刷新当前目录中的文件列表。本实例将通过 File 文件类来实现文件的复制与粘贴。运行实例，单击要复制的文件，将弹出确认文件复制的对话框，如图 3.28 所示；在指定的粘贴目录下单击“按钮”，如图 3.29 所示。

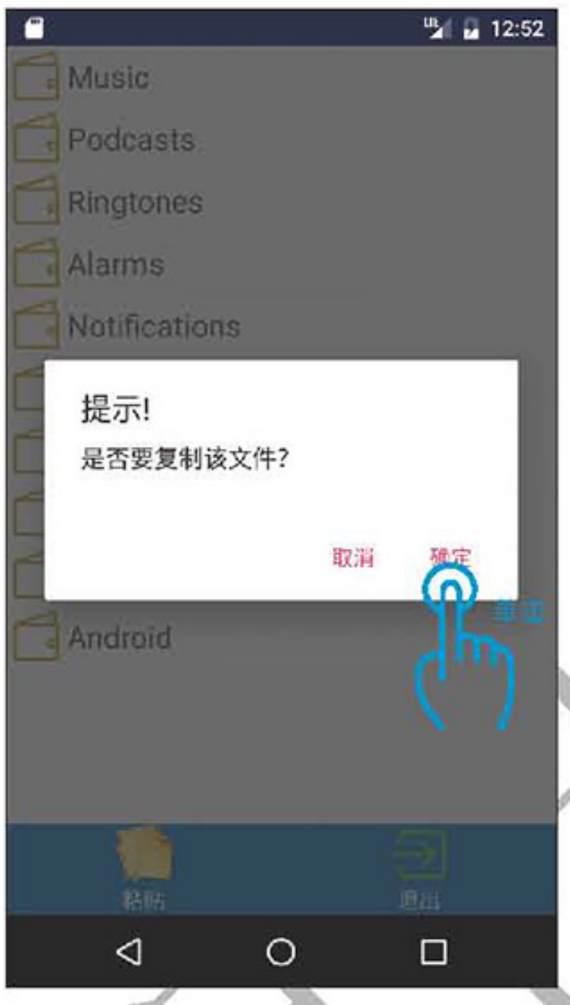


图 3.28 文件复制对话框

图 3.29 复制的文件

关键技术

本实例主要使用 File 类中的 isDirectory() 方法、listFiles() 方法、getPath() 方法、getName() 方法，实现文件的复制与粘贴功能。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“FileCopyAndPaste”，公司域名为“mingrisoft. com”。

（2）导入资源文件：复制“光盘\Code\03\033\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）打开 MainActivity 类，使该类继承自 ListActivity 类，并实现 View.OnClickListener 和 Adapter View.OnItemLongClickListener 接口。在该类中创建 copyDialog() 方法，用于实现确认复制对话框的功能。代码如下：

```
01  private void copyDialog(final File file) {
02      new AlertDialog.Builder(MainActivity.this)
03              .setTitle("提示!")
04              .setMessage("是否要复制该文件？")
05              .setPositiveButton("确定", new DialogInterface.OnClickListener() {
06                  public void onClick(DialogInterface dialog, int which) {
07                      //复制标志，表明已复制文件
08                      isCopy = true;
09                      //取得复制文件的名字
10                      mCopyFileName = file.getName();
11                      //记录复制文件的路径
12                      mOldFilePath = lCurrentFilePath + java.io.File.separator + mCopyFileName;
13                      Toast.makeText(MainActivity.this, "已复制!", Toast.LENGTH_SHORT).show();
14                      //重新遍历该文件的父目录
15                      initFileListInfo(file.getParent());
16                  }
17              })
18              .setNegativeButton("取消", null).show();
19  }
```

创建 onPaste() 方法，用于实现“粘贴”按钮事件的处理。代码如下：

```
01  private void onPaste() {
02      if (!mOldFilePath.equals(mNewFilePath) && isCopy == true) {  //在不同路径下复制才生效
03          //得到新路径
04          mNewFilePath = lCurrentFilePath + java.io.File.separator + mCopyFileName;
05          copyFile(mOldFilePath, mNewFilePath);                    //调用复制方法
06          initFileListInfo(lCurrentFilePath);                      //刷新当前目录的文件列表
07      } else {
08          Toast.makeText(MainActivity.this, "未复制文件！", Toast.LENGTH_LONG).show();
09      }
10  }
```

创建 copyFile() 方法，在该方法中判断如果复制的是目录，就将目录中的子目录进行递归，如果复制的是文件，直接调用 CopySdcardFile() 方法进行粘贴。代码如下：

```
01  if (from.isDirectory()) {
02      //如果存在，则获取当前目录下的全部文件填充到数组中
03      currentFiles = from.listFiles();
04      //目标目录
05      File targetDir = new File(toFile);
06      //创建目录
07      if (!targetDir.exists()) {
08          targetDir.mkdirs();
09      }
10      //遍历要复制目录下的全部文件
```

明日科技

```
    for (int i = 0; i < currentFiles.length; i++) {
        if (currentFiles[i].isDirectory())          //如果是子目录就进行递归
        {
            copyFile(currentFiles[i].getPath() + "/", toFile + "/"
                    + currentFiles[i].getName() + "/");
        } else                                      //如果是文件，就粘贴文件
        {
            CopySdcardFile(currentFiles[i].getPath(), toFile
                    + "/" + currentFiles[i].getName());
        }
    }
} else {                                            //如果是文件，就直接粘贴
    CopySdcardFile(fromFile, toFile);
}
```

创建 CopySdcardFile() 方法，在该方法中实现文件或目录的粘贴功能。代码如下：

```
public int CopySdcardFile(String fromFile, String toFile) {
    try {
        //需要复制文件目录的输入流
        InputStream fos = new FileInputStream(fromFile);
        //文件粘贴目录的输出流
        OutputStream to = new FileOutputStream(toFile);
        byte bt[] = new byte[1024];
        int c;
        //根据复制文件目录写入新的粘贴目录
        while ((c = fos.read(bt)) > 0) {
            to.write(bt, 0, c);
        }
        fos.close();
        to.close();
        return 0;
    } catch (Exception ex) {
        return -1;
    }
}
```

扩展学习

本实例通过 File 文件类实现了文件的复制与删除功能，请尝试实现文件或文件夹重新命名的功能。首先在“app\src\main\res\layout”目录中创建 rename_dialog.xml 文件，作为显示修改名称对话框的布局文件，然后修改 MainActivity 类中的 copyDialog() 方法，用于实现在复制文件的对话框中添加重新命名的功能，最后添加 renameDialog() 方法，用于显示修改名称的对话框。

第4章 第三方支持

实例034 使用 Bmob 第三方后台数据

应用场景：自定义后台数据

源码位置：光盘\Code\04\034

实例说明

本实例将介绍使用第三方已经搭建好的后台数据，将手机联系人数据同步到第三方的后台，再从第三方后台将联系人数据同步到其他的手机上。运行实例，后台数据如图 4.1 所示。

objectId String	num String	name String	createdAt Date	updatedAt Date
c4bc912cfd	158••••4520	王华	2017-06-05 13:30:26	2017-06-05 13:30:26
30a11d257a	130••••9418	王女士__南湖游泳区湖光小区	2017-06-05 13:30:26	2017-06-05 13:30:26
6b2dcf582c	183••••2686	张亦弛	2017-06-05 13:30:26	2017-06-05 13:30:26
c554aab71c	188••••7452	张艳梅.	2017-06-05 13:30:26	2017-06-05 13:30:26
c9c366ceee	186••••8423	王盛鑫	2017-06-05 13:30:26	2017-06-05 13:30:26
bff9b5fc43	183••••2696	我的小粉丝儿	2017-06-05 13:30:26	2017-06-05 13:30:26
4b388db97b	+86186••••9928	余婷婷	2017-06-05 13:30:26	2017-06-05 13:30:26
4575d23cef	183••••0744	田强	2017-06-05 13:30:26	2017-06-05 13:30:26
db29a4e079	153••••9700	老妹（程）	2017-06-05 13:30:26	2017-06-05 13:30:26
0dc7d2c47d	05798••••900	众泰售后服务	2017-06-05 13:30:26	2017-06-05 13:30:26

图 4.1　上传到服务器的数据

关键技术

关键技术

本实例主要使用 Bomb 服务，要使用该服务，需要先申请 Bmob 账号，并创建应用和表。具体步骤如下。

（1）打开 Bmob 官方网站（http://www.bmob.cn），并注册账号。

（2）注册好账号之后，进入“我的控制台”，在控制台中创建应用，如图 4.2 所示。

图 4.2　创建应用

（3）输入要创建项目的名称，并选择项目的类型，然后单击“创建应用”按钮，如图 4.3 所示。

图 4.3　应用信息

（4）单击创建的项目名称，进入详情界面，在详情界面添加一张表，表的名称为“Content”，如图 4.4 所示。单击“创建表”按钮，在进入的页面中，再在创建的表中添加两列，分别命名为“name”和“num”。

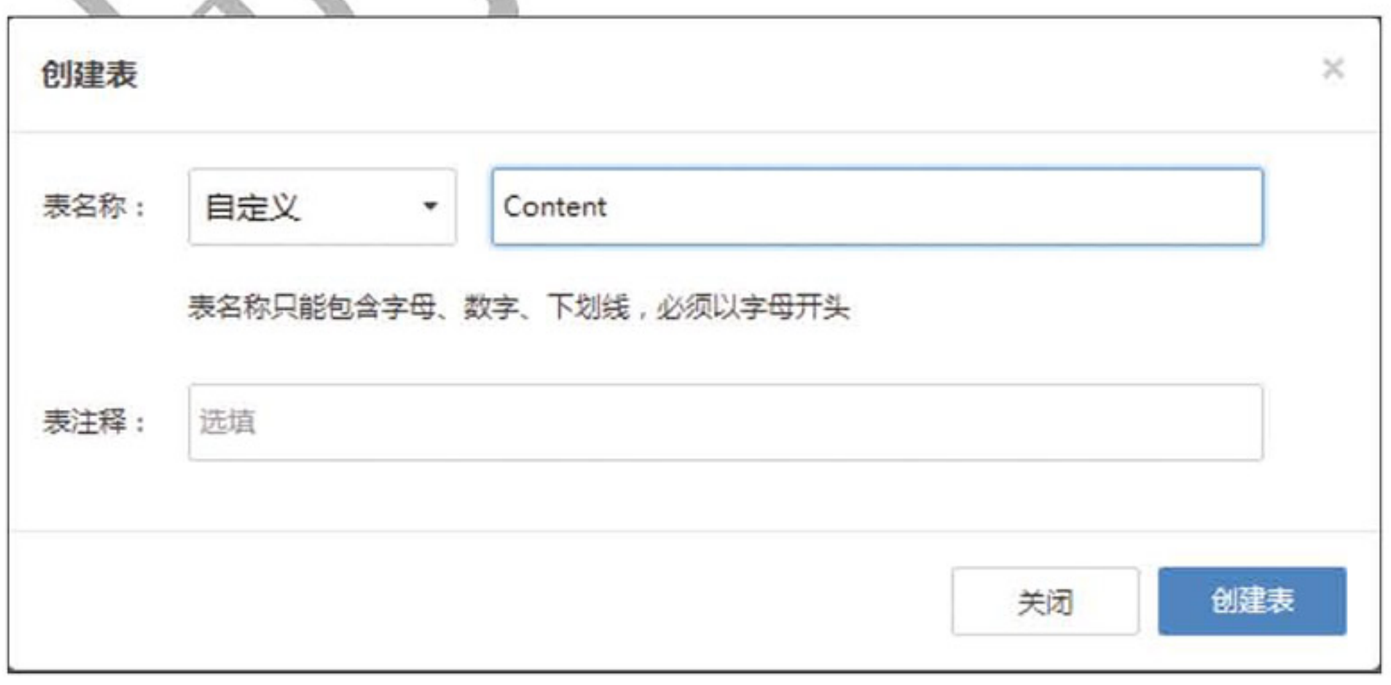


图 4.4　创建表

实现过程

（1）打开 Android Studio，新建一个项目，命名为“Bmob”，公司域名为“mingrisoft.com”。

（2）打开 Bmob 的官网，申请 Appkey，并下载 Android 版的 SDK。

（3）在“app\libs”文件夹中拷入相关的 Jar 包，用于手机 App 与 Bmob 后台进行数据交互。

（4）创建名称为 Content 的实体类，让该类继承 BmobObject 类，并且让该类中的字段与创建表中的字段保持一致，注意大小写。

（5）在 MainActivity 类中获取手机中的联系人数据，关键代码如下：

```
//获取手机联系人
Cursor phoneCursor = resolver.query(ContactsContract.CommonDataKinds.Phone.CONTENT_URI,
        PHONES_PROJECTION, null, null, null);
if (phoneCursor != null) {
    while (phoneCursor.moveToNext()) {
        //获取手机号码
        String phoneNumber = phoneCursor.getString(1);
        //当手机号码为空的或者为空字段 跳过当前循环
        if (TextUtils.isEmpty(phoneNumber))
            continue;
        //获取联系人名称
        String contactName = phoneCursor.getString(0);
        //创建实体类解析联系人
        Contact mContact = new Contact(contactName, phoneNumber);
        list.add(mContact);                                    //添加到集合里
    }
    phoneCursor.close();                                       //关闭游标对象
```

（6）实现将获取到的联系人发送到后台，关键代码如下：

```
/**
 * 向后台发送联系人数据
 */
obj.setName(name);                                             //联系人姓名
obj.setNum(number);                                            //联系人电话
obj.save(new SaveListener<String>() {                          //向后台发送的回调方法
    @Override
    public void done(String s, BmobException e) {
        Log.e("------", s);                                    //发送成功后的回调可以再次数添加
    }
});
```

（7）Bomb 官方提供了很多从服务器获取数据的方法，本实例将根据时间获取服务器的数据，并将获取到的数据添加到手机中，关键代码如下：

```
/**
 * 根据时间从后台获取数据
```

明日科技

```
 **/
BmobQuery<Contact> query = new BmobQuery<Contact>();                    //初始化
List<BmobQuery<Contact>> and = new ArrayList<BmobQuery<Contact>>();
//大于00：00：00
BmobQuery<Contact> q1 = new BmobQuery<Contact>();
String start = "2017-01-01 13:47:42";                                   //设置获取时的时间
SimpleDateFormat sdf = new SimpleDateFormat("yyyy-MM-dd HH:mm:ss");
Date date = null;
try {
    date = sdf.parse(start);
} catch (ParseException e) {
    e.printStackTrace();
}
q1.addWhereGreaterThanOrEqualTo("createdAt", new BmobDate(date));
and.add(q1);
query.and(and);                                                         //添加获取数据的条件
query.findObjects(new FindListener<Contact>() {
    @Override
    public void done(List<Contact> list, BmobException e) {
        /**
         * 将获取到的后台数据用for循环添加到手机中
         **/
        for (int i = 0; i < list.size(); i++) {
            testAddContacts(list.get(i).getName(), list.get(i).getNum());
        }
    }
});
```

扩展学习

本实例从手机中获取联系人后，向 Bmob 的后台提交数据，从而实现数据的交互，想一想如何向 Bmob 的后台插入单条数据。

实例 035　第三方登录

应用场景：借助第三方平台实现登录

源码位置：光盘\Code\04\035

实例说明

多数 App 应用会借助第三方平台实现登录。使用第三方登录输入用户名和密码，借助第三方平台即可获取用户的相关资料。本实例将借助第三方平台，实现微博登录。运行实例，将显示如图 4.5 所

示的登录界面；如果选择 QQ 登录方式，将显示正在使用 QQ 登录的界面，如图 4.6 所示；如果选择微博登录方式，将显示正在使用微博登录的界面，如图 4.7 所示。

图 4.5　登录界面

图 4.6　QQ 登录

图 4.7　微博登录

关键技术

本实例主要使用第三方平台——友盟社会化分享服务，下面介绍如何使用该服务。

（1）打开友盟的官方网站（ http://www.umeng.com），注册账号。

（2）账号注册完成后，需要去腾讯或新浪平台申请第三方的账号。

（3）申请完第三方平台账号之后，回到友盟，进入社交类的 U-Share 分享，选择 Android 的集成文档，如图 4.8 所示。

图 4.8　应用信息

（4）单击下载 SDK，友盟的 SDK 会给用户提供 Demo。当用户只开发登录界面时，如图 4.9 所示的类对用户来说是必要的。

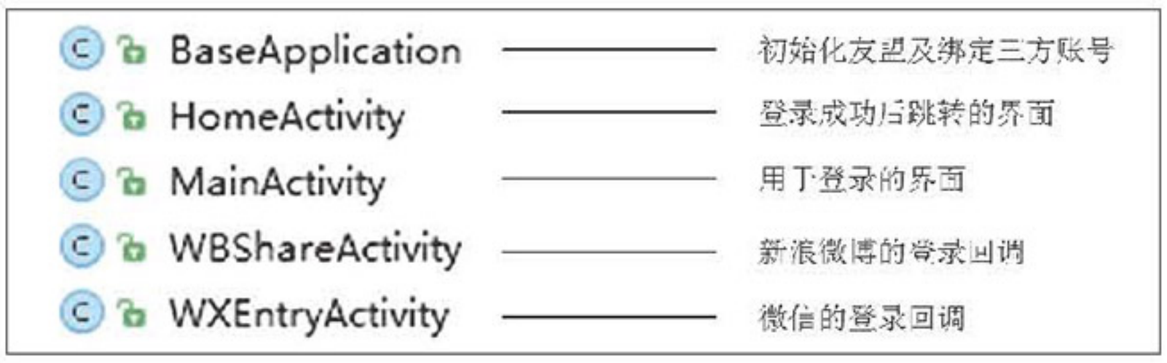


图 4.9　关键类

实现过程

（1）打开 Android Studio，新建一个项目，命名为“UmengEnter”。
（2）在 AndroidManifest.xml 文件中注册相关权限，并声明相关的类文件。
（3）将相关的 Jar 包复制到 libs 文件夹中。
（4）本实例中使用的关键类及其作用，如表 4.1 所示。

表 4.1　关键类

类　名	描　述
MainActivity	该类为登录界面
BaseApplication	该类为绑定三方登录平台的 ID
WBShareActivity	该类为新浪微博登录的回调类
WXEntryActivity	该类为微信登录的回调类

（5）在 BaseApplication 类中，初始化友盟，初始化代码如下：

```
UMShareAPI.get(this);                    //初始化友盟
```

（6）在 BaseApplication 的类中调用微信、QQ 或新浪微博登录时，还需要添加第三方平台申请的账号，添加的格式如下：

```
01  //微信
02  PlatformConfig.setWeixin("wxdc1e388c3822c80b", "3baf1193c85774b3fd9d18447d76cab0");
03  //新浪微博
04  PlatformConfig.setSinaWeibo("3921700954", "04b48b094faeb16683c32669824ebdad");
05  //QQ登录
06  PlatformConfig.setQQZone("100424468", "c7394704798a158208a74ab60104f0ba");
```

扩展学习

请尝试当返回成功后获取用户的用户名、密码、头像等相关信息。

实例 036　极光推送

应用场景：App 推送消息

源码位置：光盘\Code\04\036

实例说明

大多数 App 都有消息推送的功能，有些是后台写的长链接，有些直接应用第三方软件实现。本实例将应用极光推送实现消息推送。运行实例，在极光后台操作需要推送的通知，如图 4.10 所示；在手机界面显示接收到系统推送的通知，如图 4.11 所示。

图 4.10　后台操作推送通知

图 4.11　手机界面显示通知

关键技术

关键技术

本实例主要使用极光开发者服务，下面介绍如何使用该服务。

（1）打开极光推送的官方网站（http://www.jiguang.cn/push），注册账号。

（2）注册账号之后，系统默认打开的是个人中心，在个人中心单击创建项目，如图 4.12 所示，添加相关的信息。

图 4.12　创建应用

（3）应用创建完成后，系统会弹出应用详情的的界面，如图 4.13 所示，单击“下载 Demo”按钮，将 Demo 下载到指定的位置。

图 4.13　应用信息

（4）此时下载的 Demo 里面的 AppKey 为刚刚申请的，只需要将 Demo 中需要的类文件复制到项目中，极光的初始化位置在 BaseApplication 类中，代码如下：

```
JPushInterface.init(this);          //初始化JPush
```

（5）在 Demo 中，主要使用如图 4.14 所示的 5 个关键类文件便可以实现消息的推送和接收。

图 4.14　关键类

实现过程

（1）打开 Android Studio，新建一个项目，命名为“JiGuang”，公司域名为“mingrisoft.com”。

（2）在“app\libs”文件夹中导入相关的极光推送的 Jar 包。

（3）在“app\src\main”文件夹中创建 jniLibs 文件夹，向该文件夹中导入相关的 .so 文件。

（4）在 AndroidManifest.xml 文件中添加相关的权限及声明类文件。

（5）将 MyReceiver 复制到自己的项目中，此类用来接收通知。当单击通知时，跳转到指定界面的代码如下：

```
01  Log.d(TAG, "[MyReceiver] 用户点击打开了通知");
02  //打开自定义的Activity
03  Intent i = new Intent(context, TestActivity.class);
04  i.putExtras(bundle);
05  context.startActivity(i);
```

（6）在 BaseApplication 的类中要对极光进行初始化，初始化代码如下：

```
JPushInterface.init(this);          //初始化极光推送
```

实例 037　第三方分享

应用场景：App 借助第三方平台登录分享数据

源码位置：光盘\Code\04\037

实例说明

本实例介绍如何使用友盟的分享平台将数据分享到其他的第三方平台。运行实例，如图 4.15 所示的分享界面；选择并单击“微信朋友圈”平台，显示如图 4.16 所示的微信朋友圈分享界面；选择 QQ 空间平台，则显示如图 4.17 所示的 QQ 空间分享界面。

图 4.15　分享界面

图 4.16　微信朋友圈分享

图 4.17　QQ 空间分享

关键技术

本实例主要使用友盟社会化分享服务，下面介绍如何使用该服务。

（1）打开友盟的官方网站（http:/ http://www.umeng.com），注册账号。

（2）账号注册完成之后，需要去腾讯或新浪的平台申请第三方的账号。

（3）申请完第三方账号之后，回到友盟，进入社交类的 U-Share 分享，选择 Android 的集成文档，如图 4.18 所示。

图 4.18　应用信息

（4）单击下载 SDK，友盟的 SDK 会给用户提供 Demo，当用户只做分享的相关操作时，如图 4.19 所示的关键类对用户来说是有用的。

图 4.19　关键类

实现过程

（1）打开 Android Studio，新建一个项目，命名为“UMShare”。

（2）将 AndroidManifest.xml 文件中添加相关的权限及声明类文件。

（3）将相关的 Jar 包复制到 libs 文件夹中。

（4）在本实例中有如表 4.2 所示的关键类及其作用。

表 4.2 关键类

类　　名	描　　述
ShareMenuActivity	该类为分享数据时弹出分享到某平台的菜单类
BaseApplication	该类为绑定第三方登录平台的 ID
WBShareActivity	该类为新浪微博分享的回调类
WXEntryActivity	该类为微信分享的回调类
Defaultcontent	该类为分享的内容

（4）在 BaseApplication 类中，初始化友盟，代码如下：

```
UMShareAPI.get(this);  //初始化友盟
```

（5）在 BaseApplication 的类中若使用微信、腾讯、新浪微博分享时，需添加第三方平台申请的账号，添加的格式如下：

```
01  //微信分享
02  PlatformConfig.setWeixin("wxdc1e388c3822c80b", "3baf1193c85774b3fd9d18447d76cab0");
03  //新浪微博分享
04  PlatformConfig.setSinaWeibo("3921700954", "04b48b094faeb16683c32669824ebdad");
05  //QQ分享
06  PlatformConfig.setQQZone("100424468", "c7394704798a158208a74ab60104f0ba");
```

扩展学习

本实例是从底部显示分享菜单，请试着实现分享的菜单在中间显示出来。

实例 038　微信支付

应用场景：App 支付界面

源码位置：光盘\Code\04\038

实例说明

目前，微信抢红包的功能深受社会上各种人群的欢迎，微信抢红包应用到了支付功能，本实例将主要讲解微信支付的流程。

关键技术

本实例主要使用微信支付服务，下面介绍如何使用该服务。

（1）打开微信支付官方网站（https://pay.weixin.qq.com/index.php/core/home/login?return_url=%2F），注册账号。

（2）开放平台需进行开发者资质认证后才可申请微信支付，认证费用 300 元 / 次；提交 App 基本信息，通过开放平台应用审核，以获得 AppID。

（3）登录开放平台，单击“管理中心”，选择需要申请支付功能对应的 App，开始填写资料等待审核，审核时间为 1-5 个工作日内。

（4）资料审核通过后，请登录联系人邮箱查收商户号和密码，完成账户验证。

（5）本协议为线上电子协议，签署后方可进行交易及资金结算，签署完立即生效。

（6）进行数据传输时，请遵循以下规则：

传输方式：为保证交易安全性，采用 HTTPS 传输。

提交方式：采用 POST 方法提交。

数据格式：提交和返回数据都为 XML 格式，根节点名为 xml。

字符编码：统一采用 UTF-8 字符编码。

签名算法：MD5，后续会兼容 SHA1、SHA256、HMAC 等。

签名要求：请求和接收数据均需要校验签名。

证书要求：调用申请退款、撤销订单接口需要商户证书。

判断逻辑：先判断协议字段返回，再判断业务返回，最后判断交易状态。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“WeiXinPay”。

（2）在官网下载支付的 Demo，将 Demo 用开发工具打开。

（3）导入相关的 Jar 包到“app\libs”目录中

（4）导入 .So 文件到“app\java\jniLibs”目录中。

（5）在 Constants 类中将 API_ID 换成自己申请的 APP_ID，关键代码如下：

```
public static final String APP_ID = "wxd930ea5d5a258f4f";  //更换成自己的ID
```

（6）AppRegister 广播类用于验证记录的 APPID，该类继承 BroadcastReceiver，实现 onReceiver() 方法，关键代码如下：

```
01  final IWXAPI msgApi = WXAPIFactory.createWXAPI(context, null);
02  //支付平台记录AppId
03  msgApi.registerApp(Constants.APP_ID);
```

（7）支付的回调类为 PayActivity，支付后判断成功或者失败的代码均在此类中。

实例 039　支付宝支付

应用场景：App 支付界面

源码位置：光盘\Code\04\039

实例说明

App 中涉及到购物、交易等需要支付的操作，都可以使用支付宝进行支付，本实例主要讲解支付宝支付的流程。

关键技术

关键技术

本实例主要使用蚂蚁金融服务实现支付宝支付功能，下面介绍如何使用该服务。

（1）打开蚂蚁金服官方网站（https://openhome.alipay.com/platform/home.htm），注册账号。

（2）账号注册完成之后，进入开发者中心，在开发者中心单击创建项目，将显示创建名称应用页面，在该页面中输入应用名称，为“明日科技”，如图 4.20 所示。

图 4.20　创建应用

（3）单击“创建”按钮，可以看到如图 4.21 所示的界面，红框区域为以后要用的 APPID。

图 4.21　应用信息

（4）在进行支付宝开发时，除了需要 APPID 外还需要一些其他相关的信息，如图 4.22 所示。

图 4.22　设置相关信息

实现过程

（1）打开 Android Studio，新建一个项目，命名为“PayZhiFuBao”。

（2）在官网下载支付的 Demo，将 Demo 用开发工具 Android Stidio 打开。

（3）导入相关的 Jar 包到“app\libs”目录中

（4）导入 .So 文件到“app\java\jniLibs”目录中。

（5）其中最重要的类就是 PayDemoActivity，在该类中重点要找到以下代码，并根据注释进行设置。关键代码如下：

```
01  //TODO/** 支付宝支付业务：入参app_id */
02  public static final String APPID = "";
03  //TODO/** 支付宝账户登录授权业务：入参pid值 */
04  public static final String PID = "";
05  //TODO/** 支付宝账户登录授权业务：入参target_id值 */
06  public static final String TARGET_ID = "";
07  //TODO/** 商户私钥，pkcs8格式 */
08  //TODO/** 如下私钥，RSA2_PRIVATE 或者 RSA_PRIVATE 只需要填入一个 */
09  //TODO/** 如果商户两个都设置了，优先使用 RSA2_PRIVATE */
10  //TODO/** RSA2_PRIVATE 可以保证商户交易在更加安全的环境下进行，建议使用 RSA2_PRIVATE */
11  //TODO/** 获取 RSA2_PRIVATE，建议使用支付宝提供的公私钥生成工具生成， */
12  //TODO/** 工具地址：*/
13  //TODO/**https://doc.open.alipay.com/docs/doc.htm?treeId=291&articleId=106097&docType=1 */
```

```
14  public static final String RSA2_PRIVATE = " ";
15  public static final String RSA_PRIVATE = " ";
16  private static final int SDK_PAY_FLAG = 1;
17  private static final int SDK_AUTH_FLAG = 2;
```

实例 040　银联支付

应用场景：App 支付界面

源码位置：光盘\Code\04\040

实例说明

中国银联为满足广大银联持卡人、发卡机构、商户以及电子商务企业的互联网支付需要，提供了手机端的支付功能。本实例将向您介绍银联支付时的流程。

关键技术

关键技术

下面介绍如何应用银联支付服务。

（1）打开银联开放平台网站（https://open.unionpay.com/ajweb/help/file/techFile?productId=3）注册账号。

（2）了解银联支付的流程，具体的流程如图 4.23 所示：

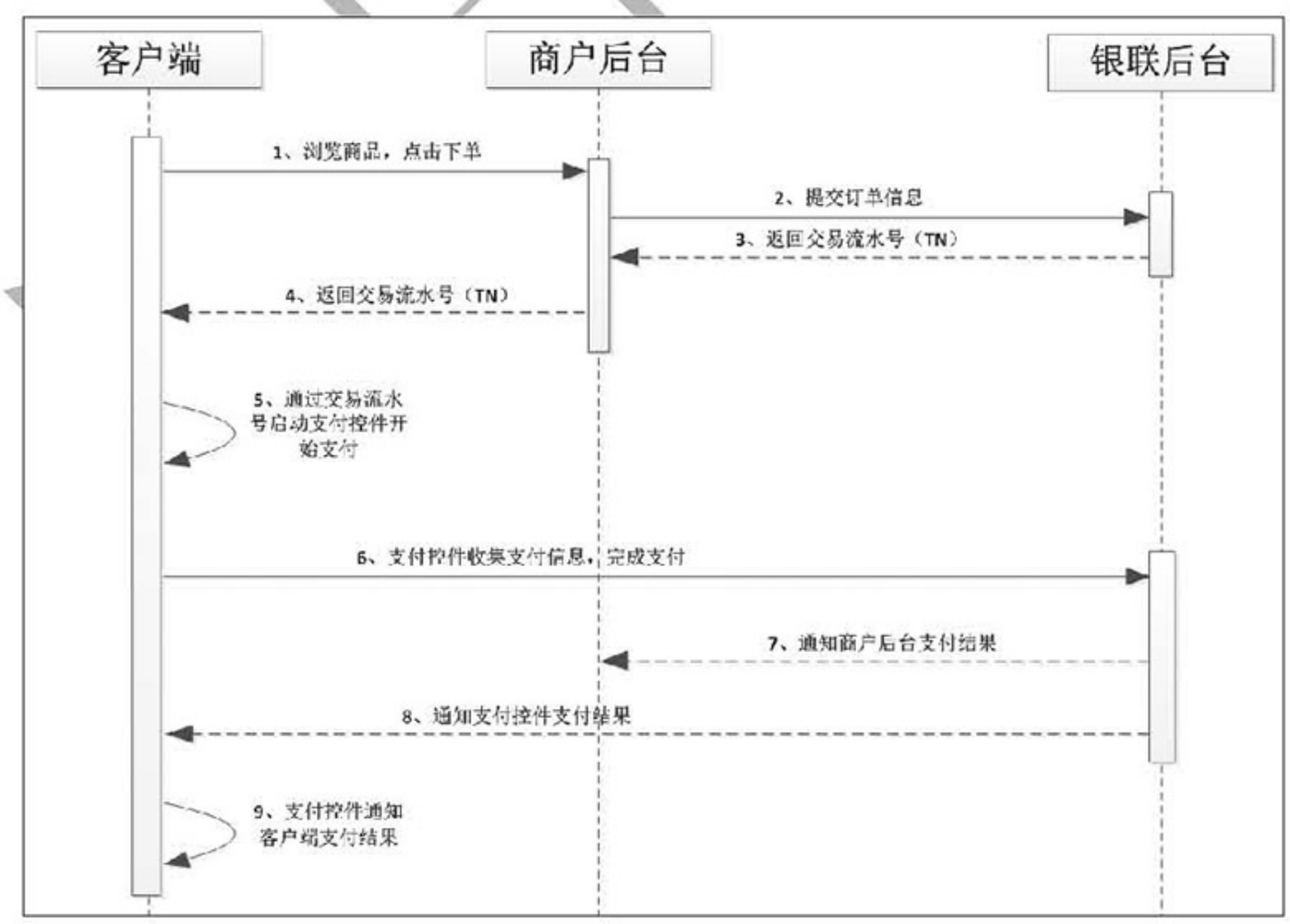


图 4.23　银联支付流程图

（3）单击“手机控件支付”按钮，如图 4.24 所示。

图 4.24　手机控件支付的位置

（4）在弹出的列表中单击下载安卓版的手机支付开发包，如图 4.25 所示。

图 4.25　下载按钮的位置

（5）下载成功后，进行解压，并使用开发工具 Android Stidio 打开下载好的项目。

实现过程

（1）从 Demo 中复制“upmp_android/sdkPro/jar/data.bin”到项目的 assets 文件夹中。

（2）复制“upmp_android/sdkPro/jar/xxx/libentryexpro.so”和“libuptsmaddon.so”到项目的 armeabi-v7a 文件夹中。

（3）复制“upmp_android/sdkPro/UPPayAssistEx.jar”到项目的 libs/ 目录下。

（4）复制“upmp_android/sdkPro/jar/UPPayPluginExPro.jar”到项目的 libs/ 目录下。

（5）在 AndroidManifest.xml 文件中添加与银联支付相关的权限代码，关键代码如下：

```
<!-- 银联支付相关配置 -->
<uses-library
    android:name="org.simalliance.openmobileapi"
    android:required="false" />
<activity
    android:name="com.unionpay.uppay.PayActivity"
    android:configChanges="orientation|keyboardHidden"
    android:excludeFromRecents="true"
    android:label="@string/app_name"
    android:screenOrientation="portrait"
    android:windowSoftInputMode="adjustResize" />
<activity
    android:name="com.unionpay.UPPayWapActivity"
    android:configChanges="orientation|keyboardHidden"
    android:screenOrientation="portrait"
    android:windowSoftInputMode="adjustResize" />
<!--银联支付的相关权限-->
<uses-permission android:name="android.permission.INTERNET" />
```

```
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
<uses-permission android:name="android.permission.CHANGE_NETWORK_STATE" />
<uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE" />
<uses-permission android:name="android.permission.READ_PHONE_STATE" />
<uses-permission android:name="android.permission.ACCESS_WIFI_STATE" />
<uses-permission android:name="android.permission.NFC" />
<uses-feature android:name="android.hardware.nfc.hce" />
<uses-permission android:name="android.permission.RECORD_AUDIO" />
<uses-permission android:name="android.permission.MODIFY_AUDIO_SETTINGS" />
<uses-permission android:name="org.simalliance.openmobileapi.SMARTCARD" />
```

（5）当支付完成后，执行回调方法，返回值为“支付成功”“支付失败”和“支付取消”，App 根据返回的数据执行相关的操作，关键代码如下：

```
@Override
protected void onActivityResult(int requestCode, int resultCode, Intent data) {
    if (data == null) {
        return;
    }
    String msg = "";
    /*
     * 支付控件返回字符串:success、fail、cancel 分别代表支付成功、支付失败、支付取消
     */
    String str = data.getExtras().getString("pay_result");
    Log.v("zftphone", "2 "+data.getExtras().getString("merchantOrderId"));
    if (str.equalsIgnoreCase("success")) {
        msg = "支付成功! ";
    } else if (str.equalsIgnoreCase("fail")) {
        msg = "支付失败! ";
    } else if (str.equalsIgnoreCase("cancel")) {
        msg = "用户取消了支付";
    }
    //支付完成，处理自己的业务逻辑
}
```

明日科技

第5章 手势交互

实例 041 仿 QQ 删除消息提示

应用场景：删除 QQ 列表的消息

源码位置：光盘\Code\05\041

实例说明

在 QQ 聊天软件中，在消息列表界面中可通过拖曳实现删除未读消息提示的功能，此功能的设计吸引了开发者的眼球。本实例将实现删除 QQ 列表消息的功能。运行实例，有未读消息提示的效果如图 5.1 所示，拖曳的效果如图 5.2 所示，拖曳后抬起手指的效果如图 5.3 所示。

图 5.1 显示未读消息提示

图 5.2 拖曳中的效果

图 5.3 拖曳后删除的效果

关键技术

本实例主要使用 View 类中的 onTouch() 接口方法，Path 类的 quadTo()、moveTo() 方法，进行手势的判断与绘制贝塞尔曲线。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“QQDeleteMsg”，公司域名为“mingrisoft.com”。

（2）导入资源文件：复制“光盘\Code\05\041\src”图片资源粘贴到“res\drawable”目录中，并在该目录中创建消息红色圆形背景。

（3）创建名称为 activity_main 的布局文件。关键代码如下：

```
<!--头布局的图片-->
<ImageView
    android:layout_width="match_parent"
    android:layout_height="50dp"
    android:background="@mipmap/top"
    android:layout_alignParentTop="true"
    android:layout_alignParentLeft="true"
    android:layout_alignParentStart="true"
    android:id="@+id/imageView" />
<!--脚布局的图片-->
<ImageView
    android:layout_width="match_parent"
    android:layout_height="50dp"
    android:background="@mipmap/buttom"
    android:layout_alignParentBottom="true"
    android:layout_alignParentLeft="true"
    android:layout_alignParentStart="true"
    android:id="@+id/imageView2" />
<!--加载数据的list-->
<ListView
    android:id="@+id/list"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:layout_below="@+id/imageView"
    android:layout_marginTop="10dp"
    android:layout_marginBottom="60dp"
    android:layout_alignParentLeft="true"
    android:layout_alignParentStart="true">
</ListView>
```

（4）创建自定义的 DrawCircle 类，主要的拖曳时的动画都在此类中绘制，在该类的初始化方法中添加以下重要代码。关键代码如下：

```
Path path = new Path();                                    //初始化路径
path.moveTo(mPointA.x, mPointA.y);                         //到A点
```

明日科技

```
path.quadTo(mPointE.x, mPointE.y, mPointC.x, mPointC.y);          //从A走贝塞尔曲线到C点
path.lineTo(mPointD.x, mPointD.y);                                //从C走直线到D点
path.quadTo(mPointE.x, mPointE.y, mPointB.x, mPointB.y);          //从D点走贝塞尔曲线到B点
path.close();                                                     //关闭路径，形成回路
canvas.drawPath(path, mPaint);                                    //画出路径
canvas.drawCircle(mCicleCenterPoint.x, mCicleCenterPoint.y, mRadius, mPaint); //画固定圆
```

（5）创建 List 适配器实现主要的功能。

扩展学习

请尝试将 QQ 的聊天界面——群聊天的界面背景更改为蓝色。

实例 042 九点锁屏

应用场景：应用锁屏软件解锁时

源码位置：光盘\Code\05\042

实例说明

手机 App 中有一款锁屏软件——九点锁屏，即屏幕上有九个圆点，随意连接几个点，如果连接后形成的图案与预先设定好的一致，则解锁成功。本实例将实现一个九点锁屏软件。运行实例，绘制手势密码，如图 5.4 和图 5.5 所示。

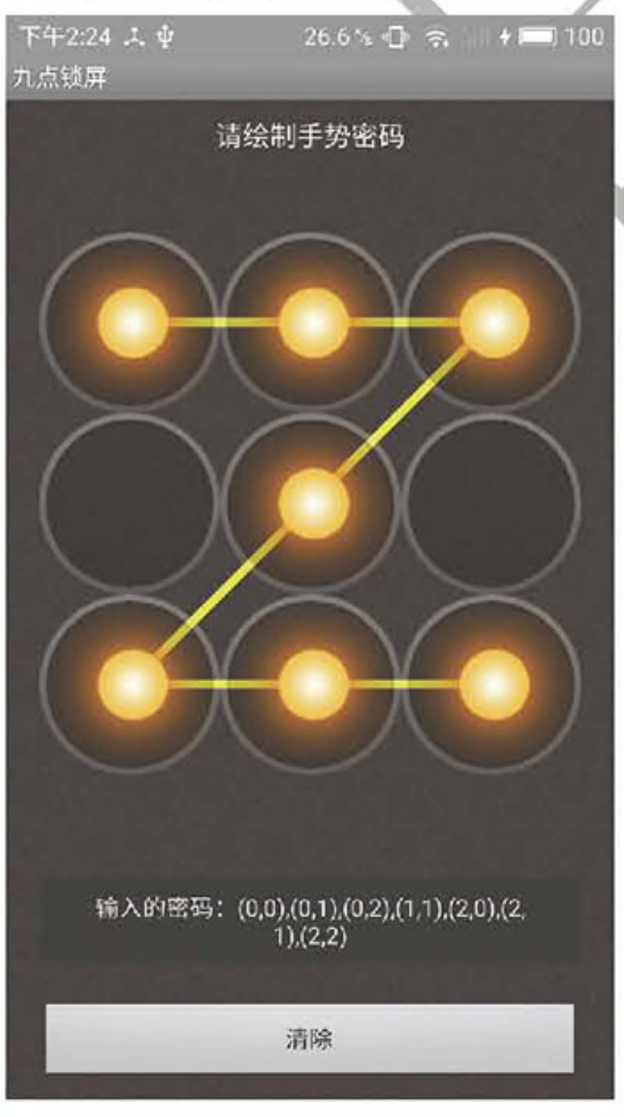


图 5.4　绘制手势密码（一）

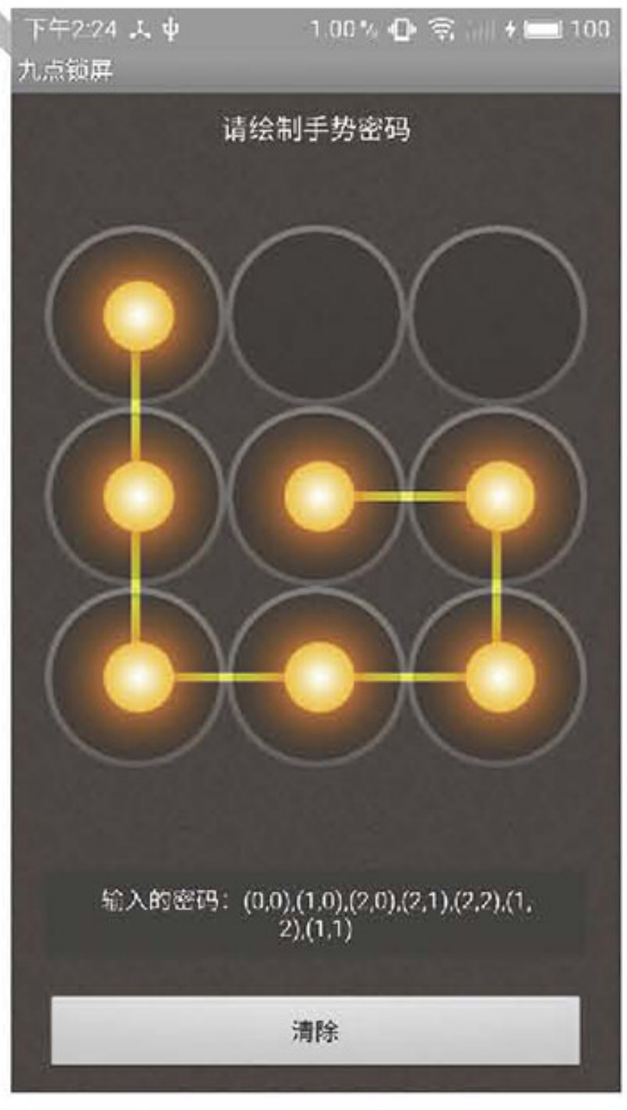


图 5.5　绘制手势密码（二）

关键技术

本实例主要使用 Math 类中的 min() 方法，对比两个数并且取最小的数，然后使用 Paint 类中的 setAlpha() 方法，设置画笔的透明度。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“ApplicationLock”，公司域名为“mingrisoft.com”。

（2）导入资源文件：复制“光盘\Code\05\042\src”中的图片资源，粘贴到“app\res”文件夹中的 drawable 文件下。

（3）在“app\res\values”文件夹中创建 attrs 文件，用于设置锁屏模式的属性。

（4）创建 LockPatternView 类用来加载九点锁屏，并执行相关的操作，在该类中首先定义一个 3×3 的矩阵，用来加载九点锁屏的九个点，关键代码如下：

```
/*
 * 表示解锁模式视图的3 x 3矩阵中的单元格
 */
public static class Cell {
    int row;
    int column;
    //保持#对象限制为9
    static Cell[][] sCells = new Cell[3][3];                    //初始化矩阵
    static {
        for (int i = 0; i < 3; i++) {
            for (int j = 0; j < 3; j++) {
                sCells[i][j] = new Cell(i, j);                  //for循环添加行与列
            }
        }
    }
}
```

（5）进行手势的判断，实现手指落下、抬起和移动时执行不同的操作，关键代码如下：

```
switch (event.getAction()) {
case MotionEvent.ACTION_DOWN:
    handleActionDown(event);                                    //手指落下时
    return true;
case MotionEvent.ACTION_UP:
    handleActionUp(event);                                      //手指抬起时
    return true;
case MotionEvent.ACTION_MOVE:
    handleActionMove(event);                                    //手指移动时
    return true;
```

（6）在布局文件中引用之前创建好的锁屏视图类文件，关键代码如下：

```
<mingrisoft.com.applicationlock.LockPatternView
    android:id="@+id/gesturepwd_lockview"
```

```
03      android:layout_width="match_parent"
04      android:layout_height="0dp"
05      android:layout_gravity="center_horizontal"
06      android:layout_margin="20dp"
07      android:layout_weight="3" />
```

扩展学习

本实例在绘制锁屏时，弹出的提示为二维数组的位置，请尝试在绘制锁屏界面中弹出（1~9）的位置。

实例 043 看着我

应用场景：游戏

源码位置：光盘\Code\05\043

实例说明

本实例将实现一个小动画，动画的效果是通过手指在界面中滑动来实现，手指滑动到哪里，手机界面中的哆啦 A 梦眼睛就看哪里。运行实例，不进行操作时的效果如图 5.6 所示，当手指滑动时哆啦 A 梦眼睛不停地看着手指，效果如图 5.7 所示。

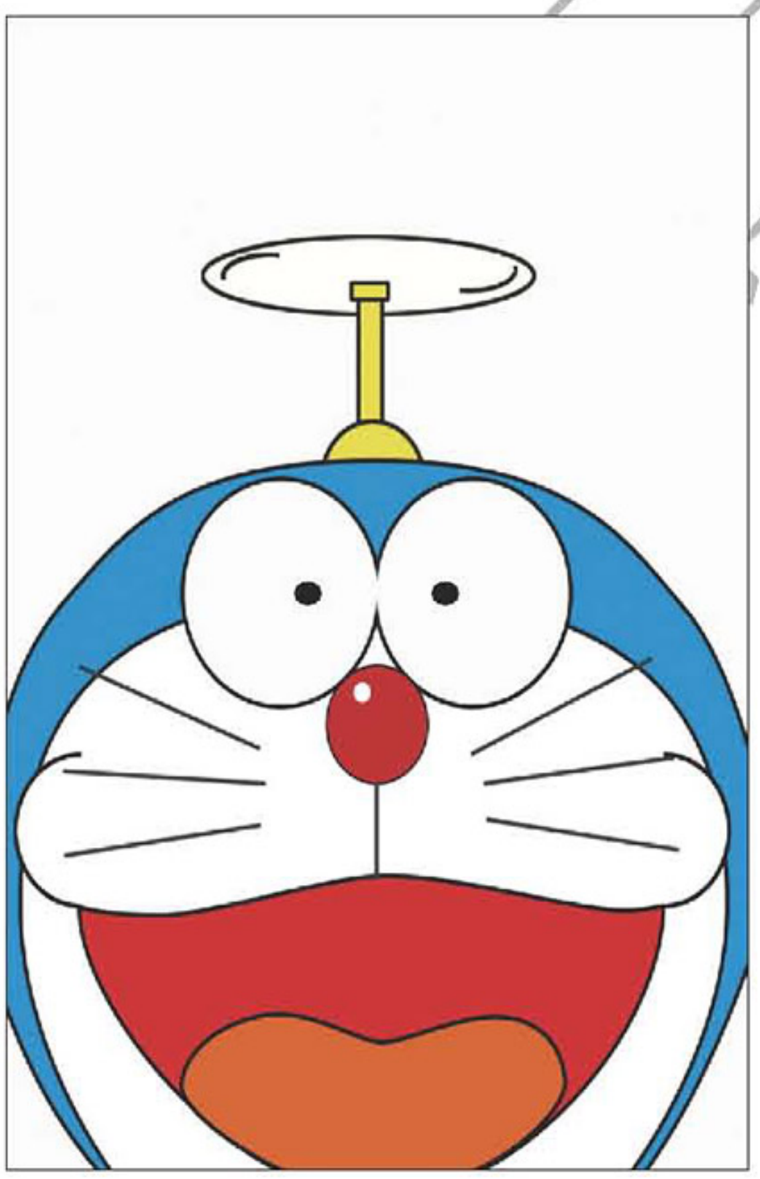


图 5.6　无操作时的效果

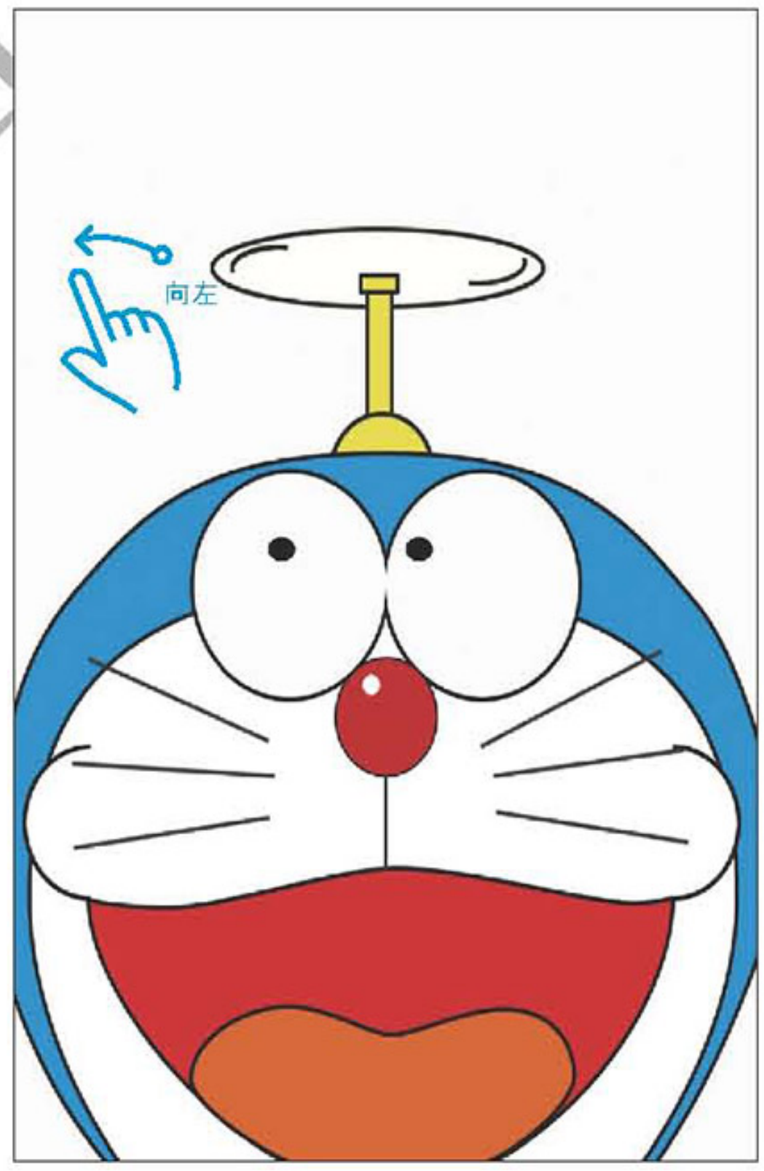


图 5.7　眼睛看着手指

关键技术

本实例主要使用 Activity 类中的 onTouchEvent () 方法（用于获取手指点在屏幕上的位置），Canvas 类中的 onDraw() 方法（用于刷新眼睛的位置），Math 类中的 abs() 方法（用于判断手指滑动的角度），实现手势监听和绘制左右眼的 View。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“LookMe”，公司域名为“mingrisoft.com”。

（2）修改显示界面的主题样式，去除系统默认添加的 ActionBar 标题栏，将“app\res\values\styles”中的“DarkActionBar”修改成“NoActionBar”。

（3）导入资源：复制“光盘\Code\05\043\src”中的图片资源，粘贴到“app\res”文件夹中的 mipmap 文件夹。

（4）为默认生成的 activity_main.xml 布局文件中的父类布局添加一个 id。

（5）创建 LeftEyeView 和 RightEyeView 类，让其继承自 View 类，用来绘制左右眼的 View 类。

（6）以 LeftEyeView 为例，介绍一下此类中的关键代码，在 onDraw() 的方法中绘制出哆啦 A 梦的眼睛，代码如下：

```
01  paint.setStyle(Paint.Style.STROKE);
02  //消除锯齿
03  paint.setAntiAlias(true);
04  //设置画笔的颜色
05  paint.setColor(Color.RED);
06  //消除锯齿
07  paint2.setAntiAlias(true);
08  //设置画笔的颜色
09  paint2.setColor(Color.BLACK);
10  //设置paint的外框宽度
11  paint.setStrokeWidth(2);
12  canvas.save();                          //锁定画布
13  canvas.rotate(s, x, y);                 //旋转45°
```

（7）创建 init() 初始化方法，用于初始化左右眼的视图，获取屏幕宽、高等。方法中的代码如下：

```
01  /**
02   * 初始化左右眼的View
03   **/
04  leftEyeView = new LeftEyeView(this);
05  rightEyeView = new RightEyeView(this);
06  /**
07   * 获取屏幕的宽、高
08   **/
09  WindowManager wm = (WindowManager) this.getSystemService(Context.WINDOW_SERVICE);
10  w = wm.getDefaultDisplay().getWidth();
11  h = wm.getDefaultDisplay().getHeight();
12  relativeLayout = (RelativeLayout) findViewById(R.id.activity_main);
```

明日科技

```
/**
 * 设置左眼的宽、高
 **/
leftEyeView.setMinimumHeight(200);
leftEyeView.setMinimumWidth(200);
/**
 * 设置右眼的宽、高
 **/
rightEyeView.setMinimumHeight(200);
rightEyeView.setMinimumWidth(200);
/**
 * 绘制出左眼和右眼，并添加到布局中
 **/
leftEyeView.invalidate();
rightEyeView.invalidate();
relativeLayout.addView(leftEyeView);
relativeLayout.addView(rightEyeView);
/**
 * 两眼最先出现的坐标
 **/
rightEyeView.setXY(w / 2 + 100, h / 2);
leftEyeView.setXY(w / 2 - 100, h / 2);
```

（8）在 onTouchEvent() 方法中，获取手指移动时的坐标，通过 Math 类的函数获取到手指滑动的弧度，将弧度传递给左右眼绘制的方法中，进行重绘。关键代码如下：

```
int x1 = w / 2, y1 = h / 2;                                    //点1坐标
int x2 = (int) event.getX(), y2 = (int) event.getY();          //点2坐标
int m = Math.abs(x1 - x2);
int n = Math.abs(y1 - y2);
double z = Math.sqrt(m * m + n * n);
int jiaodu = Math.round((float) (Math.asin(n / z) / Math.PI * 180)); //最终角度
if (x > (w / 2) && y < (h / 2)) {                              //手指在第一象限
    leftEyeView.setS((float) (90 - jiaodu));
    rightEyeView.setS((float) (90 - jiaodu));
} else if (x < (w / 2) && y < (h / 2)) {                       //手指在第二象限
    leftEyeView.setS((float) (270 + jiaodu));
    rightEyeView.setS((float) (270 + jiaodu));
} else if (x < (w / 2) && y > (h / 2)) {                       //手指在第三象限
    leftEyeView.setS((float) (270 - jiaodu));
    rightEyeView.setS((float) (270 - jiaodu));
} else if (x > (w / 2) && y > (h / 2)) {                       //手指在第四象限
    leftEyeView.setS((float) (90 + jiaodu));
    rightEyeView.setS((float) (90 + jiaodu));
}
```

明日科技

扩展学习

在 onDraw() 方法中绘制眼睛时，请尝试实现哆啦 A 梦的左眼为蓝色，右眼为黑色的效果。

实例 044	手势打电话 应用场景：绘制特定图案拨打指定电话	源码位置：光盘\Code\05\044

实例说明

手势打电话可以省去用户背手机号的烦恼，打开该软件后绘制一个手势，手机将读取预先与该手势绑定的手机号，并自动拨号。运行实例，绑定手机号与手势，如图 5.8 所示；绘制相应手势运行的效果，如图 5.9 所示；手机自动拨号的效果，如图 5.10 所示。

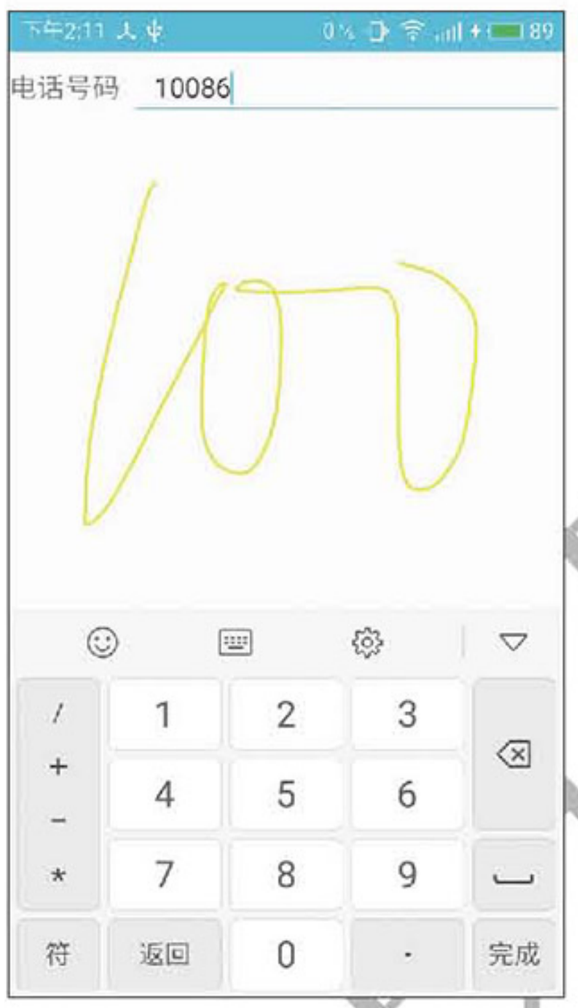
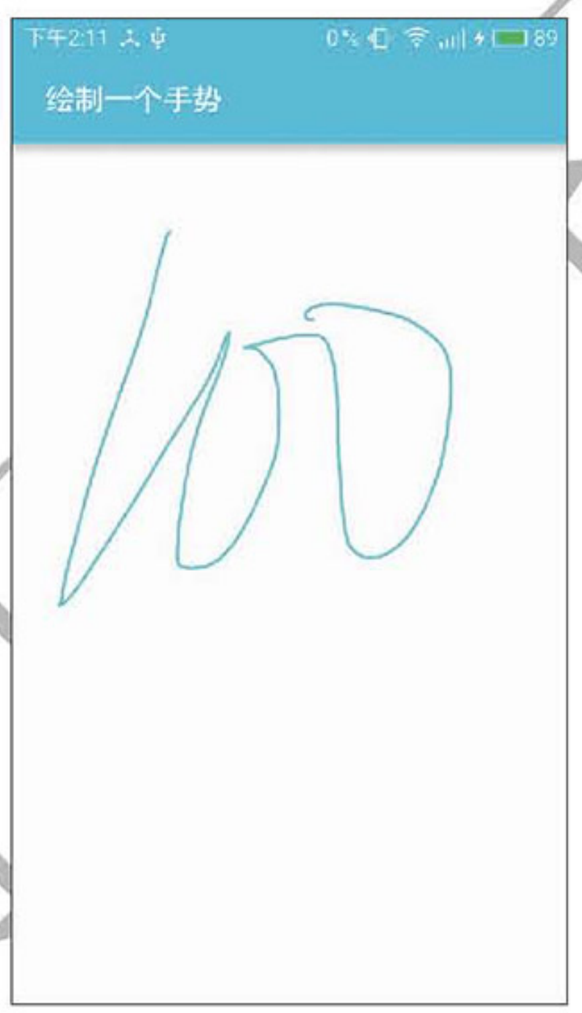


图 5.8　绑定手机号与手势图



5.9　绘制一个手势图

5.10　自动拨号

关键技术

本实例主要使用 GestureOverlayView 类中的 setGestureStrokeType() 方法，GestureLibrary 类中的 addGesture() 方法，实现创建手势画笔与保存手势。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“GestureRecognition”，公司域名为“mingrisoft.com”。

（2）创建 CreateGestureActivity 类，主要用于显示创建的手势以及为手势绑定电话号码。关键代码如下：

```
/*保存手势的相关操作*/
final GestureLibrary store = GestureBuilderActivity.getStore();  //初始化
store.addGesture(name.toString(), mGesture);                     //保存手势的方法
store.save();                                                    //进行保存
setResult(RESULT_OK);                                            //保存成功后的回调
```

（3）创建 GesturePerformedActivity 类，主要用于用户将绘制出的图形与之前保存的图形进行对比，吻合或者相似度高则获取其电话号码并拨打电话，关键代码如下：

```
/*弹出与之相对应的号码提示*/
Toast.makeText(GesturePerformedActivity.this, prediction.name, Toast.LENGTH_SHORT).show();
Intent intent = new Intent(Intent.ACTION_CALL);                  //调用系统拨打电话应用
Uri data = Uri.parse("tel:" + prediction.name);                  //获取电话号
intent.setData(data);                                            //将电话号码传给intent
startActivity(intent);                                           //启动intent
```

（4）在 GestureBuilderActivity 类实现展示之前所存入的手势以及跳转到手势添加和手势识别的界面，关键代码如下：

```
/*跳转到添加手势界面*/
public void addGesture(View v) {
    Intent intent = new Intent(this, CreateGestureActivity.class);
    startActivityForResult(intent, REQUEST_NEW_GESTURE);         //开始跳转
}
/*跳转到手势识别界面*/
public void gesturePerform(View v) {
    Intent intent = new Intent(this, GesturePerformedActivity.class);
    startActivity(intent);                                       //开始跳转
}
```

扩展学习

本实例应用手势进行拨打电话，为防止误拨打电话，请尝试实现进行呼叫前询问用户是否同意操作。

实例 045　手势翻书

应用场景：仿 QQ 的阅读效果

源码位置：光盘\Code\05\045

实例说明

通常手机中自带的电子书都带有翻书的效果，其效果非常炫酷。本实例将模仿 QQ 实现通过手势翻书的效果。运行实例，原始效果如图 5.11 所示，滑动中的效果如图 5.12 所示，继续滑动的效果如图 5.13 所示。

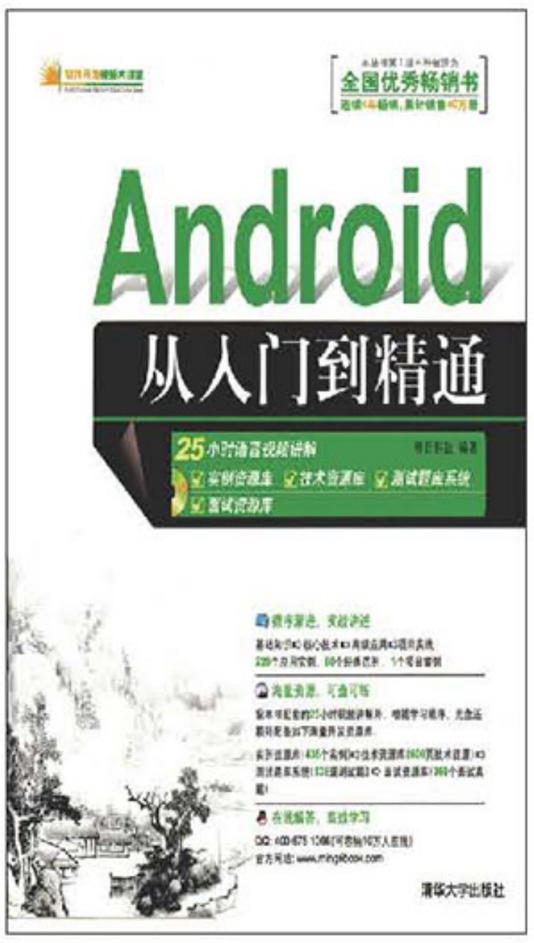


图 5.11　原始效果

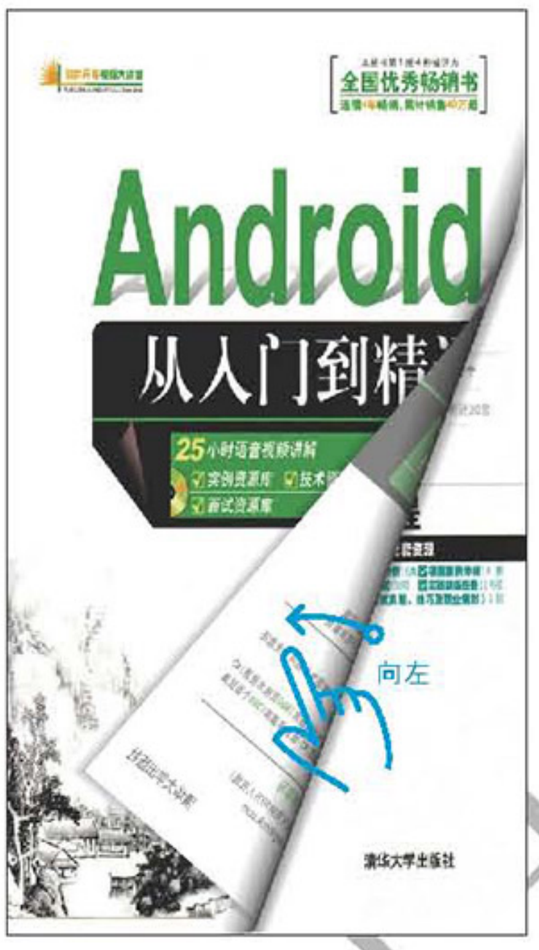


图 5.12　滑动中的效果

图 5.13　继续滑动的状态

关键技术

关键技术

本实例主要使用 GLSurfaceView 类中的 onDrawFrame() 和 setRenderer() 方法，实现绘制 3D 效果。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“OpenBook”，公司域名为“mingrisoft.com”。

（2）导入资源文件：复制“光盘\Code\05\045\src”文件中的图片资源，粘贴到“app\res”文件夹中的 mipmap 文件夹。

（3）修改显示界面的主题样式，并去除系统默认的 ActionBar 标题栏，然后打开“app\res\values\styles”文件，将“DarkActionBar”修改为“NoActionBar”。

（4）创建一个 MyView 视图类，让它继承 GLSurfaceView。关键代码如下：

```
public class MyView extends GLSurfaceView
```

（5）创建 CurlRenderer 类，该类继承自 GLSurfaceView.Renderer，用于通过 Renderer 对象调用绘图方法绘制 GLSurfaceView 的当前帧。该类中的接口代码如下：

```
01  /**
02   * 等待渲染引擎/状态更新
03   */
04  public interface Observer {
05      /**
06       *用于绘制界面时调用
07       */
08      void onDrawFrame();
```

```
09      /**
10       * 当界面改变时调用
11       */
12      void onPageSizeChanged(int width, int height);
13      /**
14       *初始化surface调用
15       */
16      void onSurfaceCreated();
17  }
```

（6）实现 MyView 视图类的所有方法后，让它在 Activity 的布局文件中显示出来。关键代码如下：

```
01  <LinearLayout xmlns:android=http://schemas.android.com/apk/res/android
02      android:layout_width="match_parent"
03      android:layout_height="match_parent">
04      <!--显示MyView-->
05      <mingrisoft.com.openbook.MyView
06          android:id="@+id/my_view"
07          android:layout_width="match_parent"
08          android:layout_height="match_parent" />
09  </LinearLayout>
```

（7）在创建好的 Activity 中添加图片资源，代码如下：

```
01  //图片资源
02  private int[] mBitmapIds = { R.mipmap.book1, R.mipmap.book2,
03          R.mipmap.book3, R.mipmap.book4 };
```

（8）在 Activity 中设置每页的翻书效果，主要是通过正面与背面设置不同的图像实现的，关键代码如下：

```
01  //正面是一张图片，背景是该图片的镜像
02  case 0: {
03      Bitmap front = loadBitmap(width, height, 0);
04      page.setTexture(front, CurlPage.SIDE_BOTH);
05      page.setColor(Color.argb(127, 255, 255, 255),
06              CurlPage.SIDE_BACK);
07      break;
08  }
```

扩展学习

三维坐标系统与二维坐标系统并不相同，二维坐标系统上的坐标值通常直接使用系统的像素数量；三维坐标系统的坐标值则取决于 glFrustumf() 方法的设置。

实例 046　图片的放大

应用场景：需要显示图片的放大效果

源码位置：光盘\Code\05\046

实例说明

在微信的朋友圈中，如果想更清晰地观察图片，通常会把图片放大一些。本实例将实现图片的放大效果。运行实例，图片的实际效果如图 5.14 所示，放大后的效果如图 5.15 所示，继续放大的效果如图 5.16 所示。

图 5.14　实际大小效果

图 5.15　放大后的图片

图 5.16　继续放大的图片

关键技术

本实例主要使用 BitmapFactory 类中的 decodeResource() 方法、View 类中的 onTouch() 方法，实现设置图片资源及手势的判断。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“ZoomImage”，公司域名为“mingrisoft.com”。

（2）导入资源文件：复制“光盘\Code\05\046\src”文件中的图片资源粘贴到“app\res”文件夹中的 mipmap 文件。

（3）修改显示界面的主题样式，去除系统默认添加的 ActionBar 标题栏，在“app\res\values\styles.

xml”文件中将“DarkActionBar”改为“NoActionBar”。

（4）创建自定义的名称为“ZoomImageView”的类文件，用于实现自定义的图片放大或缩小效果。

（5）ZoomImageView 类创建完成后，在该类的初始化方法中添加以下重要代码。

```
/*
 * 初始化
 **/
public ZoomImageView(Context context, AttributeSet attrs) {
    super(context, attrs);
    mZoomControl = new BasicZoomControl();                    //初始化基本的缩放控制
    mZoomListener = new BasicZoomListener();                  //初始化基本的缩放监听
    mZoomListener.setZoomControl(mZoomControl);               //设置图片的变焦控制
    setZoomState(mZoomControl.getZoomState());                //设置缩放状态
    setOnTouchListener(mZoomListener);                        //设置手势监听
    mZoomControl.setAspectQuotient(getAspectQuotient());
}
```

（6）在 activity_main 布局文件中，添加以下代码加载自定义的 ZoomImageView。

```
<!--自定义的ZoomImageView-->
 <mingrisoft.com.zoomimage.ZoomImageView
     android:id="@+id/image"
     android:layout_width="wrap_content"
     android:layout_height="wrap_content"
     />
```

（7）在 MainActivity 中，将添加的图片文件传给 ZoomImageView，关键代码如下：

```
bitmap = BitmapFactory.decodeResource(this.getResources(), R.mipmap.c); //添加图片文件
zoomImg.setImage(bitmap);                                               //把图片传给ZoomImageView
```

扩展学习

本实例非全屏显示，请尝试去除顶部状态栏，使界面全屏显示。

实例 047　长按碎屏效果

应用场景：模拟屏幕破碎的效果

源码位置：光盘\Code\05\047

实例说明

本实例将实现当手指长按手机屏幕的时候，屏幕会出现以按住屏幕点为中心向四周扩散的碎屏效果。运行实例，将显示如图 5.17 所示的未操作的界面；长按后将显示碎屏界面，实现碎屏效果，如图 5.18 所示；单击右上方的刷新按钮，屏幕中心会随机出现一张新图。

明日科技

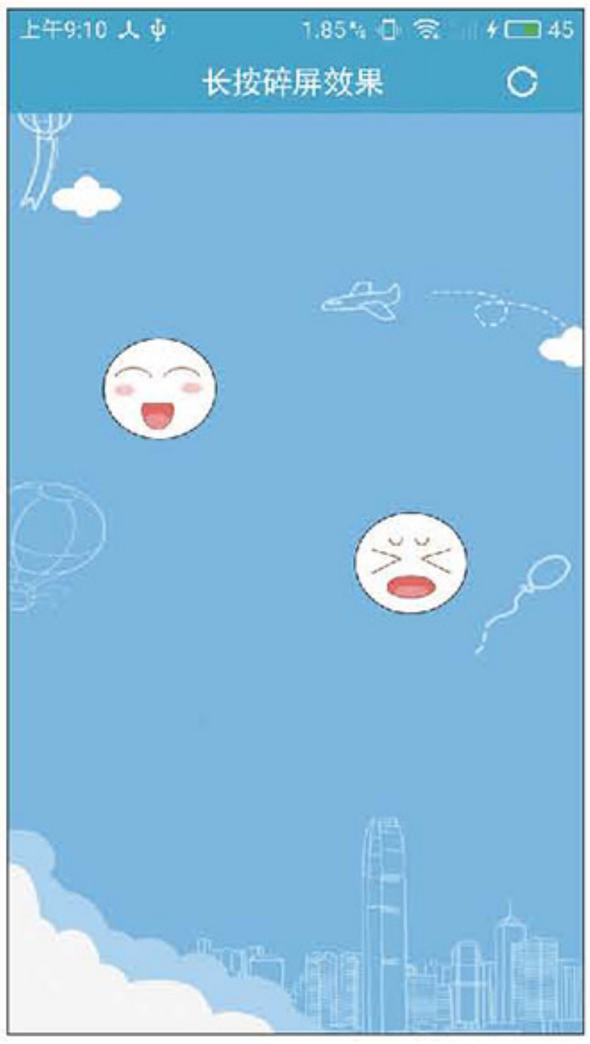


图 5.17　未操作的界面

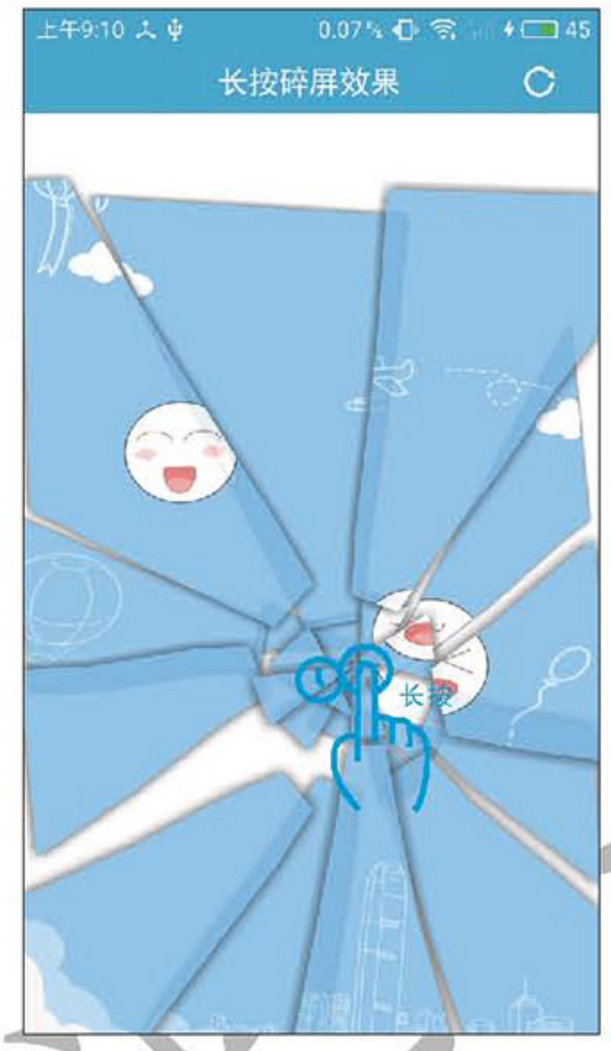


图 5.18　长按碎屏的界面

关键技术

关键技术

本实例主要使用 Paint 类中的 setColor() 方法，View 类中的 setVisibility()setOnTouchListener() 方法，实现设置画笔的颜色与设置图片显示 / 消失的效果。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“BrokenScreen”，公司域名为“mingrisoft.com”。

（2）导入资源文件：复制“光盘\Code\05\047\src”文件中的图片资源粘贴到“app\res”文件夹中的 mipmap 文件。

（3）在“app\res\values”文件夹中创建 array 文件，用于随机取出图片。

（4）在“app\java”文件夹中创建 tool 包，该包中主要为创建碎屏效果的工具类，具体的关键类及其作用如表 5.1 所示。

表 5.1　关键类

类　　名	描　　述
BrokenAnimator	碎屏动画类
BrokenCallback	碎屏回调类
BrokenTouchListener	碎屏手势监听类

（5）在 activity_main.xml 布局文件中，将系统默认的标题栏换成自定义的标题栏，代码如下：

```
<!--标题栏父类布局-->
<RelativeLayout
    android:layout_width="match_parent"
    android:layout_height="40dp"
    android:background="@android:color/holo_blue_light">
    <!--标题栏名称-->
    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_centerHorizontal="true"
        android:layout_centerVertical="true"
        android:text="长按屏幕碎屏"
        android:textColor="@android:color/white"
        android:textSize="19sp" />
    <!--标题栏刷新-->
    <ImageButton
        android:id="@+id/action_refresh"
        android:layout_width="20dp"
        android:layout_height="20dp"
        android:layout_alignParentEnd="true"
        android:layout_alignParentRight="true"
        android:layout_centerVertical="true"
        android:layout_marginEnd="28dp"
        android:layout_marginRight="28dp"
        android:background="@mipmap/shuaxin" />
</RelativeLayout>
```

（6）在 MainActivity 类中，初始化碎屏视图和画笔等，关键代码如下：

```
mBrokenView = BrokenView.add2Window(this);                    //绑定碎屏视图
whitePaint = new Paint();                                     //初始化画笔
whitePaint.setColor(0xffffffff);                              //设置画笔的颜色
//初始化碎屏监听
colorfulListener = new BrokenTouchListener.Builder(mBrokenView).build();
//将画笔设置到监听上
whiteListener = new BrokenTouchListener.Builder(mBrokenView).setPaint(whitePaint).build();
```

（7）绘制当屏幕破碎的时候，向下方掉落的“屏幕碎片”，关键代码如下：

```
//绘制阴影
paint.setStyle(Paint.Style.FILL);
paint.setShadowLayer(shadow,0,0,0xff333333);
canvas.drawPath(offsetPath,paint);
paint.setShadowLayer(0,0,0,0);

```

```
07  //如果视图有阿尔法通道
08  paint.setXfermode(new PorterDuffXfermode(PorterDuff.Mode.XOR));
09  canvas.drawPath(offsetPath,paint);
10  paint.setXfermode(null);
11
12  //绘制位图
13  paint.setShader(mBitmapShader);
14  paint.setAlpha(0xcc);
15  canvas.drawPath(offsetPath, paint);
```

（8）当单击标题栏中的刷新按钮的时候，执行刷新，并重置监听方法，关键代码如下：

```
01  //单击刷新按钮时，将界面重新加载
02  imageButton.setOnClickListener(new View.OnClickListener() {
03      @Override
04      public void onClick(View v) {
05          switch (v.getId()) {
06              case R.id.action_refresh:
07                  mBrokenView.reset();
08                  refreshDate();                    //刷新数据
09                  setOnTouchListener();             //设置监听的方法
10                  setViewVisible();                 //将视图显示出来
11                  break;
12          }
13      }
14  });
```

扩展学习

本实例的碎屏操作都针对布局以及图片，请尝试实现文字与按钮的碎屏效果。

实例 048　抓不到我

应用场景：屏幕矫正

源码位置：光盘\Code\05\048

视频位置：光盘\Video\05\048

实例说明

视频讲解

本实例将模拟一个“抓不到我”游戏。运行实例，如图 5.19 所示，当单击屏幕上的红色圆形时，红色圆形随机出现在另一个位置上，如图 5.20 所示；再次单击又会转换到另一个位置上，如图 5.21 所示。

明日科技

图 5.19　单击圆形

图 5.20　单击位置变动的圆形

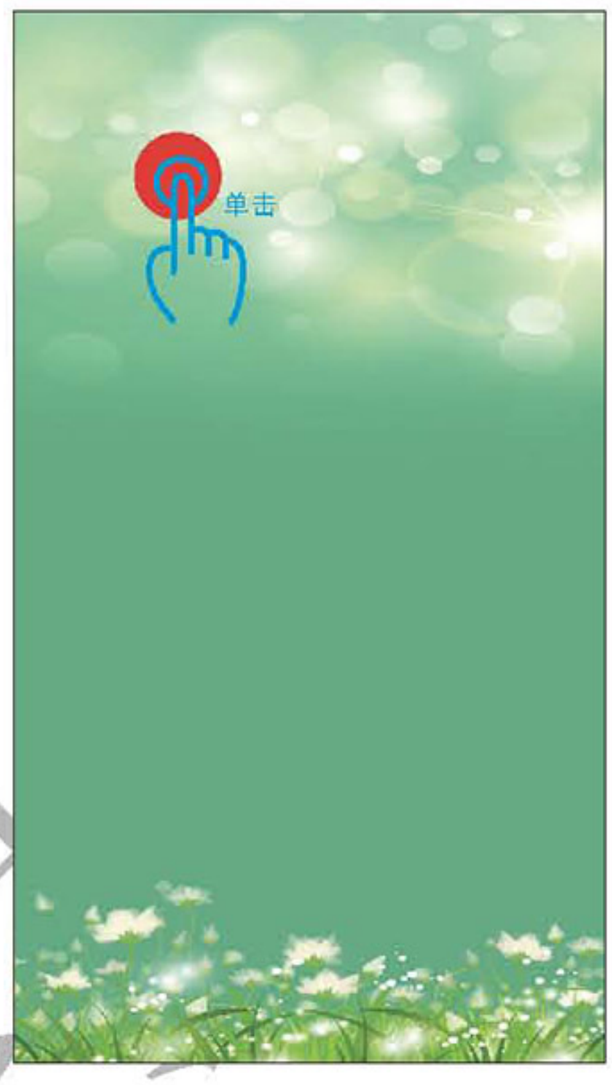


图 5.21　单击再次变动的圆形

关键技术

本实例主要使用 Canvas 类中的 drawCircle() 方法、View 类中的 invalidate() 方法，实现绘制圆形及刷新视图。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“CatchMe”，公司域名为“mingrisoft.com”。

（2）导入资源文件：复制“光盘\Code\05\048\src”文件中的图片资源粘贴到“app\res”文件夹中的 mipmap 文件夹中。

（3）创建 DrawCircle 类，让其继承 View 类，在重写的 onDraw() 方法中绘图，关键代码如下：

```
01  paint = new Paint();                          //初始化画笔
02  paint.setColor(Color.RED);                    //设置红色
03  paint.setAntiAlias(true);                     //设置画笔的锯齿效果。 true是去除，大家一看效果就明白了
04  canvas.drawCircle(x, y, 80, paint);           //大圆
```

（4）在 DrawCircle 类中添加一个 setXY() 方法，此方法是用来接收 MainActivity 传过来的坐标点，并且重新绘制圆。关键代码如下：

```
01  /**
02   * 此方法用于接收MainActivity传递过来的坐标点
```

```
03   **/
04  public void setXY(float x,float y){
05      this.x = x;
06      this.y = y;
07      invalidate();
08  }
```

（5）在 MainActivity 的手势判断的方法中判断手指单击的点是否已单击到圆内，如果单击到圆内，便在其他的位置重新绘制一个圆，关键代码如下：

```
01  //判断单击的位置是不是圆内
02  if ((drawCircle.getOldValues()[0] + 80 > x && x > drawCircle.getOldValues()[0] - 80)
03      && (drawCircle.getOldValues()[1] + 80 > y && y > drawCircle.getOldValues()[1] - 80)) {
04      //判断随机生成的坐标在不在当前圆内
05      Random random = new Random();
06      int sendX, sendY;
07      do {
08          //随机产生圆心坐标
09          sendX = random.nextInt(w - 80);
10          sendY = random.nextInt(h - 80);
11      } while ((sendX < 80 && sendY < 200)&& (drawCircle.getOldValues()[1] + 80 > sendY
12              && sendY > drawCircle.getOldValues()[1] - 80));
13      drawCircle.setXY(sendX, sendY);                    //根据X与Y的值画圆
14  }
```

扩展学习

本实例中的圆形为红色，请尝试将圆形换成蓝色并逐渐变为更大的圆形。

实例 049　转动的地球

应用场景：自定义手势

源码位置：光盘\Code\05\049

实例说明

本实例将实现当手指在屏幕上绘制放大或缩小的手势时，屏幕中心的地球会随之进行相应的变化，当手指在屏幕上进行旋转时，地球也会随之转动。运行实例，默认打开的界面如图 5.22 所示，正在进行放大旋转的操作的界面如图 5.23 所示，继续进行放大旋转操作的界面如图 5.24 所示界面。

明日科技

图 5.22　默认界面

图 5.23　放大旋转的界面

图 5.24　继续放大旋转

关键技术

本实例主要使用 Resources 类中的 getDrawable() 方法，Matrix 类中的 postScale() 方法、postRotate() 方法和 postTranslate() 方法，实现获取图片资源及对图片进行放大或旋转的变换。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“MyApplication”，公司域名为“mingrisoft.com”。

（2）导入资源文件：复制“光盘\Code\05\049\src”中的图片资源，粘贴到“app\res”文件夹中的 drawable 文件下。

（3）在 MainActivity 类中获取屏幕的宽与高，代码如下：

```
01  //获取屏幕的中心点
02  Display display = getWindowManager().getDefaultDisplay();
03  mFocusX = display.getWidth() / 2f;                    //屏幕宽度的一半
04  mFocusY = display.getHeight() / 2f;                   //屏幕高度的一半
```

（4）通过Matrix类中的平移方法，设置地球的位置，本实例将地球设置在屏幕中心，关键代码如下：

```
01  /**
02   * 用Matrix类中的平移方法设置地球的位置
03   **/
04  mMatrix.postTranslate(mFocusX - scaledImageCenterX, mFocusY - scaledImageCenterY);
05  earthView.setImageMatrix(mMatrix);                    //将地球设置到屏幕中心
```

（5）在 MainActivity 类中编写两个内部类，分别是图片缩放的内部类和图片旋转的内部类，代码如下：

```
/**
 * 图片放大缩小的监听
 **/
private class ScaleListener extends ScaleGestureDetector.SimpleOnScaleGestureListener {
    @Override
    public boolean onScale(ScaleGestureDetector detector) {
        mScaleFactor *= detector.getScaleFactor();
        //不要让物体太小或太大
        mScaleFactor = Math.max(0.1f, Math.min(mScaleFactor, 10.0f));
        return true;
    }
}
/**
 * 图片转动的监听
 **/
private class RotateListener extends RotateGestureDetector.SimpleOnRotateGestureListener {
    @Override
    public boolean onRotate(RotateGestureDetector detector) {
        mRotationDegrees -= detector.getRotationDegreesDelta();
        return true;
    }
}
```

（6）给图片旋转与缩放的手势设置监听事件，关键代码如下：

```
mScaleDetector.onTouchEvent(event);                    //为图片的缩放手势设置监听
mRotateDetector.onTouchEvent(event);                   //为图片的旋转手势设置监听
```

扩展学习

本实例中，地球可以随意移动，请尝试将地球固定在某一指定位置。

明日科技

第6章 图像处理

实例 050 浏览朋友圈图片

应用场景：朋友圈

源码位置：光盘\Code\06\050

视频位置：光盘\Video\06\050

视频讲解

实例说明

在即时通讯软件的朋友圈中，可以查看好友发表的动态，有些动态还会配上图片，如果长按其中的一张图片，还会弹出一些提示。本实例将实现类似的功能。运行实例，将显示如图 6.1 所示的模拟界面，单击第 3 张图片，将显示如图 6.2 所示的放大效果，长按后将显示如图 6.3 所示的提示。

图 6.1　打开的模拟界面

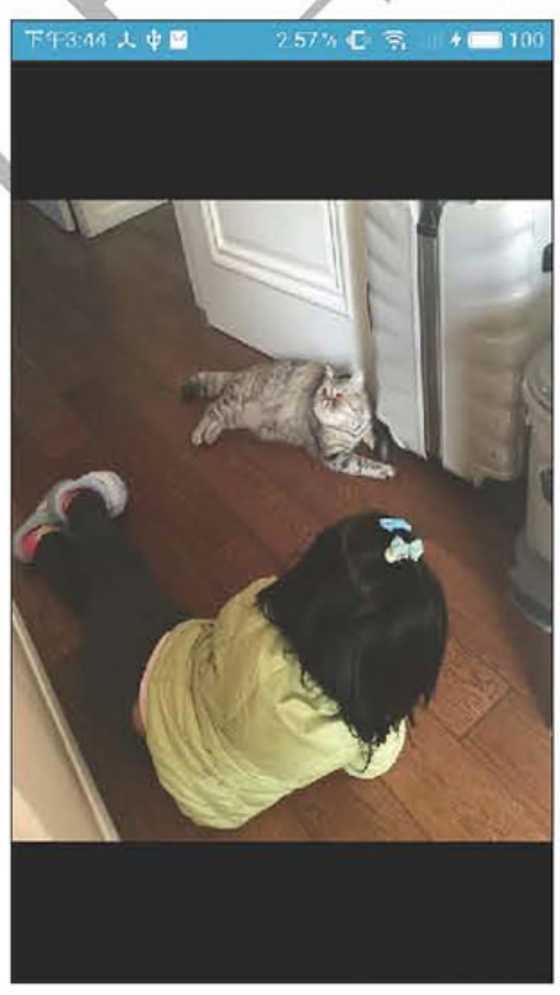


图 6.2　单击第三张图片

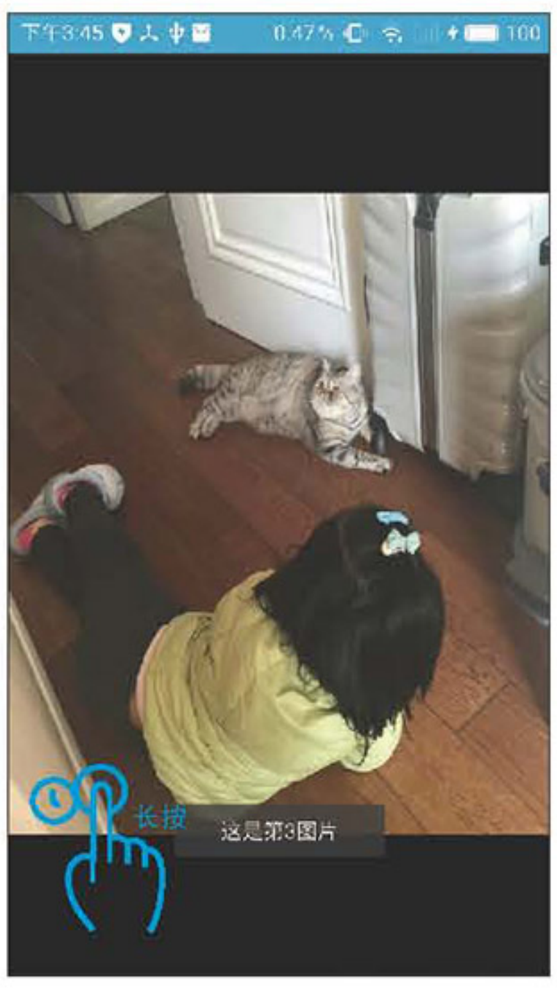


图 6.3　长按第三张图片

关键技术

本实例主要使用 Intent 类中的 putExtra() 方法、ViewPager 类中的 setCurrentItem() 方法，实现传递图片数据，并且放大显示图片。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“FriendCircle”，公司域名为“mingrisoft.com”。

（2）导入资源文件：复制“光盘\Code\06\050\src”文件中的图片资源，粘贴到“app\res”文件夹中的 mipmap 文件夹。

（3）在“app\java”文件夹中创建包名为“adpter”的包，此包用于设置 ListView 与 ViewPager 的适配器。

（4）在 ListView 嵌套 GridView 时，GridView 会有显示不全的问题，为了解决该问题，创建一个 MyGridView 类，继承 GridView，在该类中进行处理，关键代码如下：

```
/**
 * ListView嵌套GridView，解决GridView显示不全问题
 **/
@Override
protected void onMeasure(int widthMeasureSpec, int heightMeasureSpec) {
    //TODO Auto-generated method stub
    int expandSpec = MeasureSpec.makeMeasureSpec(Integer.MAX_VALUE >> 2, MeasureSpec.AT_MOST);
    super.onMeasure(widthMeasureSpec, expandSpec);
}
```

（5）在布局中调用创建的 MyGridView 类，用于解决嵌套显示不全的问题。

（6）在 ListView 的 Adapter 中给 GridView 绑定适配器，GridView 的适配器可以使用系统自带的 SimpleAdapter，并给 GridView 的 Item 添加单击监听事件，关键代码如下：

```
/**
 * 使用simpleAdapter封装数据，将图片显示出来
 * 参数一是当前上下文Context对象
 * 参数二是图片数据列表，要显示数据都在其中
 * 参数三是布局的XML文件，注意，不是整体界面，而是要显示在GridView中的单个Item的布局XML文件
 * 参数四是动态数组中与map中图片对应的项，也就是map中存储进去的对应于图片value的key
 * 参数五是单个Item的布局XML中的图片ID
 **/
SimpleAdapter simpleAdapter = new SimpleAdapter(context, imagelist,
        R.layout.grid_item_layout, new String[]{"image"}, new int[]{R.id.grid_item_image});
holder.gridView.setAdapter(simpleAdapter);                          //给GridView设置适配器
holder.gridView.setOnItemClickListener(new GridViewItemOnClick());  //添加GridView的单击事件
```

（7）单击指定的 Item 时，启用 Intent 进行界面的跳转，并将单击图片的 position 传递过去，关键代码如下：

```
/**
 * 启动展示图片的Activity
 **/
Intent intent = new Intent(context, ShowImageActivity.class);
intent.putExtra("id", position);            //将当前单击的位置传递过去
context.startActivity(intent);              //启动Activity
```

（8）在查看图片时，还可长按图片弹出提示，关键代码如下：

```
/**
 * 图片的长按监听
 **/
iv.setOnLongClickListener(new View.OnLongClickListener() {
    @Override
    public boolean onLongClick(View v) {
        //弹出提示，提示内容为当前的图片位置
        Toast.makeText(ShowImageActivity.this, "这是第" + (index + 1) + "图片",
                                                    Toast.LENGTH_SHORT).show();
        return false;
    }
});
```

扩展学习

本实例在查看图片时，长按图片弹出的是 Toast，请尝试做成微信那种弹出提示框。

实例 051　仿大街网选择图片

应用场景：浏览图片

源码位置：光盘\Code\06\051

实例说明

为了提供良好的用户体验，App 应用经常在图片进行翻页时，添加翻页动画。本实例将模仿大街网的选择图片功能。运行实例，将显示如图 6.4 的默认界面；单击右侧图片，使图片从右侧飞出，如图 6.5 所示；单击左侧按钮，使图片从左侧飞出，如图 6.6 所示。

关键技术

本实例主要使用 AnimationUtils 类中的 loadAnimation() 方法、ViewGroup 类中的 addView() 方法和 View 类中的 setRotation() 方法，实现设置图片的旋转及移除动画。

图 6.4　打开的默认界面

图 6.5　图片从右侧飞出

图 6.6　图片从左侧飞出

实现过程

（1）打开 Android Studio，新建一个项目，命名为“ImageChoose”，公司域名为“mingrisoft.com”。

（2）导入资源文件：复制“光盘\Code\06\051\src”文件夹中的图片资源，粘贴到“app\res”文件夹中的 drawable 文件夹。

（3）在“app\res\values”文件夹中创建 attrs 文件，用于将 drawable 文件中的图片添加到数组中。

（4）创建 MyImageLoading 类，初始化后在布局文件中调用该类文件，关键代码如下：

```
<mingrisoft.com.imagechoose.MyImageLoading
    android:id="@+id/image_loading"
    android:layout_width="270dp"
    android:layout_height="350dp"
    android:layout_centerHorizontal="true" />
```

（5）在 MyImageLoading 类中将图片资源获取到并添加到 ViewGroup 中，然后给图片资源设置相应的布局参数，关键代码如下：

```
/*
 * 设置需要添加视图的布局参数
 **/
FrameLayout.LayoutParams lytp = new FrameLayout.
                                LayoutParams(dip2px(context, 170),
                                        dip2px(context, 270), Gravity.CENTER);
/**
 * 将图片添加到视图上
```

```
 **/
addView(image1, lytp);
addView(image2, lytp);
addView(image3, lytp);
addView(image4, lytp);
addView(image5, lytp);
int len = ar.length();                                    //获取数组的长度
resids = new int[len];                                    //初始化加载图片的数组
for (int i = 0; i < len; i++) {
    resids[i] = ar.getResourceId(i, 0);
}                                                         //通过for循环把图片读取出来
```

（6）设置单击左侧或右侧按钮时图片飞出的动画。在 res 文件夹中创建 anim 文件夹，在创建好的 anim 文件夹中编写相应的动画文件，并在 MyImageLoading 类中调用，关键代码如下：

```
/**
 * 设置图片移除的动画
 */
private void setOutAnimation(Context cnt, ImageView iv, int i) {
    if (i == 1) {                                         //如果单击的是右侧的按钮
        //使图片从右侧飞出
        iv.setAnimation(AnimationUtils.loadAnimation(cnt, R.anim.right_out));
    } else {                                              //如果单击的是左侧的按钮
        //使图片从左侧飞出
        iv.setAnimation(AnimationUtils.loadAnimation(cnt, R.anim.left_out));
    }
}
```

（7）在 MyImageLoading 类中，添加按钮的单击事件，并将相关的参数传递给 MyImageLoading，关键代码如下：

```
/**
 * 定义左右按钮的单击监听事件
 **/
btnListener = new OnClickListener() {
   @Override
   public void onClick(View v) {
      int type = 0;
      /**
       * 单击不同按钮执行不同的操作，并传递相关的参数
       **/
      if (v.getId() == leftClickArea.getId()) {
         type = -1;                                       //如果单击左侧的按钮
         imageLoading.leftBtn(getActivity(),type); //调用imageLoading类中的leftBtn()方法
      } else if (v.getId() == rightClickArea.getId()) {
         type = 1;                                        //如果单击右侧的按钮
```

```
16            imageLoading.rightBtn(getActivity(),type);  //调用imageLoading类中的rightBtn()方法
17        }
18    }
19 };
```

扩展学习

在单击左侧或者右侧的按钮时，可以使图片从不同的方向飞出，请尝试实现让图片从两侧飞出的速度变慢的效果。

实例 052　仿微博发表动态

应用场景：发布信息页

源码位置：光盘\Code\06\052

实例说明

在微信、QQ 空间和微博中发表信息，都是大家比较熟悉的功能。本实例将实现发表评价页，主要是完成编辑评价文字以及选择多张图片的功能。运行实例，效果如图 6.7 所示；单击“+”号图片，弹出图片选择弹窗，如图 6.8 所示；单击“从相册中选取”按钮，进入图片选取页面，如图 6.9 所示；单击要选择的图片后，再单击“完成”按钮，返回到首页，如图 6.10 所示。

图 6.7　首页效果

图 6.8　选择相册

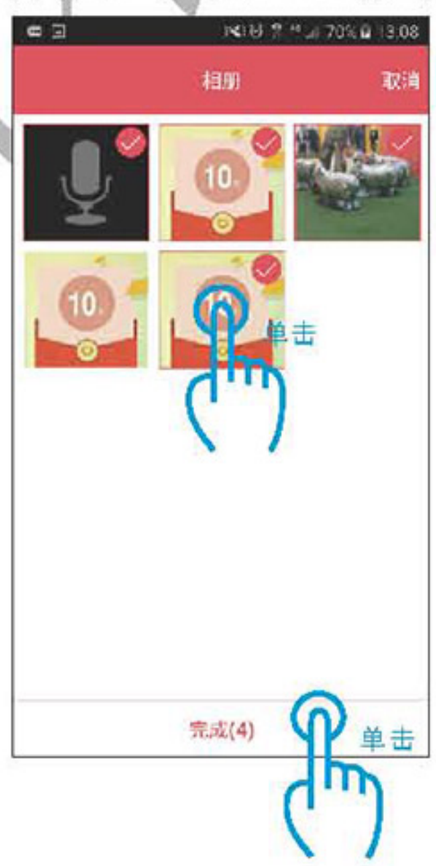


图 6.9　选择图片

图 6.10　选择完成后首页

关键技术

本实例主要使用 GridView 类中的 setAdapter() 方法、AdapterView 类的 setOnItemClickListener() 方法、Intent 类的 Intent() 方法、putExtra() 方法，实现发表动态功能。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“NineIV”，公司域名为“mingrisoft.com”。

（2）导入资源文件：复制“光盘\Code\06\052\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）在“app\java\com\mingrisoft\nineiv”文件夹下，默认生成的 MainActivity 类中，初始化显示图片控件，并绑定单击事件。代码如下：

```
//初始化控件
noScrollgridview = (GridView)findViewById(R.id.noScrollgridview);
//设置gridview分割线为透明
noScrollgridview.setSelector(new ColorDrawable(Color.TRANSPARENT));
//初始化适配器
adapter = new GridAdapter(this);
//更新数据
adapter.update();
//绑定图片数据
noScrollgridview.setAdapter(adapter);
//绑定单击事件
noScrollgridview.setOnItemClickListener(new OnItemClickListener()
{
    public void onItemClick(AdapterView<?> arg0, View arg1, int arg2, long arg3)
    {
        //判断单击的是否是图片
        if (arg2 == Bimp.bmp.size())
        {
            //显示选择提示窗
            new PopupWindows(PublishedActivity.this, noScrollgridview);
        }
        else
        {
            //进入图片预览页面
            Intent intent = new Intent(PublishedActivity.this, PhotoActivity.class);
            //传递图片标识
            intent.putExtra("ID", arg2);
            startActivity(intent);
        }
    }
});
```

（4）在 MainActivity 类中，开启相机选择照片。代码如下：

```
public void photo() {
    //开启相机
    Intent openCameraIntent = new Intent(MediaStore.ACTION_IMAGE_CAPTURE);
    //创建文件
```

明日科技

```
05      File file = new File(Environment.getExternalStorageDirectory() + "/myimage/",
06          String.valueOf(System.currentTimeMillis()) + ".jpg");
07      path = file.getPath();
08      Uri imageUri = Uri.fromFile(file);
09      //传递参数
10      openCameraIntent.putExtra(MediaStore.EXTRA_OUTPUT, imageUri);
11      //开始跳转确定回调标识
12      startActivityForResult(openCameraIntent, TAKE_PICTURE);
13  }
```

扩展学习

请尝试实现为发送按钮添加单击事件，显示弹出选择的图片信息以及文字内容。

实例 053　毛玻璃效果

应用场景：背景模糊页面

源码位置：光盘\Code\06\053

实例说明

iOS 在视觉方面能够吸引人，是因为在整个系统中使用了模糊效果。本实例将模仿 iOS 的毛玻璃效果，使用第三方的 BlurredView 开源控件，实现图片模糊效果。运行实例，效果如图 6.11 所示；向上滑动列表，背景图片模糊，如图6.12所示；再次向上滑动列表，背景图片更加模糊，如图6.13所示。

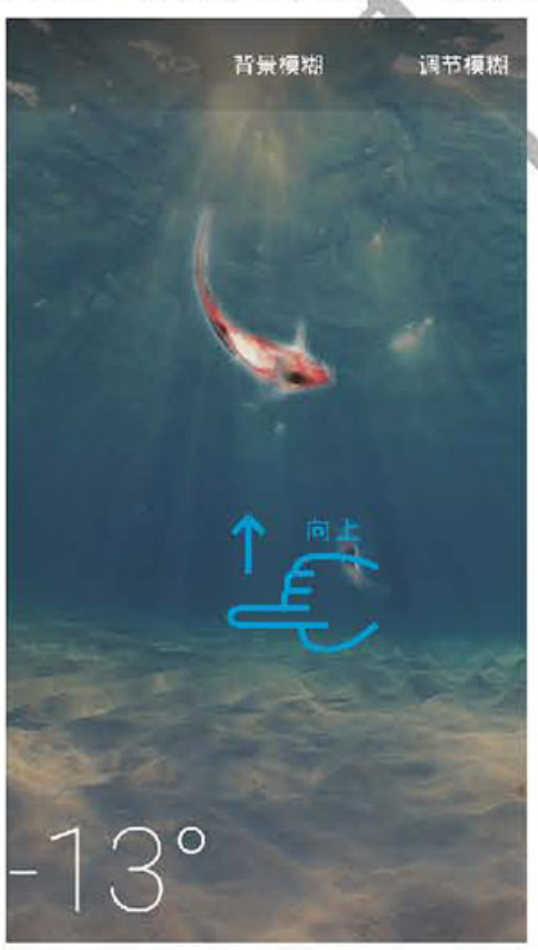


图 6.11　图片初始效果

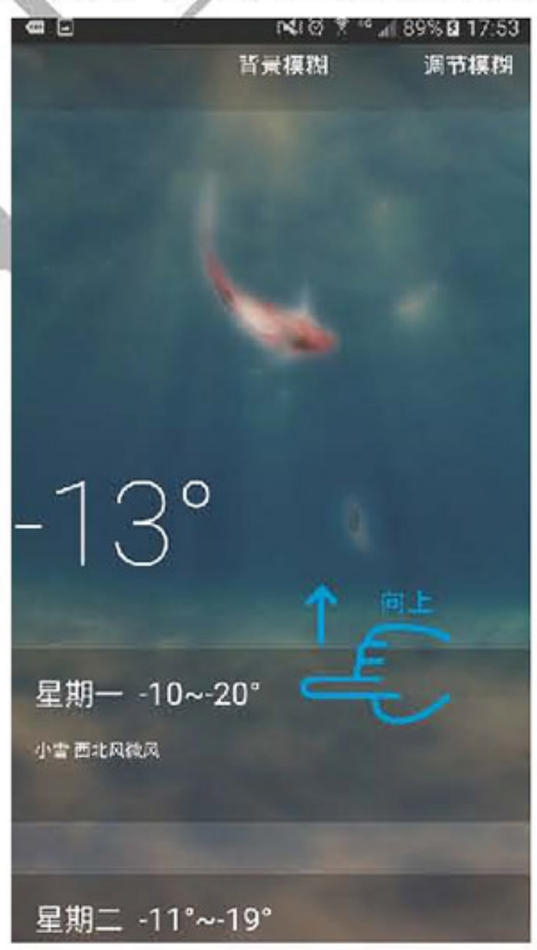
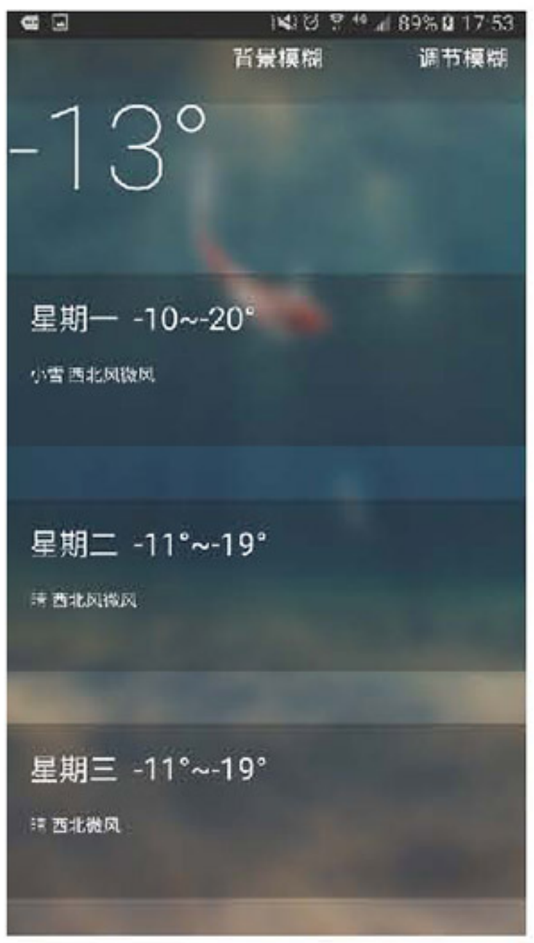


图 6.12　向上滑动效果



6.13　向上滑动到底部效果

关键技术

本实例主要使用 RecyclerView 类中的 addOn ScrollListener() 方法、setAdapter() 方法，View 类中的 setOnClickListener() 方法，BlurredView 类中的 setBlurredLevel() 方法，实现图片模糊的功能。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“MBLIV”，公司域名为“mingrisoft.com”。

（2）导入资源文件：复制“光盘\Code\06\053\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）使用 RecyclerView 控件和第三方的开源控件 BlurredView 实现图片模糊效果时，需要先在 build.gradle 配置文件的 dependencies{} 中加入引用。代码如下：

```
dependencies {
    compile fileTree(dir: 'libs', include: ['*.jar'])
    androidTestCompile('com.android.support.test.espresso:espresso-core:2.2.2', {
        exclude group: 'com.android.support', module: 'support-annotations'
    })
    compile 'com.android.support:appcompat-v7:24.2.1'
    testCompile 'junit:junit:4.12'
    compile 'com.qiushui:blurredview:0.8.1'
    compile 'com.android.support:design:24.0.0'
    compile 'com.android.support:cardview-v7:24.0.0'
    compile 'com.android.support:recyclerview-v7:24.0.0'
}
```

（4）在“app\java\com\mingrisoft\mbliv”文件夹下默认生成的 MainActivity 类中，监听 RecyclerView 控件的滑动事件，从而设置监听级别。代码如下：

```
//监听滑动事件
mRecyclerView.addOnScrollListener(new RecyclerView.OnScrollListener() {
    //滑动完成改变调用
    @Override
    public void onScrollStateChanged(RecyclerView recyclerView, int newState) {
        super.onScrollStateChanged(recyclerView, newState);
    }
    //滑动时调用
    @Override
    public void onScrolled(RecyclerView recyclerView, int dx, int dy) {
        super.onScrolled(recyclerView, dx, dy);
        mScrollerY += dy;
        if (Math.abs(mScrollerY) > 1000) {
            //设置图片上移距离
            mBlurredView.setBlurredTop(100);
            mAlpha = 100;
```

```
17          } else {
18              //设置图片上移距离
19              mBlurredView.setBlurredTop(mScrollerY / 10);
20              //获取绝对值
21              mAlpha = Math.abs(mScrollerY) / 10;
22          }
23          mBlurredView.setBlurredLevel(mAlpha); //决定模糊的级别，alpha在0到100之间
24      }
25  });
```

扩展学习

在设置背景模糊的界面效果时，本实例是在向上滑动的过程中使图片模糊，请尝试实现先让图片模糊，然后在向上滑动的过程中使其清晰。

实例 054　拼接图片

应用场景：图片编辑

源码位置：光盘\Code\06\054

实例说明

本实例将实现多张图片合成一张图片的功能。运行实例，效果如图6.14所示；单击首页中的“+”，可在相应的位置添加选择的图片，如图 6.15 所示；选择完成所有图片，将显示如图 6.16 所示的界面；单击“分享”按钮即可分享合成的图片，如图 6.17 所示。

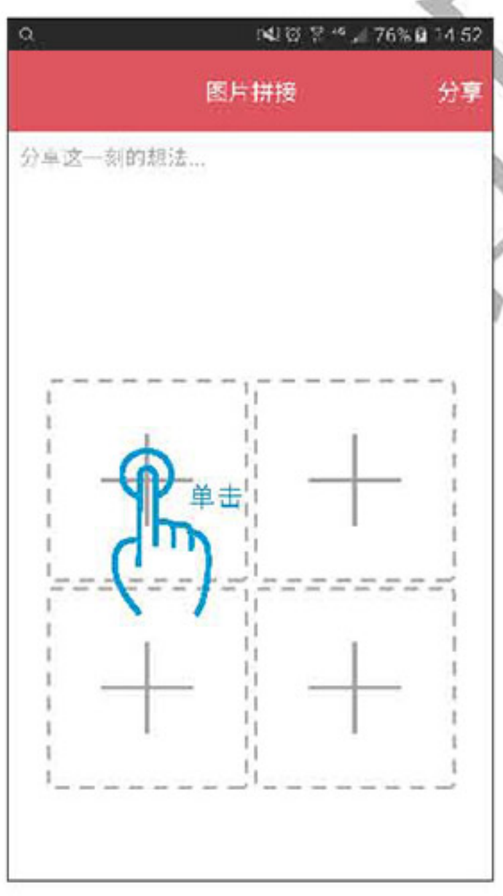


图 6.14　首页效果

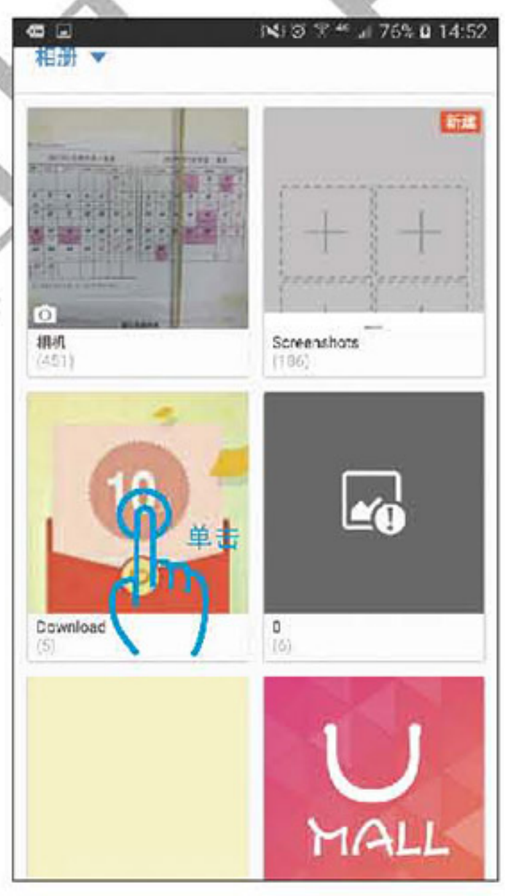


图 6.15　选择图片页

图 6.16　图片拼合完成效果

图 6.17　分享页面

关键技术

本实例主要使用 View 类中的 setOnClickListener() 方法、File 类中的 File() 方法、Uri 类中的 fromFile() 方法、FragmentActivity 类中的 startActivityForResult() 方法和 Canvas 类中的 drawBitmap() 方法，实现选择图片功能。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“ImagePJ”，公司域名为“mingrisoft.com”。

（2）导入资源文件：复制“光盘\Code\06\054\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）在“app\java\com\mingrisoft\imagepj”文件夹下创建 ImageUtil 工具类用于拼接图片，在其中创建 add2Bitmap() 方法用于横向拼接图片。代码如下：

```
/**
 * 横向拼接
 * <功能详细描述>
 * @param first 第一张图片
 * @param second 第二张图片
 * @return
 */
public static Bitmap add2Bitmap(Bitmap first, Bitmap second) {
    int width = first.getWidth() + second.getWidth();
    int height = Math.max(first.getHeight(), second.getHeight());
    //创建一个新图，高度和原图片高度一样，宽度为2张图片合起来的宽度
    Bitmap result = Bitmap.createBitmap(width, height, Bitmap.Config.ARGB_8888);
    //创建图片画布
    Canvas canvas = new Canvas(result);
    //绘制第一张图片
    canvas.drawBitmap(first, 0, 0, null);
    //绘制第二张图片
    canvas.drawBitmap(second, first.getWidth(), 0, null);
    return result;
}
```

（4）在“app\java\com\mingrisoft\imagepj”文件夹下，默认生成的 MainActivity 类中，添加 onActivity Result() 方法用于接收回调信息，并将接收到的从相册选择的图片添加到控件中。代码如下：

```
if (data != null) {                                          //判断数据是否为空
    Uri url1 = data.getData();                               //获得图片的uri
    try {
        String[] proj = {MediaStore.Images.Media.DATA};
        //Android多媒体数据库的封装接口
```

```
06          Cursor cursor = managedQuery(url1, proj, null, null, null);
07          //获得用户选择的图片的索引值
08          int column_index = cursor.getColumnIndexOrThrow(MediaStore.Images.Media.DATA);
09          //将光标移至开头，这个很重要，不小心很容易引起越界
10          cursor.moveToFirst();
11          //根据索引值获取图片路径
12          String path = cursor.getString(column_index);
13          bm1 = Bimp.revitionImageSize(path);              //获取bitmap
14          if (getBitmapDegree(path) != 0) {
15              bm1 = rotateBitmapByDegree(bm1, getBitmapDegree(path));
16          }
17          //设置图片
18          iV1.setImageBitmap(bm1);
19          //设置图片占满控件
20          iV1.setScaleType(ImageView.ScaleType.FIT_XY);
21      } catch (IOException e) {
22          e.printStackTrace();
23      }
24  }
```

扩展学习

请尝试把添加的文字拼接到图片中。

实例 055　扭曲变形效果

应用场景：图片页面

源码位置：光盘\Code\06\055

实例说明

本实例通过自定义控件实现图片的单击位置扭曲，并且在单击位置显示红色圆圈，红色圆圈出现后将自动放大并慢慢消失。运行实例，初始效果如图 6.18 所示；单击图片上的任一区域，该区域会出现扭曲变形效果，如图 6.19 所示；在界面中连续多次单击，界面效果如图 6.20 所示。

关键技术

本实例主要使用 View 类中的 onTouchEvent() 方法，BitmapFactory 类中的 decodeResource() 方法、Paint 类中的 setColor() 方法、setAlpha() 方法、setStyle() 方法，实现扭曲图片变形的效果。

明日科技

图 6.18　初始效果

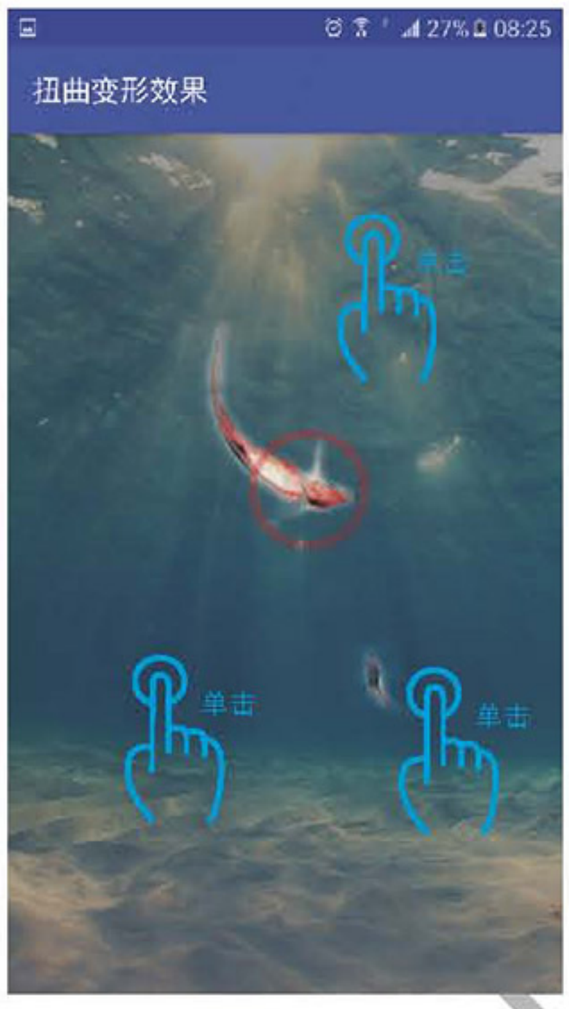


图 6.19　扭曲效果

图 6.20　多次单击后效果

实现过程

（1）打开 Android Studio，新建一个项目，命名为“NQBXImage”，公司域名为“mingrisoft.com”。

（2）导入资源文件：复制“光盘\Code\06\055\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）在“app\java\com\mingrisoft\nqbximage”文件夹下创建 MyView 类，继承 View 类，实现自定义扭曲图片控件。

（4）在“app\java\com\mingrisoft\nqbximage”文件夹下默认生成的 MainActivity 类中，加载 MyView 自定义控件类。代码如下：

```
01  public class MainActivity extends AppCompatActivity {
02      @Override
03      protected void onCreate(Bundle savedInstanceState) {
04          super.onCreate(savedInstanceState);
05          //初始化布局，装载自定义布局
06          setContentView(new MyView(this , R.drawable.bj));
07      }
08  }
```

（5）在 MyView 类中初始化画笔类，用于绘制圆环。代码如下：

```
01  private Paint initPaint(int alpha, float width) {
02      Paint paint = new Paint();                          //初始化画笔
03      paint.setAntiAlias(true);                           //抗锯齿
04      paint.setStrokeWidth(width);                        //描边宽度
```

```
05        paint.setStyle(Paint.Style.STROKE);          //圆环
06        paint.setAlpha(alpha);                       //透明度
07        paint.setColor(Color.RED);                   //颜色
08        return paint;
09    }
```

扩展学习

本实例实现单击后显示红色圆形图案，请尝试修改程序，使单击后显示其他颜色的方形。

实例 056　为图片添加特殊效果

应用场景：图片处理

源码位置：光盘\Code\06\056

实例说明

本实例将实现对图片进行特效处理，主要为灰度、怀旧和冰冻等效果。运行实例，效果如图 6.21 所示；单击“灰度处理”按钮，显示灰度图片，如图 6.22 所示；单击“怀旧处理”按钮，显示怀旧处理后的图片，如图 6.23 所示；单击“冰冻处理”按钮，显示冰冻处理后的图片，如图 6.24 所示。

图 6.21　运行效果

图 6.22　灰度处理效果

图 6.23　怀旧处理效果

图 6.24　冰冻处理效果

关键技术

本实例主要使用 BitmapUtil 类中的 huidu() 方法，Canvas 类中的 drawBimap() 方法，View 类中的 setVisibility() 方法、setOnClickListener() 方法，实现图片特效处理。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“BitmapChange”，公司域名为“mingrisoft.com”。

（2）导入资源文件：复制“光盘\Code\06\056\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）在“app\java\com\mingrisoft\bitmapchange”文件夹下创建 BitmapUtil 类，用于对图片进行特效处理，以灰度处理为例。代码如下：

```
//图片灰度化处理
public static Bitmap huidu(Bitmap bmSrc) {
    //获取图片的长和宽
    int width = bmSrc.getWidth();
    int height = bmSrc.getHeight();
    //创建目标灰度图像
    Bitmap bmpGray = null;
    bmpGray = Bitmap.createBitmap(width, height, Bitmap.Config.RGB_565);
    //创建画布
    Canvas c = new Canvas(bmpGray);
    Paint paint = new Paint();
    ColorMatrix cm = new ColorMatrix();
    cm.setSaturation(0);
    ColorMatrixColorFilter f = new ColorMatrixColorFilter(cm);
    paint.setColorFilter(f);
    c.drawBitmap(bmSrc, 0, 0, paint);
    return bmpGray;
}
```

（4）在“app\java\com\mingrisoft\bitmapchange”文件夹下默认生成的 MainActivity 类中，为按钮绑定单击事件，在单击事件中隐藏单击后的按钮，并显示处理后的图片。以灰度处理为例，代码如下：

```
//获取图片资源转换为Bitmap类型
bitmap = BitmapFactory.decodeResource(getResources(),R.drawable.bb);
//单击事件
huiduBT.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        huiduBT.setVisibility(View.GONE);                    //隐藏处理按钮
        huiduIV.setImageBitmap(BitmapUtil.huidu(bitmap));    //显示灰度图片
    }
});
```

扩展学习

在图片处理等待的过程中显示加载框能给用户更好的体验，请尝试添加图片处理加载框。

明日科技

实例 057　图片圆角处理

应用场景：图片处理

源码位置：光盘\Code\06\057

实例说明

本实例将实现对图片进行特效处理的功能，主要是对图片进行圆角处理。运行实例，初始效果如图 6.25 所示；单击“选择图片”按钮，进入相册选择图片，图片选择完成后的效果如图 6.26 所示；单击“图片圆角处理”按钮，图片以圆角形式显示，如图 6.27 所示。

图 6.25　初始效果

图 6.26　选择图片效果

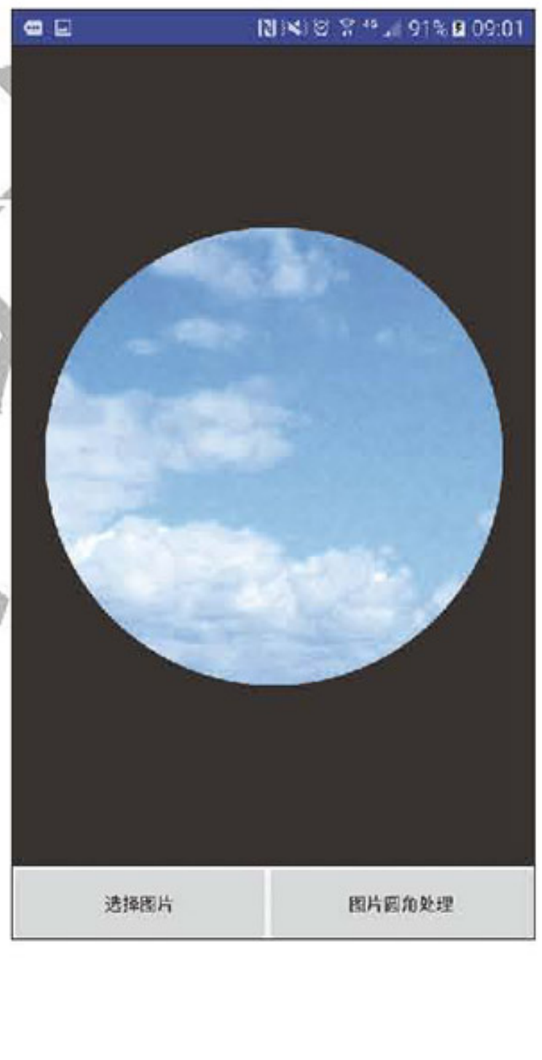


图 6.27　图片圆角效果

关键技术

关键技术

本实例主要使用 ImageView 类中的 setImageBitmap() 方法、BitmapFactory 类中的 decodeStream() 方法、Canvas 类中的 drawBimap() 方法、View 类中的 setOnClickListener() 方法，实现图片圆角处理。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“ImageFramer”，公司域名为“mingrisoft.com”。

（2）在“app\java\com\mingrisoft\imageframer”文件夹下默认生成的 MainActivity 中，为“选择图片”按钮添加单击事件监听，实现单击后打开图库选择图片。代码如下：

```
01  openImageBn.setOnClickListener(new OnClickListener() {
02      @Override
```

```
    public void onClick(View v) {
        //启动相册
        Intent intent = new Intent(Intent.ACTION_PICK,
                android.provider.MediaStore.Images.Media.EXTERNAL_CONTENT_URI);
        startActivityForResult(intent, IMAGE_OPEN);
    }
});
```

（3）在 MainActivity 类中实现 onActivityResult() 方法，用于接收选择图片后的信息，并设置显示选择的图片。代码如下：

```
//接收回调信息
protected void onActivityResult(int requestCode, int resultCode, Intent data) {
    super.onActivityResult(requestCode, resultCode, data);
    if(resultCode==RESULT_OK && requestCode == IMAGE_OPEN) {
        //图片路径
        Uri imageFileUri = data.getData();
        //获得系统信息
        DisplayMetrics dm = new DisplayMetrics();
        getWindowManager().getDefaultDisplay().getMetrics(dm);
        int width = dm.widthPixels;                    //手机屏幕水平分辨率
        int height = dm.heightPixels;                  //手机屏幕垂直分辨率
        try {
            //只想获得图片的尺寸，但又不想将其加载到内存，设置inJustDecodeBounds的属性为true
            BitmapFactory.Options bmpFactoryOptions = new BitmapFactory.Options();
            bmpFactoryOptions.inJustDecodeBounds = true;
            //获得Bitmap对象
            bmp = BitmapFactory.decodeStream(getContentResolver().openInputStream(imageFileUri),
                    null, bmpFactoryOptions);
            int heightRatio = (int)Math.ceil(bmpFactoryOptions.outHeight/(float)height);
            int widthRatio = (int)Math.ceil(bmpFactoryOptions.outWidth/(float)width);
            //inSampleSize表示图片占原图比例，1表示原图的尺寸一样大
            if(heightRatio > 1 && widthRatio > 1) {
                if(heightRatio > widthRatio) {
                    bmpFactoryOptions.inSampleSize = heightRatio;
                }
                else {
                    bmpFactoryOptions.inSampleSize = widthRatio;
                }
            }
            //获取图片
            bmpFactoryOptions.inJustDecodeBounds = false;
            //获取Bitmap对象
```

```
            bmp = BitmapFactory.decodeStream(getContentResolver().openInputStream(imageFileUri),
                    null, bmpFactoryOptions);
            //选择图片
            imageShow.setImageBitmap(bmp);
        }
        catch(FileNotFoundException e) {
            e.printStackTrace();
        }
    }
}
```

（4）在 MainActivity 类中创建 getRoundedCornerBitmap() 方法，用于对图片进行处理，主要是获取圆角类型的 Bitmap 对象。代码如下：

```
//生成圆角图片
private Bitmap getRoundedCornerBitmap(Bitmap bitmap)
{
    Bitmap roundBitmap = Bitmap.createBitmap(bitmap.getWidth(),
            bitmap.getHeight(), Config.ARGB_8888);
    Canvas canvas = new Canvas(roundBitmap);
    int color = 0xff424242;
    Paint paint = new Paint();
    //设置圆形半径
    int radius;
    if (bitmap.getWidth() > bitmap.getHeight()) {
        radius = bitmap.getHeight() / 2;
    }
    else {
        radius = bitmap.getWidth() / 2;
    }
    //绘制圆形
    paint.setAntiAlias(true);
    canvas.drawARGB(0, 0, 0, 0);
    paint.setColor(color);
    canvas.drawCircle( bitmap.getWidth() / 2, bitmap.getHeight() / 2, radius, paint);
    paint.setXfermode(new PorterDuffXfermode(Mode.SRC_IN));
    canvas.drawBitmap(bitmap, 0, 0, paint);
    return roundBitmap;
}
```

扩展学习

请尝试添加分享功能，然后分享生成后的圆角图片。

实例 058 滑动浏览图片

应用场景：图片浏览

源码位置：光盘\Code\06\058

实例说明

本实例基于 Universal-Image-Loader 开源图片处理框架及 Facebook 推出的 Rebound 物理和动画库实现，以完成滑动浏览图片的效果。运行实例，初始效果如图 6.28 所示；滑动卡片，滑动效果如图 6.29 所示；滑动超过一定距离后将显示下一张图片，如图 6.30 所示。

图 6.28 初始效果

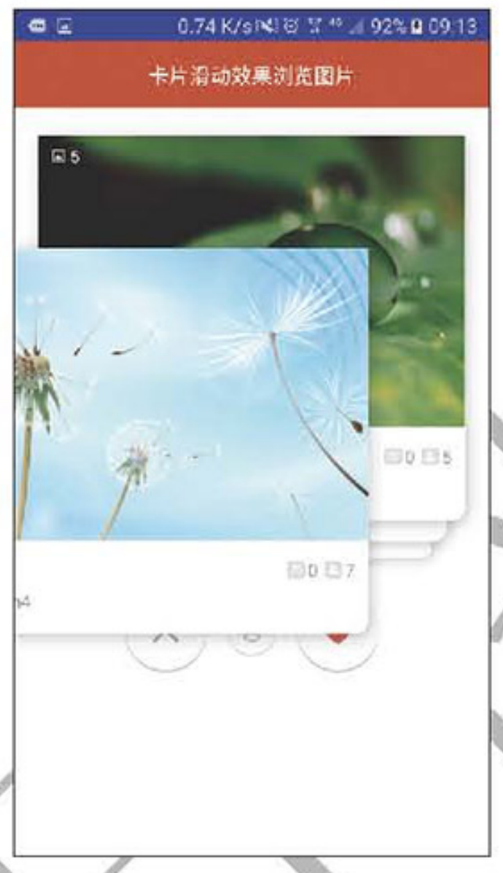


图 6.29 滑动卡片

图 6.30 显示下一张卡片

关键技术

关键技术

本实例主要使用 CardSlidePanel 类中的 setCardSwitchListener() 方法，GestureDetectorCompat 类中的 GestureDetectorCompat () 方法，CardItemView 类中的 shouldCapture () 方法、setCurrentSpringPos () 方法、animTo () 方法，实现滑动浏览图片的效果。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“AndroidCards”，公司域名为“mingrisoft.com”。

（2）导入资源文件：复制“光盘\Code\06\058\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。复制“光盘\06\058\src\assets”中的所有文件夹，粘贴到“app”目录下。

（3）在 build.gradle 文件中，加入开源的第三方依赖库。代码如下：

```
compile 'com.nostra13.universalimageloader:universal-image-loader:1.9.5'
compile 'com.facebook.rebound:rebound:0.3.8'
```

（4）在“app\java\com\mingrisoft\androidcards”文件夹下默认生成的 MainActivity 类中，配置 ImageLoaderConfiguration 参数用于图片的加载，参数只能配置一次，如多次配置，则默认第一次的配置参数。代码如下：

```
@SuppressWarnings("deprecation")
private void initImageLoader() {
    ImageLoaderConfiguration config = new ImageLoaderConfiguration.Builder(
            this)
            .memoryCacheExtraOptions(480, 800)
            //默认设备屏幕尺寸
            .threadPoolSize(3)
            .threadPriority(Thread.NORM_PRIORITY - 1)
            //设置图片加载和显示队列处理的类型，默认为QueueProcessingType.FIFO
            .tasksProcessingOrder(QueueProcessingType.FIFO)
            //设置拒绝在内存中缓存一个图片多个大小，默认为允许，
            //(同一个图片URL)根据不同大小的imageview保存不同大小的图片
            .denyCacheImageMultipleSizesInMemory()
            //设置内存缓存，默认为当前应用可用内存的1/8
            .memoryCache(new LruMemoryCache(2 * 1024 * 1024))
            //设置内存缓存的最大范围，默认为当前应用可用内存的1/8
            .memoryCacheSize(2 * 1024 * 1024).memoryCacheSizePercentage(13)
            //缓冲大小
            .discCacheSize(50 * 1024 * 1024)
            //缓冲文件数目
            .discCacheFileCount(100)
            //设置硬盘缓存文件名生成规范
            //默认为new HashCodeFileNameGenerator()
            .discCacheFileNameGenerator(new HashCodeFileNameGenerator())
            //设置图片下载器
            //默认为 DefaultConfigurationFactory.createBitmapDisplayer()
            .imageDownloader(new BaseImageDownloader(this))
            //设置默认的图片显示选项
            //默认为DisplayImageOptions.createSimple()
            .defaultDisplayImageOptions(DisplayImageOptions.createSimple())
            //输出DebugLogs
            .writeDebugLogs().build();

    //单例ImageLoader类的初始化，全局初始化此配置
    ImageLoader imageLoader = ImageLoader.getInstance();
    imageLoader.init(config);
}
```

（5）在“app\java\com\mingrisoft\androidcards”文件夹下创建 CardFragment 类，用于显示控件。

（6）在“app\java\com\mingrisoft\androidcards”文件夹下创建 CardSlidePanel 类，用于卡片滑动面板逻辑的实现。在其中创建接口类 CardSwitchListener，用于处理回调方法获取卡片状态。代码如下：

```
/**
 * 卡片回调接口
 */
public interface CardSwitchListener {
    /**
     * 新卡片显示回调
     *
     * @param index 在最顶层显示的卡片的index
     */
    public void onShow(int index);
    /**
     * 卡片飞向两侧回调
     *
     * @param index 飞向两侧的卡片数据的index
     * @param type  飞向左边或右边
     */
    public void onCardVanish(int index, int type);
    /**
     * 卡片单击事件
     *
     * @param cardImageView 卡片上的图片view
     * @param index         单击到的index
     */
    public void onItemClick(View cardImageView, int index);
}
```

扩展学习

将本实例中的图片替换成自己的图片。

实例 059 ViewPager 实现画廊效果

应用场景：宣传页

源码位置：光盘\Code\06\059

视频位置：光盘\Video\06\059

实例说明

本实例使用 ViewPager 类实现滑动动画效果。运行实例，默认界面如图 6.31 所示；向左滑动，查看下一张图片，如图 6.32 所示。

明日科技

图 6.31　默认界面

图 6.32　查看下一张图片

关键技术

关键技术

本实例主要使用 ViewPager 类中的 setOffscreenPageLimit() 方法、setPageTransformer() 方法，以及 View 类中的 setScale() 方法、Bitmap 类中的 createBitmap() 方法，实现切换时的动画效果。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“HLDemo”，公司域名为“mingrisoft.com”。

（2）导入资源文件：复制“光盘\Code\06\059\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）在“app\res\layout”文件夹中默认生成的 activity_main.xml 布局文件中，添加 ViewPager 类，用于显示滑动图片。

（4）在“app\java\com\mingrisoft\hldemo”文件夹下新建 ViewPageTransformer 类，用于对 ViewPager 添加滑动动画。

（5）在“app\java\com\mingrisoft\sjzlistview”文件夹下默认生成的 MainActivity 类中，获取要显示的图片添加到 ImageVeiw 控件中，再把 ImageView 添加到数组中。代码如下：

```
01  private void initData() {
02      //装载图片控件数组
03      imageViews = new ArrayList<>();
04      //新建图片控件
05      ImageView first = new ImageView(MainActivity.this);
06      //设置图片背景
07      first.setImageBitmap(ImageUtil.getReverseBitmapById(R.drawable.book1, MainActivity.this));
```

```
        ImageView second = new ImageView(MainActivity.this);
        second.setImageBitmap(ImageUtil.getReverseBitmapById(R.drawable.book2, MainActivity.this));
        ImageView third = new ImageView(MainActivity.this);
        third.setImageBitmap(ImageUtil.getReverseBitmapById(R.drawable.book3, MainActivity.this));
        ImageView fourth = new ImageView(MainActivity.this);
        fourth.setImageBitmap(ImageUtil.getReverseBitmapById(R.drawable.book4, MainActivity.this));
        //添加图片显示控件到数组
        imageViews.add(first);
        imageViews.add(second);
        imageViews.add(third);
        imageViews.add(fourth);
    }
```

（6）在“app\java\com\mingrisoft\sjzlistview”文件夹下新建 ImageUtil 图片工具类，其中使用 scaleImg() 方法对图片进行压缩处理，防止内存溢出错误。代码如下：

```
protected static Bitmap scaleImg(Context context,int resId, int newWidth, int newHeight) {
    //图片源
    Bitmap bm = BitmapFactory.decodeResource(context.getResources(), resId);
    //获得图片的宽和高
    int width = bm.getWidth();
    int height = bm.getHeight();
    //设置想要的大小
    int newWidth1 = newWidth;
    int newHeight1 = newHeight;
    //计算缩放比例
    float scaleWidth = ((float) newWidth1) / width;
    float scaleHeight = ((float) newHeight1) / height;
    //获取想要缩放的matrix参数
    Matrix matrix = new Matrix();
    matrix.postScale(scaleWidth, scaleHeight);
    //获取新的图片
    Bitmap newbm = Bitmap.createBitmap(bm, 0, 0, width, height, matrix,
            true);
    //返回新图片
    return newbm;
}
```

（7）使用 getReverseBitmapById() 方法，实现图片翻转，以及将原图与翻转后的图片合成一张图片。代码如下：

```
public static Bitmap getReverseBitmapById(int resId, Context context) {
    //压缩图片到指定宽、高，防止OOM（内存溢出）
    Bitmap sourceBitmap = scaleImg(context,resId, 600, 800);
```

```
04        //绘制原图的下一半图片
05        Matrix matrix = new Matrix();
06        //缩放
07        //倒影翻转，将y轴坐标翻转
08        matrix.setScale(1, -1);
09        Bitmap inverseBitmap = Bitmap.createBitmap(sourceBitmap, 0, sourceBitmap.getHeight() / 2,
10                sourceBitmap.getWidth(), sourceBitmap.getHeight() / 3, matrix, false);
11        //合成图片
12        Bitmap groupbBitmap = Bitmap.createBitmap(sourceBitmap.getWidth(), sourceBitmap.
13                getHeight() + sourceBitmap.getHeight() / 3 + 50, sourceBitmap.getConfig());
14        //以合成图片为画布
15        Canvas gCanvas = new Canvas(groupbBitmap);
16        //将原图和倒影图片画在合成图片上
17        gCanvas.drawBitmap(sourceBitmap, 0, 0, null);
18        gCanvas.drawBitmap(inverseBitmap, 0, sourceBitmap.getHeight() + 100, null);
19        //添加遮罩
20        Paint paint = new Paint();
21        Shader.TileMode tileMode = Shader.TileMode.CLAMP;
22        LinearGradient shader = new LinearGradient(0, sourceBitmap.getHeight() + 100, 0,
23                groupbBitmap.getHeight(), Color.BLACK, Color.TRANSPARENT, tileMode);
24        paint.setShader(shader);
25        //获取矩形渐变区和图片的交集
26        paint.setXfermode(new PorterDuffXfermode(PorterDuff.Mode.DST_IN));
27        gCanvas.drawRect(0, sourceBitmap.getHeight() + 150, sourceBitmap.getWidth(),
28                groupbBitmap.getHeight(), paint);
29        //返回新图
30        return groupbBitmap;
31    }
```

扩展学习

请尝试修改 ViewPageTransformer 类的代码，以调试出不同的切换动画。

明日科技

第7章 图表处理

实例 060 书籍销量柱状统计图

应用场景：统计图

源码位置：光盘\Code\07\060

实例说明

为方便观察数据，经常会用图表的形式展示数据。本实例将实现通过使用第三方 MPAndroidChart 图表控件，完成数据销量统计表。运行实例，将显示如图 7.1 所示的界面；纵向放大的展示效果，如图 7.2 所示，横向放大的展示效果，如图 7.3 所示。

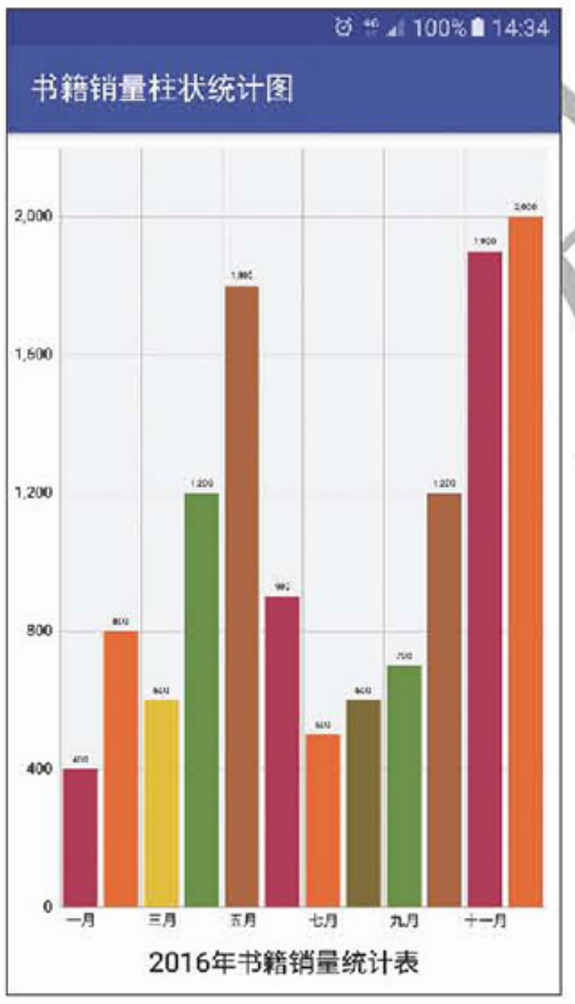


图 7.1 初始效果

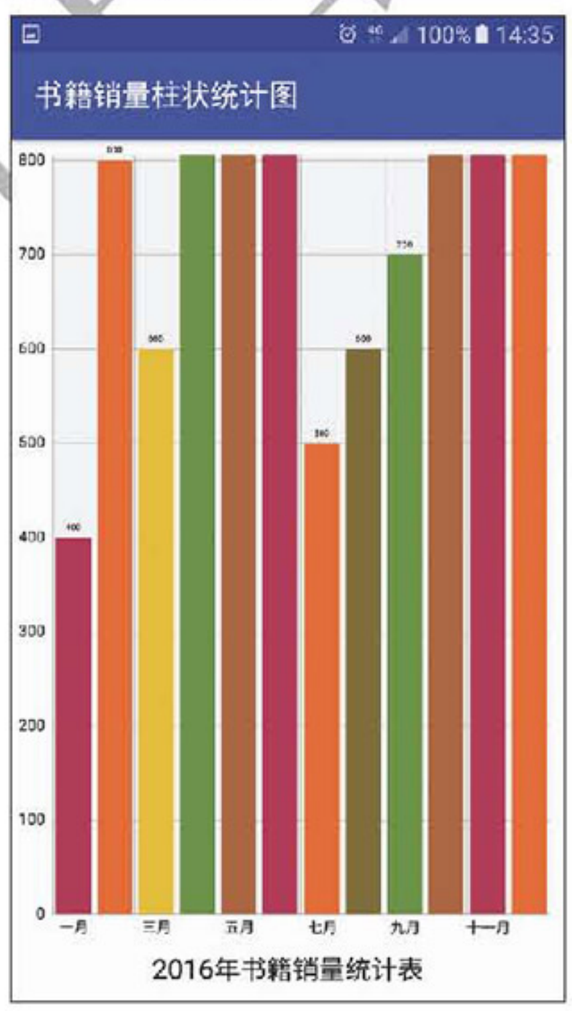


图 7.2 纵向放大展示效果

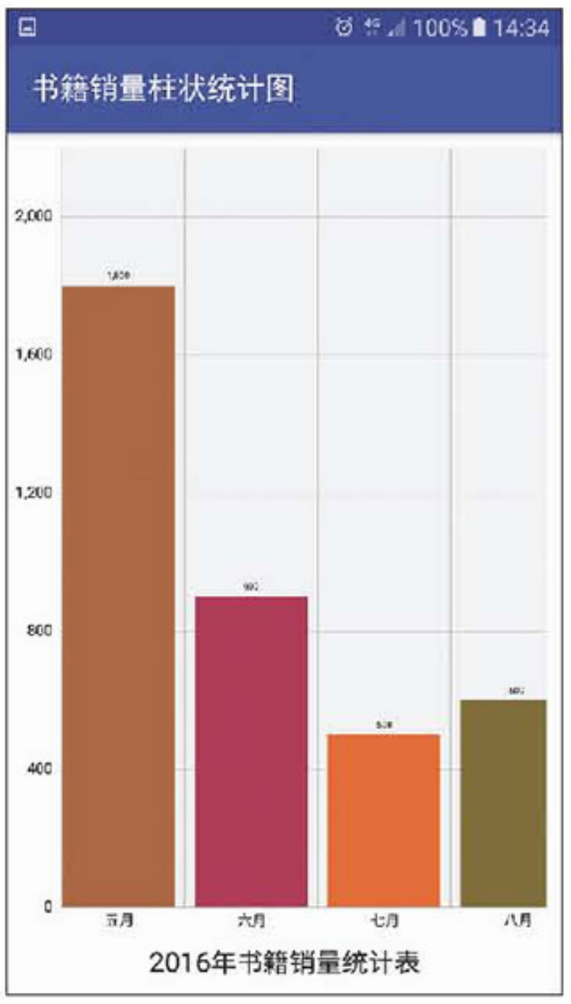


图 7.3 横向放大展示效果

关键技术

本实例主要使用 BarDataSet 类中的 BarDataSet() 方法、setColors() 方法，BarData 类中的 BarData() 方法，BarChart 类中的 setData() 方法，XAxis 类中的 setPosition() 方法，Chart 类中的 animateY () 方法，实现柱状统计图的效果。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“BarChart”，公司域名为“mingrisoft.com”。

（2）导入资源文件：复制“光盘\Code\07\060\src\libs”中的 mpandroidchartlibrary-2-1-6.jar 文件，粘贴到“app\libs”目录下，该文件为开源的图表类的 Jar 包。

（3）在“app\res\layout”目录下默认生成的 activity_main.xml 布局文件中，加入饼状图显示控件。代码如下：

```
<!--柱状图-->
<com.github.mikephil.charting.charts.BarChart
    android:id="@+id/bar_chart"
    android:layout_width="match_parent"
    android:layout_height="fill_parent"
    android:layout_gravity="center_horizontal"
    android:layout_above="@+id/title"
    />
```

（4）在“app\java\com\mingrisoft\barchart”文件夹下默认生成的 MainActivity 类中，配置柱状图显示数据。代码如下：

```
//添加数据
public void initEntriesData() {
    entries.add(new BarEntry(400f, 0));
    entries.add(new BarEntry(800f, 1));
    entries.add(new BarEntry(600f, 2));
    entries.add(new BarEntry(1200f, 3));
    entries.add(new BarEntry(1800f, 4));
    entries.add(new BarEntry(900f, 5));
    entries.add(new BarEntry(500f, 6));
    entries.add(new BarEntry(600f, 7));
    entries.add(new BarEntry(700f, 8));
    entries.add(new BarEntry(1200f, 9));
    entries.add(new BarEntry(1900f, 10));
    entries.add(new BarEntry(2000f, 11));
}
```

（5）在 MainActivity 类中，配置 x 轴显示数据。代码如下：

```
//添加x轴值
public void initLableData() {
    labels.add("一月");
```

```
04        labels.add("二月");
05        labels.add("三月");
06        labels.add("四月");
07        labels.add("五月");
08        labels.add("六月");
09        labels.add("七月");
10        labels.add("八月");
11        labels.add("九月");
12        labels.add("十月");
13        labels.add("十一月");
14        labels.add("十二月");
15    }
```

（6）在 MainActivity 类中，配置柱状图属性。代码如下：

```
01  public void show() {
02      //装载显示数据
03      dataset = new BarDataSet(entries, "书籍销量(本)");
04      dataset.setColors(ColorTemplate.COLORFUL_COLORS);
05      //封装x轴数据
06      BarData data = new BarData(labels, dataset);
07      //右侧y轴关闭
08      barChart.getAxisRight().setEnabled(false);
09      //x轴显示到下面
10      barChart.getXAxis().setPosition(XAxis.XAxisPosition.BOTTOM);
11      //装载数据
12      barChart.setData(data);
13      barChart.animateY(2000);
14      //设置右下角显示的文字
15      barChart.setDescription("");
16      //设置文字字号
17      barChart.setDescriptionTextSize(14f);
18      //设置表文字颜色
19      barChart.setDescriptionColor(Color.BLACK);
20      //设置比例图
21      Legend mLegend = barChart.getLegend();
22      //设置是否显示比例图
23      mLegend.setEnabled(false);
24  }
```

扩展学习

请尝试在 MainActivity 中改变柱状图的显示颜色。

实例 061　家庭支出分析饼状图

应用场景：分析图

源码位置：光盘\Code\07\061

视频位置：光盘\Video\07\061

实例说明

饼状图可以显示出每个数据系列在总体中所占的比例，并且每个数据具有唯一的颜色或图案。本实例主要使用第三方 MPAndroidChart 图表控件，实现家庭支出饼状图，方便分析家庭支出情况。运行实例，初始效果如图 7.4 所示；单击“行”区域，放大展示所选区域，如图 7.5 所示；单击“食”区域，放大展示所选区域，如图 7.6 所示。

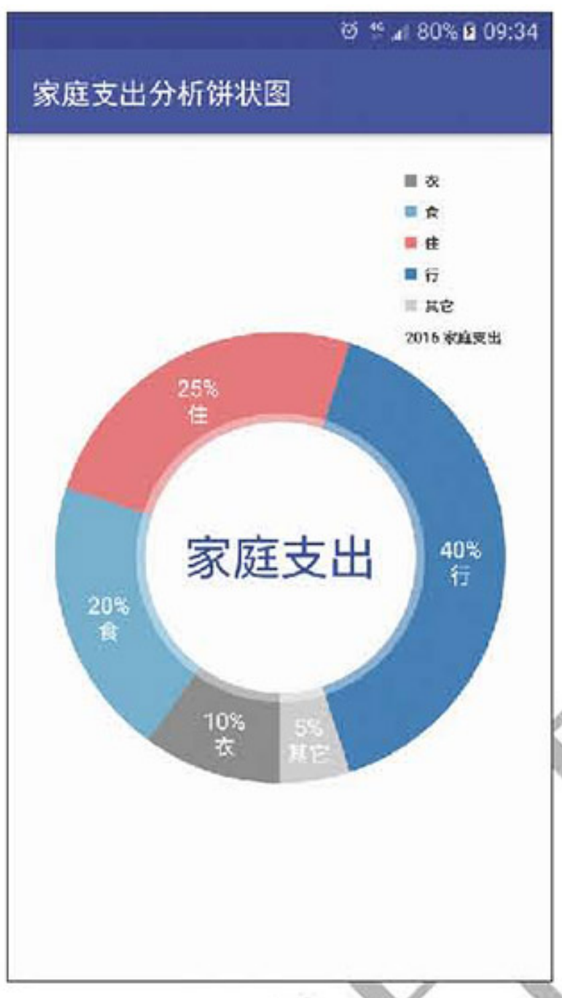


图 7.4　初始效果

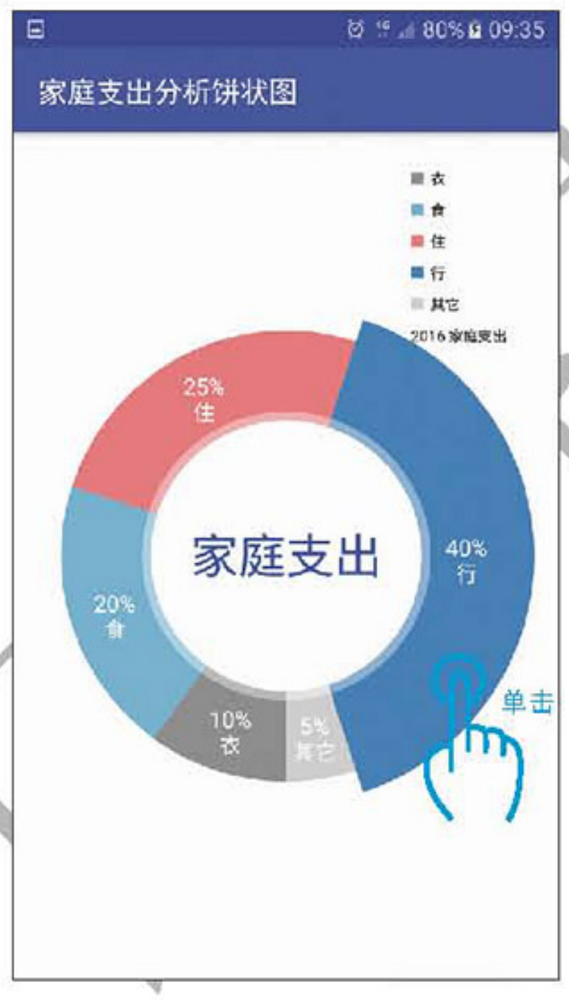


图 7.5　放大“行”分类

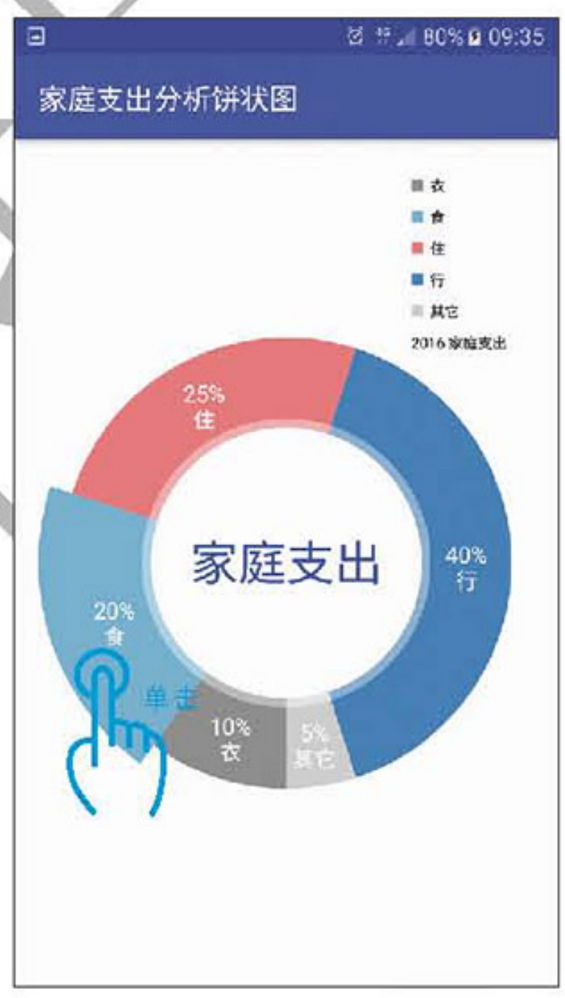


图 7.6　放大“食”分类

关键技术

本实例主要使用 PieDataSet 类中的 PieDataSet() 方法，PieChart 类中的 setHoleRadius() 方法，setCenterText() 方法、setCenterTextColor() 方法，Legend 类中的 setPosition() 方法，Chart 类中的 animateXY () 方法，PieDataSet 类中的 setSelectionShift () 方法，实现家庭支出情况饼状图。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“PieChar”，公司域名为“mingrisoft.com”。

（2）导入资源文件：复制“光盘\Code\07\061\src\libs”中的 mpandroidchartlibrary-2-1-6.jar 文件，粘贴到“app\libs”目录下，该文件为开源的图表类的 Jar 包。

（3）在“app\res\layout”目录下默认生成的 activity_main.xml 布局文件中，加入饼状图显示控件。代码如下：

```
<!--饼状图-->
<com.github.mikephil.charting.charts.PieChart
    android:id="@+id/pie_chart"
    android:layout_width="match_parent"
    android:layout_height="fill_parent"
    android:layout_gravity="center_horizontal"
    />
```

（4）在“app\java\com\mingrisoft\piechar”文件夹下默认生成的 MainActivity 类中，配置饼状图基础属性。代码如下：

```
//设置中间半透明
pieChart.setHoleColorTransparent(true);
//半径
pieChart.setHoleRadius(60f);
//半透明圈
pieChart.setTransparentCircleRadius(64f);
pieChart.setDescription(" ");
//饼状图中间可以添加文字
pieChart.setDrawCenterText(true);
pieChart.setDrawHoleEnabled(true);
//初始旋转角度
pieChart.setRotationAngle(90);
//可以手动旋转
pieChart.setRotationEnabled(true);
//显示成百分比
pieChart.setUsePercentValues(true);
//饼状图中间的文字
pieChart.setCenterText("家庭支出");
pieChart.setCenterTextColor(Color.BLUE);
pieChart.setCenterTextSize(16f);
pieChart.setCenterTextSizePixels(100f);
//设置数据
pieChart.setData(pieData);
//设置比例图
Legend mLegend = pieChart.getLegend();
//最右边显示
mLegend.setPosition(Legend.LegendPosition.RIGHT_OF_CHART);
mLegend.setXEntrySpace(7f);
mLegend.setYEntrySpace(5f);
//设置动画
pieChart.animateXY(1000, 1000);
```

（5）在 MainActivity 类中，配置饼状图模块颜色。代码如下：

```
//饼状图颜色
```

明日科技

```
02  colors.add(Color.GRAY);
03  colors.add(Color.rgb(114, 188, 223));
04  colors.add(Color.rgb(255, 123, 124));
05  colors.add(Color.rgb(57, 135, 200));
06  colors.add(Color.rgb(205, 205, 205));
07  pieDataSet.setColors(colors);
```

扩展学习

请尝试在 MainActivity 类中修改饼状图样式，将其设置为实心的饼状图。

实例 062　月支出麻点统计图

应用场景：统计图

源码位置：光盘\Code\07\062

实例说明

为方便观察数据，通常会用麻点图的形式展示数据。本实例主要使用第三方 MPAndroidChart 图表控件，完成月支出统计麻点统计图。运行实例，将显示如图 7.7 所示的界面；单击某一个信息，弹出信息提示，如图 7.8 所示。

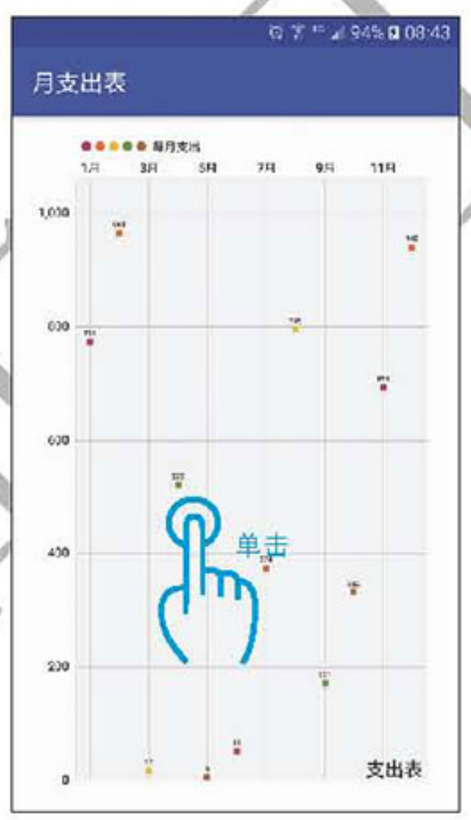


图 7.7　运行效果

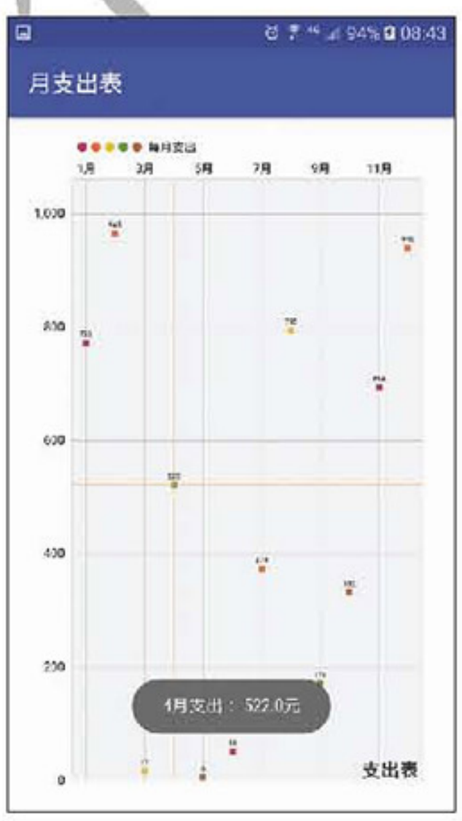


图 7.8　提示信息

关键技术

关键技术

本实例主要使用 ScatterDataSet 类中的 ScatterDataSet() 方法、setColors() 方法，ScatterChart 类中的 setData() 方法，Legend 类中的 setPosition() 方法，Chart 类中的 animateXY () 方法，实现通过麻点统计图显示数据。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“ScatterChart”，公司域名为“mingrisoft.com”。

（2）导入资源文件：复制“光盘\Code\07\062\src\libs”中的 mpandroidchartlibrary-2-1-6.jar 文件，粘贴到“app\libs”目录下，该文件为开源的图表类的 Jar 包。

（3）在“app\res\layout”目录下默认生成的 activity_main.xml 布局文件中，加入麻点图显示控件。代码如下：

```
<!--麻点图-->
<com.github.mikephil.charting.charts.ScatterChart
    android:id="@+id/scatter_chart"
    android:layout_width="match_parent"
    android:layout_height="fill_parent"
    android:layout_gravity="center_horizontal"
    />
```

（4）在“app\java\com\mingrisoft\scatterchart”文件夹下默认生成的 MainActivity 类中，配置麻点图显示数据。代码如下：

```
//数据集合
ArrayList<Entry> yVals = new ArrayList<>();
//初始化横坐标和纵坐标
final ArrayList<String> xVals = new ArrayList<>();
for (int i = 0; i < 12; i++) {
    xVals.add((i + 1) + "月");
    //产生数据
    yVals.add(new Entry(random.nextInt(1000), i));
}
ScatterDataSet scatterDataSet = new ScatterDataSet(yVals, "每月支出");
```

（5）在 MainActivity 类中，配置麻点图属性。代码如下：

```
//设置x轴位置
scatterChart.getXAxis().setPosition(XAxis.XAxisPosition.TOP);
//右侧y轴关闭
scatterChart.getAxisRight().setEnabled(false);
//设置y的最小值
scatterChart.getAxisLeft().setAxisMinValue(0.0f);
//设置纵向网格线条颜色
scatterChart.getXAxis().setGridColor(Color.GRAY);
//设置横向网格线条颜色
scatterChart.getAxisLeft().setGridColor(Color.GRAY);
//设置描述内容
scatterChart.setDescription("支出表");
//设置描述文字的字体
```

明日科技

```
14  scatterChart.setDescriptionTextSize(20.f);
15  //动画效果
16  scatterChart.animateXY(1000, 1000);
```

扩展学习

请尝试在 MainActivity 类中修改麻点图，将麻点颜色修改成红色。

实例 063　心电图

应用场景：医疗类 App

源码位置：光盘\Code\07\063

实例说明

心电图功能在医疗类 App 应用中比较常用。本实例将通过 SurfaceView+SurfaceHolder+Thread 开启线程绘制线实现心电图功能。运行实例，将显示如图 7.9 所示的界面；自动绘制后的结果，如图 7.10 所示。

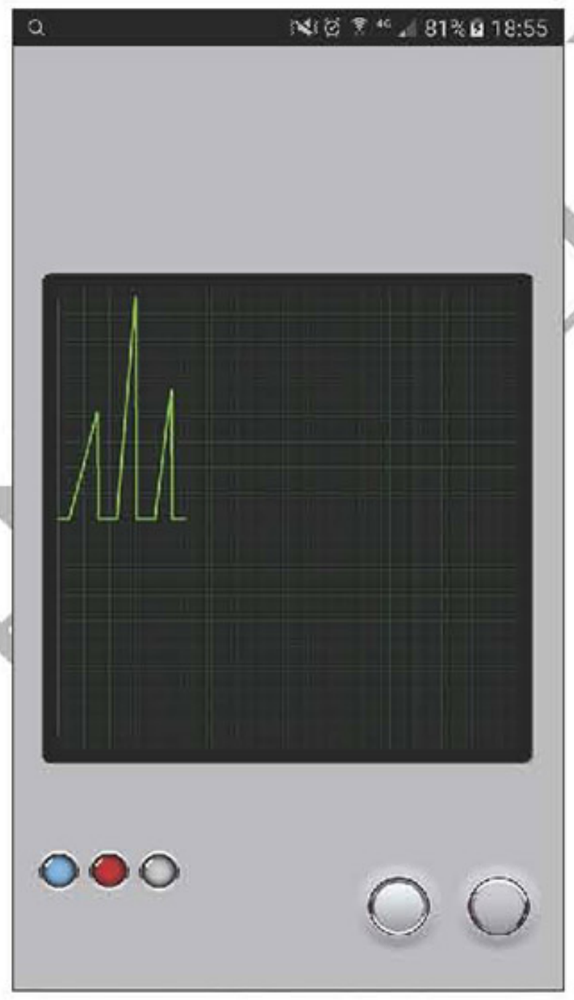


图 7.9　运行效果

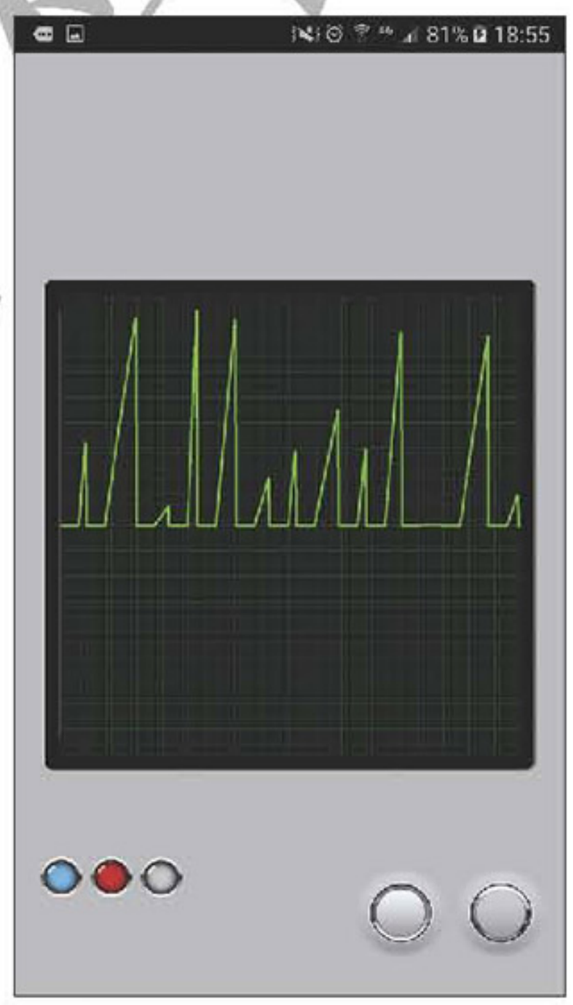


图 7.10　自动绘制后效果

关键技术

关键技术

本实例主要使用 SurfaceHolder 接口类中的 surfaceCreated() 方法、Thread 类中的 Thread() 方法、Canvas 类中的 drawline() 方法、Paint 类中的 setColor() 方法，实现绘制心电图效果。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“XDTView”，公司域名为“mingrisoft.com”。

（2）导入资源文件：复制“光盘\Code\07\063\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）在“app\java\com\mingrisoft\xdtview”文件夹下创建 XDTView 类继承 SurfaceView 实现 SurfaceHolder.Callback，用于自定义心电图控件。代码如下：

```
public class XDTView extends SurfaceView implements SurfaceHolder.Callback
```

（4）实现 SurfaceHolder.Callback 接口类的方法，创建 SurfaceView 控件开启子线程。代码如下：

```
//SurfaceView创建时调用
 @Override
 public void surfaceCreated(SurfaceHolder holder) {
     //TODO Auto-generated method stub
     new Thread(new MyLoop()).start();
 }
```

（5）实现 Runnable 接口完成子线程任务，绘制心电图折线。代码如下：

```
//子线程任务
class MyLoop implements Runnable {
    @Override
    public void run() {
        while (true) { //true执行方法，false停止线程任务
            try {
                //判断什么时候波动
                k ++;
                if(k%20==0){
                //随机获取高度坐标
                y=(int)(Math.random()*mHeight/2);
                //随机增加x坐标
                x+=10+(int)(Math.random()*50);
                }else {
                    //随机坐标为高度的一半
                    y=mHeight/2;
                }
                //添加坐标到集合
                ps.add(new Point(x,y));
                //增加坐标
                x ++;
                //删除以前生成的坐标实现移动
                rovm();
                //获得Canvas对象
                Canvas c = holder.lockCanvas();
```

明日科技

```
26                  //主要绘制方法
27                  doDraw(c);
28                  //绘制完成后更新界面
29                  holder.unlockCanvasAndPost(c);
30                  Thread.sleep(30);
31              } catch (Exception e) {
32              }
33          }
34      }
35  }
```

扩展学习

请尝试添加单击事件，实现重新绘制心电图效果。

实例 064　房价走势折线图

应用场景：统计图

源码位置：光盘\Code\07\064

实例说明

折线统计图可以很清晰地表现出数量的增减变化。本实例主要使用第三方 MPAndroidChart 图表控件，完成房价走势折线图。运行实例，效果如图 7.11 所示；单击显示辅助线以便于观察，如图 7.12 所示；左右放大展示折线图，如图 7.13 所示。

图 7.11　运行效果

图 7.12　辅助线效果

图 7.13　左右放大展示效果

关键技术

本实例主要使用 LineDataSet 类中的 LineDataSet() 方法、setCircleSize() 方法、setCircleColor() 方法，LineData 类中的 LineData() 方法，LineChart 类中的 setData() 方法，以实现通过折线统计图显示数据。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“Linechar”，公司域名为“mingrisoft.com”。

（2）导入资源文件：复制“光盘 \Code\07\064\src\libs”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）在“app\res\layout”目录下默认生成的 activity_main.xml 布局文件中，加入折线图显示控件。代码如下：

```
<!--折线图-->
<com.github.mikephil.charting.charts.LineChart
    android:padding="10dp"
    android:id="@+id/spread_line_chart"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:layout_gravity="center_horizontal"
    />
```

（4）在“app\java\com\mingrisoft\piechar”文件夹下默认生成的 MainActivity 类中，配置折线图显示数据。代码如下：

```
//y轴的数据为随机房价
ArrayList<Entry> yValues = new ArrayList<Entry>();
for (int i = 0; i < count; i++) {
    float value = (float) (Math.random() * range*10) + 6000;
    yValues.add(new Entry(value, i));
}
```

（5）在 MainActivity 类中，配置 X 轴显示数据。代码如下：

```
ArrayList<String> xValues = new ArrayList<String>();
for (int i = 0; i < count; i++) {
    //x轴显示的数据，这里默认使用数字下标显示
    xValues.add(1 + i+"月");
}
```

（6）在 MainActivity 类中，配置折线图属性。代码如下：

```
//用y轴的集合来设置参数
lineDataSet.setLineWidth(1.75f);                    //线宽
lineDataSet.setCircleSize(3f);                      //显示的圆形大小
lineDataSet.setColor(Color.WHITE);                  //显示颜色
```

明日科技

```
05  lineDataSet.setCircleColor(Color.WHITE);                    //圆形的颜色
06  lineDataSet.setHighLightColor(Color.WHITE);                  //高亮线的颜色
07  ArrayList<LineDataSet> lineDataSets = new ArrayList<LineDataSet>();
08  lineDataSets.add(lineDataSet);                               //添加数据
```

扩展学习

在 MainActivity 类中，请尝试修改折线图中的折线颜色。

实例 065　人物能力分析雷达图

应用场景：分析图

源码位置：光盘\Code\07\065

实例说明

雷达图可以将数据集中分布在一个酷似雷达的圆形图表上，用以表现各项数据占比的情况。本实例主要使用第三方 MPAndroidChart 图表控件，完成显示人物能力分析统计图。运行实例，效果如图 7.14 所示；向右滑动，雷达图跟随转动，如图 7.15 所示。

图 7.14　初始效果

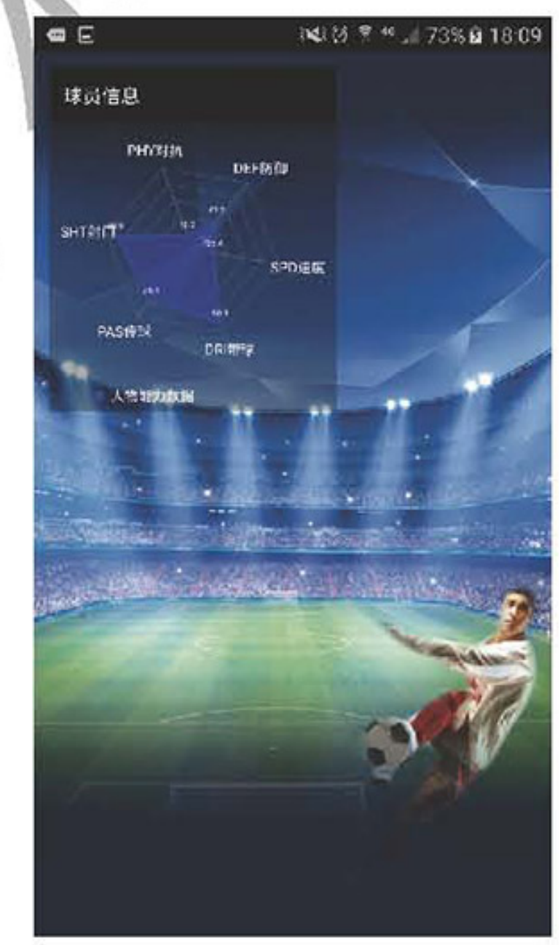


图 7.15　滑动后效果

关键技术

关键技术

本实例主要使用 RadarDataSet 类的 RadarDataSet() 方法、setColor() 方法，RadarData 类的 RadarData() 方法、RadarChart 类的 setData() 方法，实现人物能力分析雷达图显示数据。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“LDRadarChawView”，公司域名为“mingrisoft.com”。

（2）导入资源文件：复制“光盘 \Code\07\065\src\libs”中的 mpandroidchartlibrary-2-1-6.jar 文件，粘贴到“app\libs”目录下，该文件为开源的图表类 jar 包。

（3）在“app\res\layout”目录下默认生成的 activity_main.xml 布局文件中，加入雷达图显示控件。代码如下：

```
<!--雷达图-->
<com.github.mikephil.charting.charts.RadarChart
    android:id="@+id/radar_chart"
    android:layout_width="200dp"
    android:layout_height="200dp"
    android:layout_gravity="center_vertical"
    />
```

（4）在“app\java\com\mingrisoft\sjzlistview”文件夹下默认生成的 MainActivity 类中，配置雷达图显示数据。代码如下：

```
//y轴显示数据
for (int i = 0; i < count; i++) {
    float result = (float) (Math.random() * range) + 3;
    y.add(new Entry(result, i));
}
```

（5）在 MainActivity 类中，配置 x 轴显示数据。代码如下：

```
x.add("PAS传球");
x.add("SHT射门");
x.add("PHY对抗");
x.add("DEF防御");
x.add("SPD速度");
x.add("DRI带球");
```

（6）在 MainActivity 类中，配置雷达图属性。代码如下：

```
//设置样式
public void showChart() {
    radarChart.getYAxis().setEnabled(false);                    //去掉y轴值
    radarChart.getXAxis().setTextColor(Color.WHITE);            //设置x轴文字颜色
    radarChart.setBackgroundColor(getResources().getColor(R.color.colortm)); //背景颜色
    radarChart.setData(radarData);                              //设置数据
    radarChart.setDescription("");                              //设置描述内容
    Legend legend = radarChart.getLegend();                     //设置比例图片标识，显示对应组的value
    legend.setForm(Legend.LegendForm.CIRCLE);                   //样式
```

明日科技

```
10        legend.setFormSize(6f);                    //字号
11        legend.setTextColor(Color.WHITE);          //设置颜色
12        radarChart.animateXY(2000,2000);           //x轴和y轴的动画
13    }
```

扩展学习

请尝试在 MainActivity 类中为数据添加单击事件，实现单击某一能力显示数值的功能。

明日科技

第8章 摄像头

实例 066 调用系统相机获取图像

应用场景：摄像头的应用

源码位置：光盘\Code\08\066

实例说明

本实例将通过调用手机系统的自带拍照功能进行图像的采集，然后将采集后的图像进行裁剪，再将裁剪后的图像显示到界面中。运行实例，单击“添加图片”按钮，会弹出一个提示弹窗，如图 8.1 所示；单击“确定”按钮进行图像采集，图像采集后会显示在界面中，效果如图 8.2 所示。

图 8.1 调用系统相机

图 8.2 显示拍照结果

关键技术

本实例主要使用 Intent 调用系统相机进行拍照然后裁剪，另外，还使用 AlertDialog 弹窗提示用户操作。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“SystemCamera”，公司域名为“mingrisoft. com”。

（2）导入资源文件：复制“光盘\Code\08\066\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）在 MainActivity 中声明控件变量并将布局中的控件实例化。（代码位置：SystemCamera\app\src\main\java\com\mingrisoft\systemcamera\MainActivity.java）。

（4）为 Button 控件设置单击事件，当单击事件触发后弹出弹窗，提示用户是否开启系统拍照功能。在 MainActivity 类中添加 setDialog() 方法，用来设置弹窗的显示效果。代码如下：

```
01  /**
02   * 创建提示弹窗
03   */
04  private void setDialog() {
05      //创建提示窗
06      AlertDialog.Builder alertDialog = new AlertDialog.Builder(this);
07      alertDialog.setTitle("提示");                                    //设置弹窗的标题
08      alertDialog.setMessage("是否使用系统相机进行拍照？");              //设置弹窗的内容
09      alertDialog.setPositiveButton("确定",                            //设置确定按钮的单击事件
10              (dialog, which) -> {
11                  Toast.makeText(this, "单击了确认", Toast.LENGTH_SHORT).show();
12                  getPictureFromCamera();                              //打开系统摄像头拍照
13              });
14      alertDialog.setNeutralButton("取消",                             //设置取消按钮的单击事件
15              (dialog, which) -> Toast.makeText(this, "点击了取消", Toast.LENGTH_SHORT).show());
16      alertDialog.create();                                            //创建弹窗
17      alertDialog.show();                                              //显示弹窗
18  }
```

在 MainActivity 类中添加 getPictureFromCamera() 方法，用来实现跳转到系统拍照界面。代码如下：

```
01  /**
02   * 调用系统摄像头拍照
03   */
04  private void getPictureFromCamera() {
05      //判断存储卡是否可用，如果不可用则跳出方法并提示
06      if (!hasSdcard()) {
07          Toast.makeText(this, "储存卡不可用", Toast.LENGTH_SHORT).show();
08          return;
```

```
09      }
10      //如果储存卡可用，则使用下面的方法调用系统摄像头设备
11      Intent intent = new Intent("android.media.action.IMAGE_CAPTURE");     //调用系统摄像头的意图
12      //设置图片保存的位置
13      file = new File(Environment.getExternalStorageDirectory(), "my_image");
14      intent.putExtra(MediaStore.EXTRA_OUTPUT, Uri.fromFile(file));         //向意图中传值
15      startActivityForResult(intent, 1000); //调用带有返回值的跳转界面的方法，请求值设为1000
16  }
```

在 MainActivity 类中重写 onActivityResult() 方法，在该方法中获取调用系统相机拍照的结果并调用系统自带的裁剪功能。

在 MainActivity 类中添加 crop() 方法，在该方法中设置裁剪的属性。代码如下：

```
01  /**
02   * 调用系统自带的裁剪图片功能
03   * @param uri
04   */
05  private void crop(Uri uri) {
06      Log.e("URI", uri.getPath());
07      Log.e("URI", uri.toString());
08      //裁剪图片意图
09      Intent intent = new Intent("com.android.camera.action.CROP");
10      intent.setDataAndType(uri, "image/*");
11      intent.putExtra("crop", "true");
12      //裁剪框的比例，1：1
13      intent.putExtra("aspectX", 1);
14      intent.putExtra("aspectY", 1);
15      //裁剪后输出图片的尺寸大小
16      intent.putExtra("outputX", 150);
17      intent.putExtra("outputY", 150);
18      intent.putExtra("scale", true);                                  //黑边
19      intent.putExtra(MediaStore.EXTRA_OUTPUT, Uri.fromFile(file));
20      //图片格式
21      intent.putExtra("outputFormat", Bitmap.CompressFormat.JPEG.toString());
22      intent.putExtra("noFaceDetection", true);                        //取消人脸识别
23      intent.putExtra("return-data", true);                            //true:不返回uri，false：返回uri
24      startActivityForResult(intent, 2000);
25  }
```

扩展学习

请尝试给添加图片的按钮更改一个样式，实现在单击按钮的时候改变字体的颜色。

实例 067　Camera 与 SurfaceView

应用场景：自定义相机

源码位置：光盘\Code\08\067

实例说明

一般情况下，系统自带相机的风格会与开发者的 App 应用风格不统一，所以会采取自定义相机这种方式来达到与 App 应用的风格统一的效果。本实例将使用 Camera 与 SurfaceView 实现一个简易相机的功能。运行实例，将显示如图 8.3 所示的自定义相机的主界面；当拍照结束后，将显示当前所拍摄的图像，如图 8.4 所示。

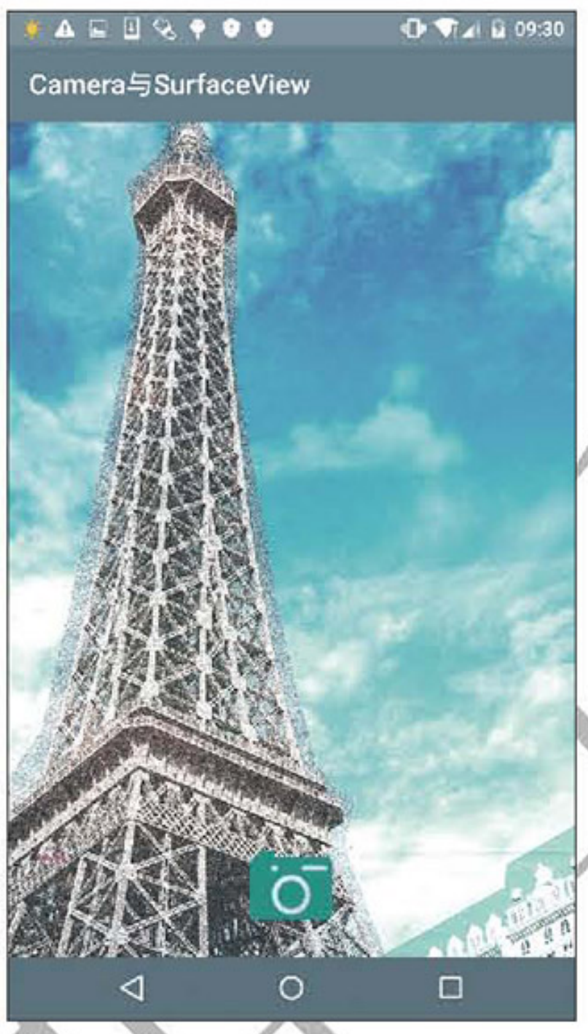


图 8.3　拍照界面

图 8.4　显示拍照结果

关键技术

本实例主要使用 SurfaceView 绘制摄像头拍摄到的图像，使用 BitmapFactory 将图像字节数据转为 Bitmap 对象。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“CustomCamera1”，公司域名为“mingrisoft. com”。

（2）导入资源文件：复制“光盘\Code\08\067\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）在 MainActivity 类中设置获取 Camera 对象的私有方法。其中，cameraID 代表着要打开的摄像头的 ID，在这里默认为后置摄像头。代码如下：

代码位置：CustomCamera1\app\src\main\java\com\mingrisoft\customcamera1\MainActivity.java

```
01  private Camera getCameraInstance() {
02      Camera camera = null;
03      try {
04          camera = Camera.open(cameraID);                    //获取摄像头对象
05      } catch (Exception e) {
06      }
07      return camera;
08  }
```

为 MainActivity 类添加 SurfaceHolder.Callback 接口，并实现 SurfaceHolder.Callback 接口的 surfaceCreated() 方法、surfaceChanged() 方法和 surfaceDestroyed() 方法。

在 MainActivity 的 surfaceCreated() 方法中，获取 Camera 对象，开启预览界面。代码如下：

```
01  mCamera = getCameraInstance();
02  setCameraDisplayOrientation(this, cameraID, mCamera);
03  mCamera.setPreviewDisplay(mHolder);                        //使holder与预览相关联
04  mCamera.startPreview();                                    //开启预览
```

声明 Camera.PictureCallback 类型的对象，并实现 Camera.PictureCallback 类型的匿名内部类，在匿名内部类中设置拍照功能，并将拍照结果写入到手机外部存储器中。代码如下：

```
01  File pictureFile = new File(Environment.getExternalStorageDirectory(),
02          System.currentTimeMillis() + "CS_image.jpg");
03  Bitmap bitmap = BitmapFactory.decodeByteArray(data, 0, data.length);     //获取bitmap对象
04  Matrix matrix = new Matrix();                              //获取矩阵对象用于图形变换
05  matrix.setRotate(degrees);                                 //设置旋转角度
06  bitmap = Bitmap.createBitmap(bitmap, 0, 0,
07          bitmap.getWidth(), bitmap.getHeight(), matrix, true);
08  BufferedOutputStream buff = null;                          //使用输出流，将文件写入SD卡上
09  try {
10      buff = new BufferedOutputStream(new FileOutputStream(pictureFile));  //获取缓存流
11      bitmap.compress(Bitmap.CompressFormat.JPEG, 100, buff);              //将文件保存到指定路径
12      Message message = Message.obtain();
13      message.what = 1;                                      //设置标记值
14      message.obj = pictureFile.getPath();                   //获取图片路径
15      handler.sendMessage(message);                          //发送一条消息
16  } catch (FileNotFoundException e) {
17      Log.d(TAG, "文件未被找到" + e.getMessage());
18  } finally {
19      if (buff != null) {
20          try {
21              buff.close();                                  //关闭流
22          } catch (IOException e) {
23              e.printStackTrace();
```

```
24            }
25        }
26    }
```

声明 Handler 类型的对象，同时实现 Handler 的匿名内部类。在该匿名内部类中接受消息，随后将接收到的消息传递给要跳转到的界面。代码如下：

```
01  private Handler handler = new Handler(msg -> {
02      if (msg.what == 1) {
03          mCamera.startPreview();
04          dialog.dismiss();                                    //取消进度条
05          String imagePath = (String) msg.obj;                 //获取图片的保存路径
06          Intent intent = new Intent(this, ShowPictureActivity.class);
07          startActivity(intent.putExtra("image_path", imagePath));   //跳转界面
08      }
09      return false;
10  });
```

（4）在 ShowPictureActivity 界面中接收从 MainActivity 界面传递过来的 intent，通过 intent 获取图片的保存路径，随后将图片和图片路径展示出来。代码如下：

```
01  Intent intent = getIntent();
02  imageView = (ImageView) findViewById(R.id.show_image);
03  textView = (TextView) findViewById(R.id.txt);                //实例化控件对象
04  if (intent != null) {
05      String path = intent.getStringExtra("image_path");
06      imageView.setImageURI(Uri.parse(path));
07      textView.setText("保存路径: " + path);
08  }
```

扩展学习

本实例简单实现了 Camera 拍照的功能，请尝试为本实例增加开启和关闭闪光灯的功能。

实例 068　Camera2 与 TextureView

应用场景：自定义相机

源码位置：光盘\Code\08\068

实例说明

Camera2 是 Camera 之后的一款特效相机，可以拍摄更高质量的照片，并且可以实现 LOMOS、复古、艺术和科幻等特效。本实例将使用 Camera2 与 TextureView 实现一个简单的自定义相机。运行实例，将显示如图 8.5 所示的拍照界面；单击拍照后，将图片存储在指定位置，并弹出图片的保存路径，如图 8.6 所示。

明日科技

图 8.5　拍照界面

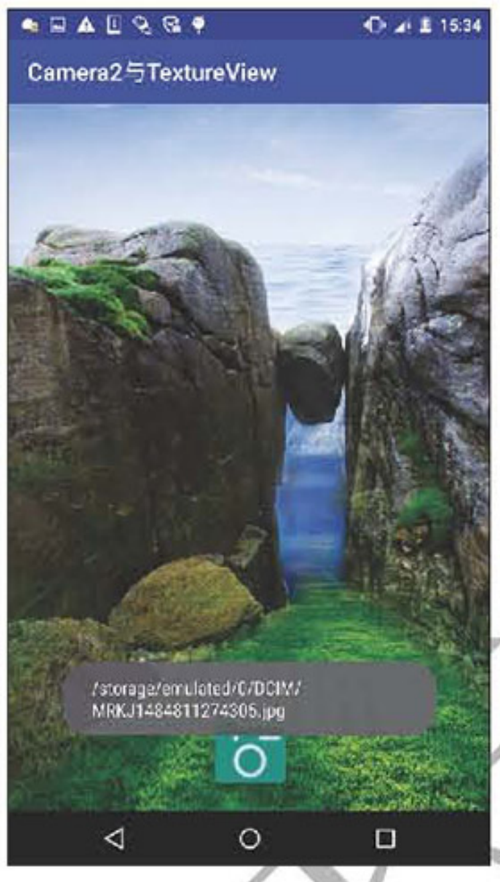


图 8.6　提示图片的保存路径

关键技术

关键技术

本实例主要使用 TextureView 类显示摄像头拍摄到的图像，使用 CameraManager 获取相机服务，使用 getCameraCharacteristics() 方法获取摄像头的特性，使用 openCamera() 方法开启摄像头，使用 CameraDevice 类的 createCaptureSession() 方法创建会话，用于预览图像。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“CustomCamera2”，公司域名为“mingrisoft. com”。

（2）导入资源文件：复制“光盘\Code\08\068\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）在 CustomCamera2 项目中的“app\src\main\java\com\mingrisoft\customcamera2”目录下创建 listener 包，该包用于保存本项目中所使用到的工具类文件，该包中包括 5 个类文件，其类名及作用如表 8.1 所示（代码位置：CustomCamera2\app\src\main\java\com\mingrisoft\customcamera2\activity\MainActivity.java）。

表 8.1　listener 包中的 5 个类

类　名	描　述
MineSessionListener	用于接收关于照相机捕获会话的状态更新的回调对象
MineImageAvailableListener	用来获取照片的数据
MineSessionListener	用于接收关于照相机捕获会话的状态更新的回调对象
MineStateCallback	返回摄像头的状态
MineTextureListener	用于显示图像的预览

（4）在 MainActivity 中添加 openCamera() 方法。该方法用来是在界面运行的时候开启摄像头，注意这个方法是在 Activity 生命周期中的 onResume() 方法中调用的。代码如下：

```
/**
 * 打开相机
 */
@TargetApi(Build.VERSION_CODES.LOLLIPOP)
private void openCamera() {
    Toast.makeText(MainActivity.this, "打开相机", Toast.LENGTH_SHORT).show();
    //通过CameraManager获取相机服务
    CameraManager manager = (CameraManager) getSystemService(Context.CAMERA_SERVICE);
    try {
        String cameraId = manager.getCameraIdList()[0];          //返回相机列表
        //获取指定相机的特性
        CameraCharacteristics characteristics = manager.getCameraCharacteristics(cameraId);
        //获取摄像头支持的流配置-->涉及图像的所有分辨率以及每一帧的信息
        StreamConfigurationMap map = characteristics.get(CameraCharacteristics
                .SCALER_STREAM_CONFIGURATION_MAP);
        //获取最大的输出尺寸
        previewSize = map.getOutputSizes(SurfaceTexture.class)[0];
        //权限判断
        if (ActivityCompat.checkSelfPermission(this, Manifest.permission.CAMERA)
                != PackageManager.PERMISSION_GRANTED) {
            return;
        }
        //打开指定的相机
        manager.openCamera(cameraId, stateCallback, null);
        //创建ImageReader对象
        imageReader = ImageReader.newInstance(previewSize.getWidth(),preview.getHeight(),
                ImageFormat.JPEG,1);
    } catch (CameraAccessException e) {
        e.printStackTrace();
    }
}
```

在 MainActivity 中添加 startPreview() 方法，用于开启相机或开启预览界面，将摄像头拍摄到的图像显示在 TextureView 上。代码如下：

```
/**
 * 开启预览
 */
@TargetApi(Build.VERSION_CODES.LOLLIPOP)
private void startPreview() {
    if (cameraDevice == null){
```

明日科技

```
            Log.e("startPreview", "cameraDevice is null");
            return;
        }
        //使用TextureView预览相机是通过使用SurfaceTexture实现的
        //与SurfaceView中使用SurfaceHolder类似
        SurfaceTexture texture = preview.getSurfaceTexture();
        if (texture == null){
            Log.e("startPreview", "surface is null");
            return;
        }

        //设置默认缓冲区的尺寸和相机所支持的最大尺寸
        texture.setDefaultBufferSize(previewSize.getWidth(), previewSize.getHeight());
        //开启预览所需要的输出Surface
        Surface surface = new Surface(texture);
        try {
            //创建预览请求
            requestBuilder = cameraDevice.createCaptureRequest(CameraDevice.TEMPLATE_PREVIEW);
            requestBuilder.addTarget(surface);
            //创建预览会话
            cameraDevice.createCaptureSession(Arrays.asList(surface),
                    new MineSessionListener() {
                @Override
                public void onSessionConfigured(CameraCaptureSession session) {

                    //获取预览会话
                    captureSession = session;
                    //配置预览时使用3A模式(自动曝光、自动对焦、自动白平衡)
                    requestBuilder.set(CaptureRequest.CONTROL_MODE,
                            CameraMetadata.CONTROL_MODE_AUTO);
                    //创建用于预览界面的线程
                    previewThread = new HandlerThread("preview");
                    previewThread.start();
                    previewHandler = new Handler(previewThread.getLooper());
                    try {
                        //设置重复捕获图像
                        captureSession.setRepeatingRequest(requestBuilder.build(),
                                null, previewHandler);
                    } catch (CameraAccessException e) {
                        e.printStackTrace();
                    }
                }
```

明日科技

```
49              },null);
50          } catch (CameraAccessException e) {
51              e.printStackTrace();
52          }
53      }
```

在 MainActivity 类中添加 takePictureAndSave() 方法，实现拍照功能，并将拍照结果写入到手机外部存储器中，由于篇幅有限，请参考实例源码进行学习。

扩展学习

查询 CameraManager 类的 API 文档，尝试为本实例增加自动开启闪光灯的功能。

实例 069　为摄像机增加滤镜效果

应用场景：滤镜效果

源码位置：光盘\Code\08\069

实例说明

本实例将实现为摄像的预览界面添加滤镜的效果。运行实例，单击选择滤镜按钮，将显示选择如图 8.7 所示的弹窗，在这个弹窗中单击“复古”图标后，将显示如图 8.8 所示的预览界面。

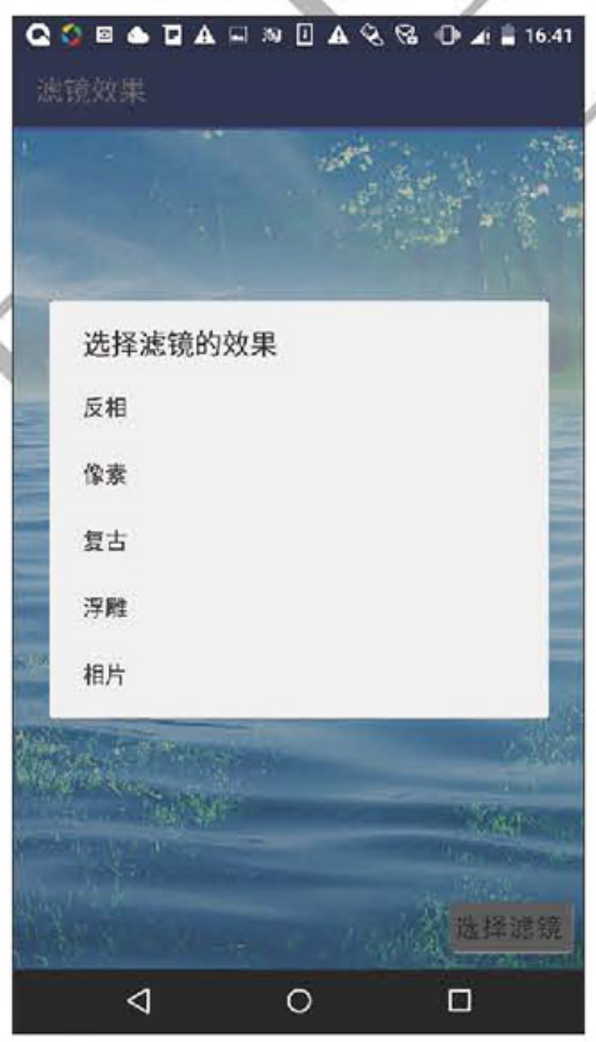


图 8.7　拍照界面

图 8.8　设置滤镜的效果

关键技术

本实例主要使用开源框架 GPUImage（项目中只需要导入：compile 'jp.co.cyberagent.android.gpuimage:gpuimage-library:1.4.1'）实现添加滤镜效果。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“GPUImageDemo”，公司域名为“mingrisoft. com”。

（2）导入资源文件：复制“光盘\Code\08\069\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）创建 SelectFilter 类，并在该类中添加 showDialog() 方法，用于创建选择滤镜的弹窗。关键代码如下：

代码位置：GPUImageDemo\app\src\main\java\com\mingrisoft\gpuimagedemo\SelectFilter.java

```
/**
 * 显示弹窗
 * @param context 上下文对象用于创建弹窗
 * @param listener 用于返回设置的滤镜类型
 */
public static void showDialog(final Context context,
                              final OnFilterChooseListener listener) {
    final FilterList filters = new FilterList();
    filters.addFilter("反相", FilterType.反相);                    //添加滤镜
    filters.addFilter("像素", FilterType.像素);                    //添加滤镜
    filters.addFilter("复古", FilterType.复古);                    //添加滤镜
    filters.addFilter("浮雕", FilterType.浮雕);                    //添加滤镜
    filters.addFilter("相片", FilterType.相片);                    //添加滤镜
    AlertDialog.Builder builder = new AlertDialog.Builder(context);
    builder.setTitle("选择滤镜的效果");                             //设置标题
    //设置Item单击事件
    builder.setItems(filters.names.toArray(new String[filters.names.size()]),
            (dialog, which) ->  listener.OnFilterChooseListener(
                    createFilterForType(filters.filters.get(which))));
    builder.create().show();                                       //显示弹窗
}
```

（4）在 MainActivity 类中初始化控件，并实例化 GPUImage 对象。关键代码如下：

代码位置：GPUImageDemo\ app\src\main\java\com\mingrisoft\gpuimagedemo\MainActivity.java

```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
    //获取GPUImage对象
    mGPUImage = new GPUImage(this);
```

```
07      //设置预览
08      mGPUImage.setGLSurfaceView((GLSurfaceView) findViewById(R.id.surfaceView));
09      //设置单击事件，用于打开具有滤镜信息的滤镜列表
10      findViewById(R.id.button_choose_filter).setOnClickListener(
11              v -> SelectFilter.showDialog(MainActivity.this,
12                      filter -> switchFilterTo(filter)));
13  }
```

（5）在 MainActivity 类中添加 switchFilterTo() 方法，用来选择弹窗中使用的滤镜效果，并且更改预览界面的显示效果。关键代码如下：

```
01  /**
02   * 设置滤镜
03   *
04   * @param filter 用于设置滤镜的效果
05   */
06  private void switchFilterTo(final GPUImageFilter filter) {
07      if (mFilter == null
08              || (filter != null && !mFilter.getClass().equals(filter.getClass()))) {
09          mFilter = filter;                          //获取滤镜对象
10          mGPUImage.setFilter(mFilter);              //设置选择的滤镜
11      }
12  }
```

扩展学习

请尝试在滤镜的选择弹窗中增加一个选择黑白滤镜的效果。

实例 070　扫描与生成二维码

应用场景：二维码扫描

源码位置：光盘\Code\08\070

实例说明

目前，二维码的使用越来越广泛，用途也越来越多。本实例将使 Zxing 精简包实现二维码的生成与扫描。运行实例，进入生成二维码页面，如图 8.9 所示；扫描已生成的二维码，如图 8.10 所示。

关键技术

本实例主要使用 Zxing 类库（Zxing 开源项目网址为“https//github.com/zxing/zxing”）中的 CaptureActivity 来扫描二维码；使用 EncodingUthls 类的 creataQRCode() 方法来生成二维码。

图 8.9　生成二维码

图 8.10　扫描二维码

实现过程

（1）打开 Android Studio，新建一个项目，命名为“QRCode”，公司域名为“mingrisoft. com”。

（2）导入资源文件：复制“光盘\Code\08\070\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）开启二维码扫描方法的关键代码如下：

代码位置：QRCode\app\src\main\java\com\mingrisoft\qrcode\MainActivity.java

```
01  startActivityForResult(new Intent(this, CaptureActivity.class),SCANCODE);
```

（4）在 MainActivity 类中重写 onActivityResult() 方法，在该方法中对扫描结果进行处理，这里是把返回结果展示到 TextView 上。代码如下：

```
01  switch (requestCode) {
02      case SCANCODE:                              //接收返回的二维码信息
03          scanMessage.setText(data.getStringExtra("result"));
04          break;
05      default:
06          break;
07  }
```

（5）在 MainActivity 类中添加 enCode() 方法，用来生成一个带 Logo 的二维码。代码如下：

```
01  /**
02   * 生成二维码的单击事件
03   * @param view
04   */
```

```
05  public void enCode (View view){
06      //获取Logo资源
07      Bitmap logoBitmap = BitmapFactory.decodeResource(getResources(),R.mipmap.mrkj);
08      //生成二维码
09      BitmapcodeBitmap = EncodingUtils
10              .createQRCode("http://www.mingrisoft.com/",500,500,logoBitmap);
11      enCodeImage.setImageBitmap(codeBitmap);        //显示二维码
12  }
```

扩展学习

请尝试为显示带 Logo 的二维码控件增加选择 Logo 图片的功能。

实例 071	人脸识别 应用场景：人脸识别	源码位置：光盘\Code\08\071

实例说明

本实例主要使用科大讯飞的人脸识别 SDK，实现实时人脸识别功能。即在预览图像时实时观看人脸在图像中的位置，并将人脸的位置用绿框标记。运行实例，打开摄像头，用绿框标记出人脸的位置，再用绿点标记眉毛、眼睛、鼻子和嘴的位置，效果如图 8.11 与图 8.12 所示。

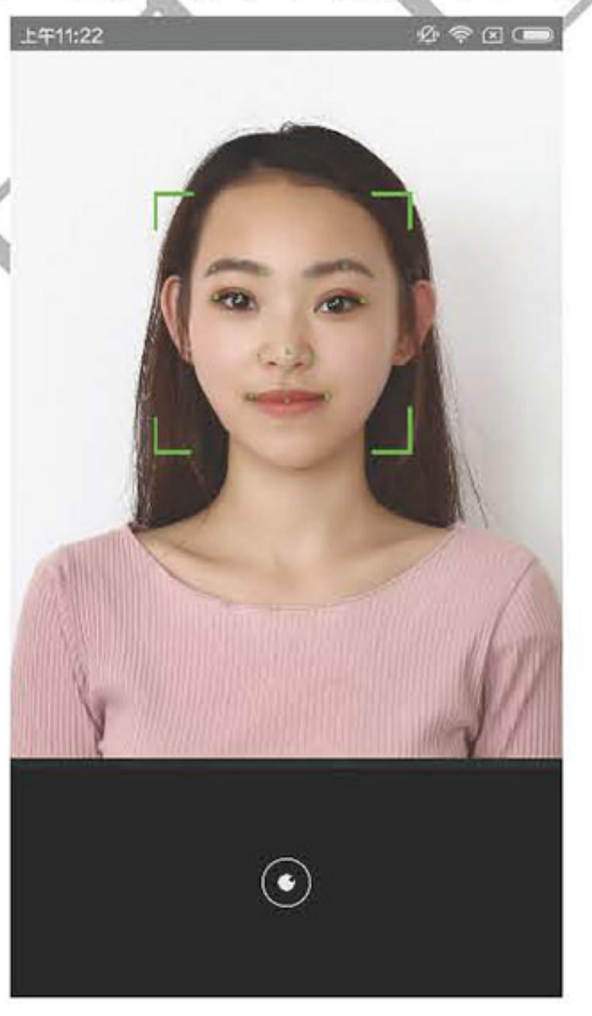


图 8.11　扫描人脸效果（一）

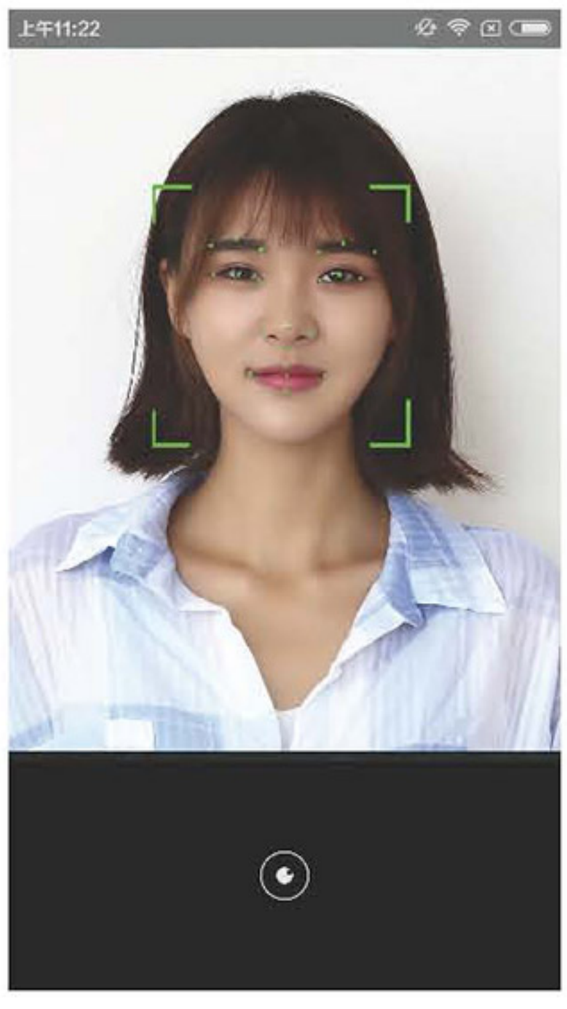


图 8.12　扫描人脸效果（二）

关键技术

本实例主要使用 Canvas 绘制图形、JSON 数据解析、调用手机摄像头和科大讯飞人脸识别的 SDK 进行人脸识别。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“FaceDetect”，公司域名为“mingrisoft. com”。

（2）导入资源文件：复制“光盘\Code\08\071\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）在科大讯飞官网下载人脸识别 SDK，随后将 SDK 导入到项目中。科大讯飞官网为 http://www.xfyun.cn/。人脸识别 SDK 下载的方式很简单，官网有相关的操作说明。

（4）参考源码在 FaceDetect 项目的“app\src\main\java\com\mingrisoft\facedetect”目录中，创建三个类，其类名及作用如表 8.2 所示。

表 8.2　facedetect 包中的三个类

类　　名	描　　述
DrawFaceRect	绘制人脸检测框
FaceRect	实体类，用于保存人脸识别的数据
Result	解析人脸返回结果

（5）在 MainActivity 类中创建 openCamera() 方法，并在该方法中设置摄像头预览的尺寸和预览图像的格式（代码位置：FaceDetect\app\src\main\java\com\mingrisoft\facedetect\MainActivity.java）。关键代码如下：

```
01  //获取摄像头参数对象
02  Camera.Parameters params = mCamera.getParameters();
03  //设置预览的格式
04  params.setPreviewFormat(ImageFormat.NV21);
05  //设置预览的分辨率，这里设置为640×480，到目前为止只支持该分辨率的人脸检测
06  params.setPreviewSize(PREVIEW_WIDTH, PREVIEW_HEIGHT);
07  //给摄像头设置配置参数
08  mCamera.setParameters(params);
```

在 MainActivity 类的 onResume() 方法中设置人脸检测，并获取人脸检测的返回结果。代码如下：

```
01  String result = mFaceDetector.trackNV21(
02          buffer, PREVIEW_WIDTH, PREVIEW_HEIGHT, 1, direction);     //获取人脸检测结果
03  FaceRect face = Result.result(result);                            //获取返回的数据
```

（6）在 Result 类中添加 result () 方法，用来解析人脸检测返回的 JSON 数据。代码如下：

```
01  /**
02   * 解析人脸返回结果
03   * @param json
04   * @return
```

```
 */
public static FaceRect result(String json) {
    FaceRect rect = null;
    if (TextUtils.isEmpty(json)) {
        return null;
    }
    try {
        JSONObject joResult = new JSONObject(json);
        JSONArray items = joResult.getJSONArray("face");  //解析face数组
        JSONObject position = items.getJSONObject(0).getJSONObject("position");
        //获取人脸检测框数据
        rect = new FaceRect();
        rect.bound.left = position.getInt("left");          //左边起始位置点
        rect.bound.top = position.getInt("top");            //上端起始位置点
        rect.bound.right = position.getInt("right");        //右边结束位置点
        rect.bound.bottom = position.getInt("bottom");      //下端结束位置点
        //提取关键点数据
        JSONObject landmark = items.getJSONObject(0).getJSONObject("landmark");
        int keyPoint = landmark.length();                   //人脸检测点数据
        rect.point = new Point[keyPoint];                   //人脸检测点数组
        Iterator it = landmark.keys();                      //使用迭代器解析人脸监测点数据
        int point = 0;
        while (it.hasNext()) {                              //检测点的坐标
            String key = (String) it.next();
            JSONObject object = landmark.getJSONObject(key);
            rect.point[point] = new Point(object.getInt("x"), object.getInt("y"));
            point++;
        }
    } catch (JSONException e) {
        e.printStackTrace();
    }
    return rect;
}
```

（7）在 DrawFaceRect 类中添加 drawFaceRect() 方法，用来根据解析出的数据绘制人脸识别框。代码如下：

```
//绘制人脸识别框，每两个一组
canvas.drawLine(drawL, drawD, drawL, drawD - len, paint);
canvas.drawLine(drawL, drawD, drawL + len, drawD, paint);
canvas.drawLine(drawR, drawD, drawR, drawD - len, paint);
canvas.drawLine(drawR, drawD, drawR - len, drawD, paint);
canvas.drawLine(drawL, drawU, drawL, drawU + len, paint);
canvas.drawLine(drawL, drawU, drawL + len, drawU, paint);
canvas.drawLine(drawR, drawU, drawR, drawU + len, paint);
canvas.drawLine(drawR, drawU, drawR - len, drawU, paint);
/**
```

```
11    * 绘制人脸监测点
12    */
13  if (face.point != null) {
14      //遍历检测点并绘制
15      for (Point p : face.point) {
16          if (frontCamera) {
17              p.y = width - p.y;
18          }
19          canvas.drawPoint(p.x, p.y, paint);
20      }
21  }
```

扩展学习

本实例绘制的人脸识别框为矩形状态，请尝试将人脸识别框绘制成圆形或者绘制图片来覆盖人脸。

实例 072 识别银行卡信息

应用场景：图像识别

源码位置：光盘\Code\08\072

实例说明

本实例将通过一个开源的第三方库 Card.io，实现银行卡扫描获取卡号功能。运行实例，将显示如图 8.13 所示的界面；当获取到银行卡卡号后，会将卡号显示出来，如图 8.14 所示。

图 8.13 未扫描前

图 8.14 扫描结果

关键技术

本实例主要使用开源库 Card.io 所提供的 SDK（项目地址：https://github.com/card-io/card.io-Android-SDK）。其中，在调用 Card.io 库扫描结束后会将扫描结果传给调用 Card.io 库的界面，然后通过 Intent 获取到 CreditCard 对象，最后通过 CreditCard 类的 getFormattedCardNumber() 方法将卡号获取出来。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“ID Card”，公司域名为“mingrisoft. com”。

（2）向项目中添加依赖“compile 'io.card:android-sdk:5.5.1'”（文件位置：ID Card\app\build.gradle）。

（3）导入资源文件：复制“光盘\Code\08\072\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（4）在 MainActivity 类的 onCreate() 方法中初始化控件，并设置按钮的单击事件，当单击事件被触发后调用 Card.io 库提供的扫描界面进行图像采集。代码如下：

代码位置：ID Card\\app\src\main\java\com\mingrisoft\idcard\MainActivity.java

```
01  private TextView txt;                                          //显示银行卡的卡号
02  private ImageView img;                                         //显示Logo
03  private int MY_SCAN_REQUEST_CODE = 100;                        //请求码
04
05  @Override
06  public void onCreate(Bundle savedInstanceState) {
07      super.onCreate(savedInstanceState);
08      setContentView(R.layout.activity_main);
09      //初始化控件
10      txt = (TextView) findViewById(R.id.txt);
11      img = (ImageView) findViewById(R.id.img);
12      txt.setText("用来显示扫描结果");
13  }
14
15  /**
16   * 按钮的单击事件
17   *
18   */
19  public void onScanPress() {
20      Intent scanIntent = new Intent(this, CardIOActivity.class);          //创建意图
21      scanIntent.putExtra(CardIOActivity.EXTRA_SUPPRESS_MANUAL_ENTRY, true); //关闭键盘显示
22      startActivityForResult(scanIntent, MY_SCAN_REQUEST_CODE);            //跳转界面
```

重写 Activity 的 onActivityResult() 方法，在这个方法中将获取到的返回结果显示在控件中。代码如下：

```
01  @Override
02  protected void onActivityResult(int requestCode, int resultCode, Intent data) {
```

明日科技

```
03      super.onActivityResult(requestCode, resultCode, data);
04      String resultStr; //用来保存获取的结果
05      if (data != null && data.hasExtra(CardIOActivity.EXTRA_SCAN_RESULT)) {
06          //获取返回内容
07          CreditCard scanResult = data.getParcelableExtra(CardIOActivity.EXTRA_SCAN_RESULT);
08          resultStr = "扫描卡号：" + scanResult.getFormattedCardNumber(); //获取银行卡卡号
09      } else {
10          resultStr = "扫描已取消";
11      }
12      txt.setText(resultStr);
13  }
```

扩展学习

Card.io 库不仅可以检测出银行卡号，也可以检测出银行卡的 Logo。请尝试将银行卡的 Logo 显示在界面中。

实例 073　相机挂件

应用场景：娱乐相机

源码位置：光盘\Code\08\073

实例说明

很多相机在自拍的时候，都可以通过相机对人脸进行面部的五官识别，并在人的五官上添加挂件。运行实例，单击帽子，在人脸的右上方添加帽子挂件，如图 8.15 所示；单击眼镜框，在人脸的眼睛位置添加眼镜挂件，如图 8.16 所示；单击猫耳朵和皇冠，在人脸的正上方显示皇冠，如图 8.17 所示。

图 8.15　挂件（一）

图 8.16　挂件（二）

图 8.17　挂件（三）

关键技术

本实例主要使用 Canvas 类中的 drawBitmap() 方法、BitmapFactory 类中的 decode Resource() 方法和 List 类中的 add() 方法，实现获取相应的挂件并绘制到指定位置。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“PendantCamera”，公司域名为“mingrisoft.com”。

（2）导入资源文件：复制“光盘 \ Code\08\073\src”文件夹中的图片资源，粘贴到“app\res”文件夹中的 mipmap 文件夹。

（3）将 Msc.jar 包导入到“app\libs”文件夹中，该 Jar 包用于脸部识别。

（4）在 MainActivity 类中的 onStart() 方法中加载挂件的图片，关键代码如下：

```
/**
 * 加载bitmap图片
 */
@Override
protected void onStart() {
    super.onStart();
    bitmapList = new ArrayList<>();    //初始化加载图片的集合
    Bitmap a = BitmapFactory.decodeResource(getResources(), R.mipmap.note);
    Bitmap b = BitmapFactory.decodeResource(getResources(), R.mipmap.lefteye);
    Bitmap c = BitmapFactory.decodeResource(getResources(), R.mipmap.righteye);
    Bitmap d = BitmapFactory.decodeResource(getResources(), R.mipmap.cattop);
    Bitmap e = BitmapFactory.decodeResource(getResources(), R.mipmap.catearleft);
    Bitmap f = BitmapFactory.decodeResource(getResources(), R.mipmap.catearright);
    Bitmap g = BitmapFactory.decodeResource(getResources(),R.mipmap.hat);
    Bitmap h = BitmapFactory.decodeResource(getResources(),R.mipmap.beard);
    Bitmap j = BitmapFactory.decodeResource(getResources(),R.mipmap.glasses);
    //将图片添加到指定的位置
    bitmapList.add(0, a);
    bitmapList.add(1, b);
    bitmapList.add(2, c);
    bitmapList.add(3, d);
    bitmapList.add(4, e);
    bitmapList.add(5, f);
    bitmapList.add(6,g);
    bitmapList.add(7,h);
    bitmapList.add(8,j);
}
```

（5）在 MainActivity 类中，当单击猫耳朵与皇冠的按钮时，将传递的参数类型及图片集合等传给画布，进行图片控件的绘制，关键代码如下：

明日科技

```
case R.id.cat_image:                                          //单击猫耳朵与皇冠的按钮
if (typeBtn != 2) {
    typeBtn = 2;
    //绘制人脸检测的区域
    //将相关的参数传递给绘制挂件的类
    DrawFaceRect.drawFaceRect(typeBtn, bitmapList, canvas, faceTwo, PREVIEW_WIDTH, frontCamera);
}
break;
```

（6）在 DrawFaceRect 类中，将 MainActivity 类传递来的参数进行接收，并将必要的参数显示出来。然后设置两个猫耳朵与皇冠的位置及大小，设置完成后，用画布的 drawBitmap() 方法绘制并显示在手机界面上，关键代码如下：

```
/**
 * 设置左边耳朵矩形的大小
 **/
dst.left = (drawL + drawR) / 2 - 150;
dst.top = drawU - 100;
dst.right = (drawL + drawR) / 2 + 150;
dst.bottom = drawU + 100;
/**
 * 设置右边耳朵矩形的大小
 **/
dst1.left = drawL;
dst1.top = drawU - 100;
dst1.right = drawL + 250;
dst1.bottom = drawU + 100;
/**
 * 设置皇冠矩形的大小
 **/
dst2.left = drawR - 250;
dst2.top = drawU - 100;
dst2.right = drawR;
dst2.bottom = drawU + 100;
canvas.drawBitmap(bitmapList.get(3), null, dst, paintBitmap);     //绘制皇冠
canvas.drawBitmap(bitmapList.get(4), null, dst1, paintBitmap);    //绘制左耳
canvas.drawBitmap(bitmapList.get(5), null, dst2, paintBitmap);    //绘制右耳
```

扩展学习

请尝试使用动态方式来显示人脸区域。

明日科技

第9章 动画

实例 074 飘落的雪花

应用场景：动画效果

源码位置：光盘\Code\09\074

实例说明

在 Andriod 系统中，SurfaceView 视图拥有独立的绘图表面，因此可以实现复杂而高效的界面效果。本实例将使用 SurfaceView 来实现一个雪花飘落的动画效果。运行实例，将显示如图 9.1 所示的界面，当屏幕图发生变化时，界面效果如图 9.2 所示。

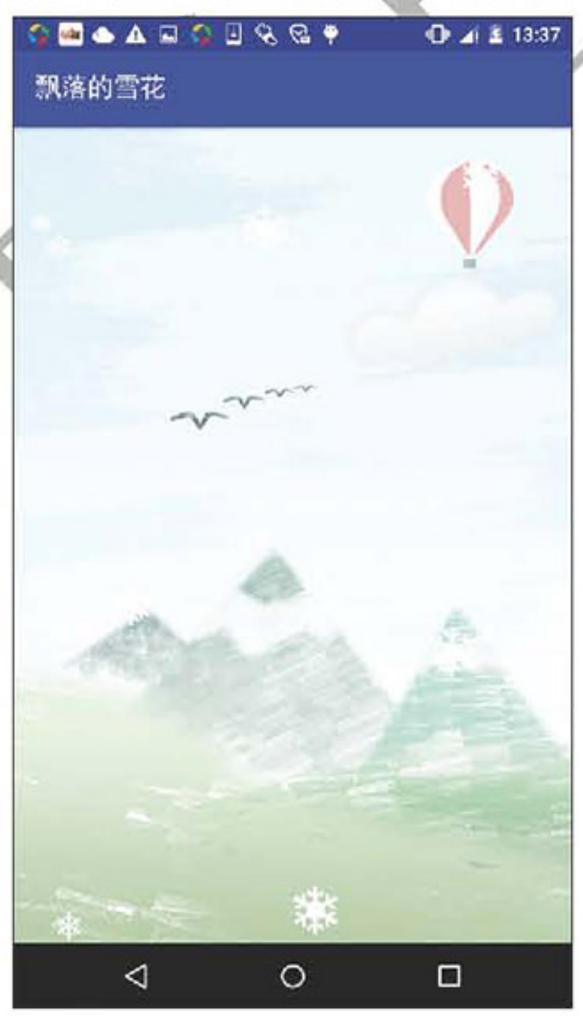


图 9.1　界面效果

图 9.2　雪花位置发生改变

关键技术

本实例主要使用两个技术：一个是 SurfaceView 控件，用来显示画面；另一个是 Thread 线程，用来控制画面的刷新。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“Snow”，公司域名为“mingrisoft. com”。

（2）导入资源文件：复制“光盘\Code\09\074\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）创建 DandelionModel 类，该类存放着将要绘制图片的位置和偏移量信息。创建实体类要声明以下变量。

代码位置：Snow\app\src\main\java\com\mingrisoft\snow\DandelionModel.java

```
01  private int pointX;                                    //绘制图片的横坐标
02  private int pointY;                                    //绘制图片的纵坐标
03  private int portOffset;                                //降落的偏移量
04  private int landOffset;                                //水平的偏移量
```

（4）创建 DandelionView 类，让该类继承 SurfaceView。然后实现 SurfaceHolder.CallBack 接口，在重写的 surfaceCreated() 方法中开启线程，代码如下：

代码位置：Snow\app\src\main\java\com\mingrisoft\snow\DandelionView.java

```
01  @Override
02  public void surfaceCreated(SurfaceHolder holder) {              //创建纹理
03      mThread = new Thread(this);
04      mThread.start();
05  }
06  @Override
07  public void surfaceChanged(SurfaceHolder holder, int format, int width, int height) {  //改变纹理
08  }
09  @Override
10  public void surfaceDestroyed(SurfaceHolder holder) {            //销毁纹理
11  }
```

实现 Runable() 接口，并在 run() 方法中填写绘制方法。在线程中会每隔 150 毫秒更新一次界面中数据模型的位置和偏移量，然后根据这些更改的数据，重新在画布中绘制界面的效果，从而形成动画的效果。代码如下：

```
01  @Override
02  public void run() {
03      while (isRunning) {
04          try {
05              mCanvas = mHolder.lockCanvas();                          //锁定画布
06              if (mCanvas != null) {
07                  mCanvas.drawColor(Color.TRANSPARENT, Mode.CLEAR);    //清空画布
```

```
            //绘制图形
            for (int i = 0; i < drawCounts; i++) {
                mCanvas.drawBitmap(bitmaps[0], dandelionModels_S.get(i).getPointX(),
                        dandelionModels_S.get(i).getPointY(), null);
                mCanvas.drawBitmap(bitmaps[1], dandelionModels_M.get(i).getPointX(),
                        dandelionModels_M.get(i).getPointY(), null);
                mCanvas.drawBitmap(bitmaps[2], dandelionModels_L.get(i).getPointX(),
                        dandelionModels_L.get(i).getPointY(), null);
                mCanvas.drawBitmap(bitmaps[3], dandelionModels_X.get(i).getPointX(),
                        dandelionModels_X.get(i).getPointY(), null);
                mCanvas.drawBitmap(bitmaps[4], dandelionModels_XX.get(i).getPointX(),
                        dandelionModels_XX.get(i).getPointY(), null);
            }
            //改变位置
            for (int i = 0; i < drawCounts;i++){
                offsetXY(dandelionModels_S.get(i),bitmaps[0]);
                offsetXY(dandelionModels_M.get(i),bitmaps[1]);
                offsetXY(dandelionModels_L.get(i),bitmaps[2]);
                offsetXY(dandelionModels_X.get(i),bitmaps[3]);
                offsetXY(dandelionModels_XX.get(i),bitmaps[4]);
            }
            Thread.sleep(150);                          //每150ms刷新一次UI
        }
            if (mCanvas != null) {
                mHolder.unlockCanvasAndPost(mCanvas);
            }
    } catch (InterruptedException e) {
        e.printStackTrace();
    }
  }
}
```

扩展学习

请尝试更改动画刷新的时间间隔，看看动画是否发生变化。

实例 075	种一朵花 应用场景：动画效果	源码位置：光盘\Code\09\075

实例说明

本实例将实现一个种一朵花的动画效果，这个开花的动画效果是由一个属性动画组合来完成的，也就是将一个个单独的动画效果组合到一起。运行实例，将显示如图 9.3 所示的界面；单击“开始动

画”按钮，将呈现一个花开的动画，并最终实现开出一朵蓝色花的效果，如图 9.4 所示。

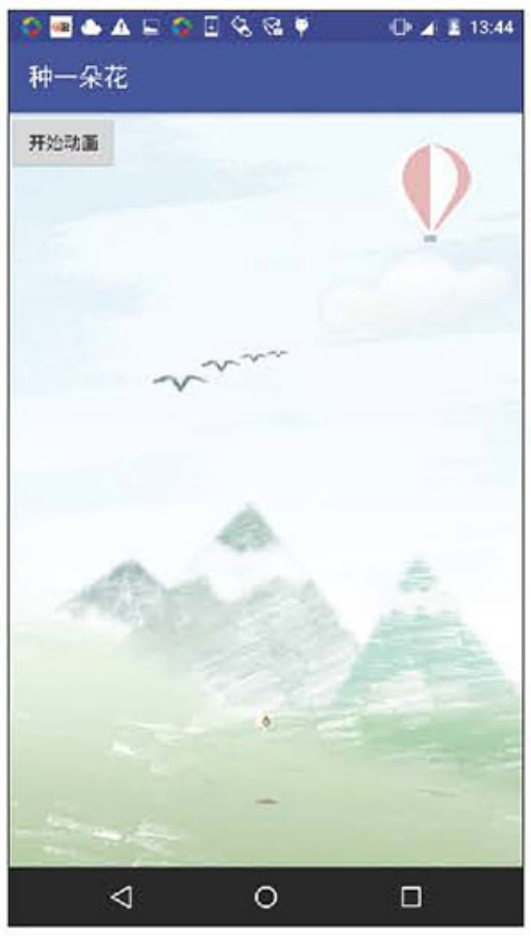


图 9.3　动画开始

图 9.4　开出花朵

关键技术

本实例主要使用属性动画实现动画效果，如通过设置控件的“translation”属性，来实现控件的上下移动。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“Plant”，公司域名为“mingrisoft. com”。

（2）导入资源文件：复制“光盘\Code\09\075\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）创建 Plant 类，在该类的 startShowPlantFlower() 方法中设置种子降落的动画效果，该动画属于一个平移动画的效果，设置动画效果的代码如下：

代码位置：Plant\app\src\main\java\com\mingrisoft\plant\Plant.java

```
ObjectAnimator seedTranslation = ObjectAnimator.ofFloat(seedImg,"translationY",
        0,childViewValues.get("seedMoveLength"));
seedTranslation.setDuration(5*1000);                                    //设置动画时间为5秒
```

设置根茎的缩放动画，缩放中心为根茎的底部，代码如下：

```
ObjectAnimator branchAnimator = ObjectAnimator.ofFloat(branchImg,"scaleY",0f,1.0f);
branchAnimator.setDuration(10*1000);                                    //设置动画时间为10秒
branchImg.setPivotY(childViewValues.get("BranchHeight"));               //设置动画的缩放中心
branchImg.invalidate();                                                 //重绘
```

设置叶子的动画效果。因为左侧叶子与右侧叶子的效果基本一致，本实例以左侧叶子为例，代码如下：

```
//缩放
PropertyValuesHolder leftLeafScaleX = PropertyValuesHolder.ofFloat("scaleX",0f,0.5f);
PropertyValuesHolder leftLeafScaleY = PropertyValuesHolder.ofFloat("scaleY",0f,0.5f);
//平移
leftLeafImg.setPivotX(childViewValues.get("LeftLeafWidth"));        //设置缩放中心x
leftLeafImg.setPivotY(childViewValues.get("LeftLeafHeight"));       //设置缩放中心y
PropertyValuesHolder leftLeafTranslation = PropertyValuesHolder.ofFloat("translationY",
        0,- childViewValues.get("BranchHeightHalf")*2/3);           //设置平移动画的效果
ObjectAnimator leftAnimator = ObjectAnimator.ofPropertyValuesHolder(leftLeafImg,
        leftLeafScaleX,leftLeafScaleY,leftLeafTranslation);
leftAnimator.setDuration(8*1000);                                   //设置动画时间为8秒
```

设置开花的动画效果。代码如下：

```
PropertyValuesHolder budGroupToFlowerXMore = PropertyValuesHolder.ofFloat(
        "scaleX",0.5f,1.0f);                                        //沿x轴的缩放效果
PropertyValuesHolder budGroupToFlowerYMore = PropertyValuesHolder.ofFloat(
        "scaleY",0.5f,1.0f);                                        //沿y轴的缩放效果
ObjectAnimator openFlowerAnimator = ObjectAnimator.ofPropertyValuesHolder(budImg,
        budGroupToFlowerXMore,budGroupToFlowerYMore );              //设置缩放动画
openFlowerAnimator.setDuration(5*1000);                             //动画时间为5秒
```

播放动画集合，将每个属性动画串联到一起，按照一定的顺序播放，从而形成一个花开的效果。代码如下：

```
//播放动画集合
//播放种子降落的动画和根茎的平移动画和叶子的缩放动画
animatorSetGroup.play(branchAnimator)
            .with(leftAnimator).with(rightAnimator).after(seedTranslation);
//播放叶子的缩放动画
animatorSetGroup.play(rightLeafAnimatorMore)
            .with(leftLeafAnimatorMore).after(leftAnimator);
//播放长花的动画
animatorSetGroup.play(budAnimator).after(branchAnimator);
//继续长花的动画
animatorSetGroup.play(budGroupAnimator).after(budAnimator);
//开花的动画
animatorSetGroup.play(openFlowerAnimator).after(budGroupAnimator);
```

（4）在 MainActivity 类中增加 start() 方法，用来开启动画，代码如下：
代码位置如下：Plant\app\src\main\java\com\mingrisoft\plant\MainActivity.java）

```
/**
 * 单击事件
```

```
03   * @param view
04   */
05  public void start(View view){
06      if (!plant.getIsRun()){
07          plant.plantAnimatorStart();                    //开启动画
08      }else {
09          Toast.makeText(this,"动画没有停止",Toast.LENGTH_SHORT).show();
10      }
11  }
```

扩展学习

为种子下落的动画设置插值器（用来控制动画的运行速率），以改变种子下落的动画效果。

实例 076　用刷子画外星人

应用场景：动画效果

源码位置：光盘\Code\09\076
视频位置：光盘\Video\09\076

视频讲解

实例说明

逐帧动画是一种常见的动画形式。本实例将使用逐帧动画实现用刷子逐步画出外星人的形象。运行实例，刷子将不断移动画出外星人的形象，如图 9.5 与图 9.6 所示。

图 9.5　绘画中（一）

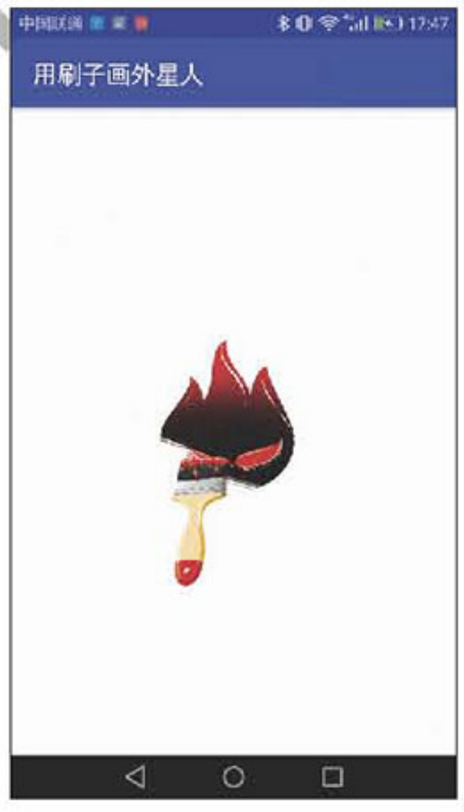


图 9.6　绘画中（二）

关键技术

本实例主要使用 AnimationDrawable 对象，控制逐帧动画的播放停止。

关键技术

实现过程

（1）打开 Android Studio，新建一个项目，命名为“Draw”，公司域名为“mingrisoft. com”。

（2）导入资源文件：复制“光盘\Code\09\076\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）因为在步骤（2）中已经把资源导入到项目中，所以该文件并不需要去创建，设置动画的 xml 关键代码如下：

代码位置：Draw\app\src\main\res\drawable\max.xml

```
<animation-list
    xmlns:android=http://schemas.android.com/apk/res/android
    android:oneshot="false">
    <item android:drawable="@mipmap/max_1" android:duration="150"/>
    <item android:drawable="@mipmap/max_2" android:duration="150"/>
    <item android:drawable="@mipmap/max_3" android:duration="150"/>
    <item android:drawable="@mipmap/max_4" android:duration="150"/>
    <item android:drawable="@mipmap/max_5" android:duration="150"/>
    <item android:drawable="@mipmap/max_6" android:duration="150"/>
    <item android:drawable="@mipmap/max_7" android:duration="150"/>
    <item android:drawable="@mipmap/max_8" android:duration="150"/>
```

（4）在 MainActivity 中的 onCreate() 方法中获取 AnimationDrawable 对象。关键代码如下：

代码位置：Draw\app\src\main\java\com\mingrisoft\draw\MainActivity.java

```
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
    imageView = (ImageView) findViewById(R.id.max);
    animationDrawable = (AnimationDrawable) imageView.getDrawable();
}
```

在 MainActivity 中的 onResume() 方法中开启逐帧动画。代码如下：

```
@Override
protected void onResume() {
    super.onResume();
    animationDrawable.start();                    //开启逐帧动画
}
```

在 MainActivity 中的 onPause() 方法中关闭逐帧动画。代码如下：

```
@Override
protected void onPause() {
    super.onPause();
    animationDrawable.stop();                     //关闭逐帧动画
}
```

扩展学习

请尝试将 XML 中 animation-list 标签的 oneshot 属性设置为 true，再查看输出的效果。

实例 077	会旋转的明日 Logo 应用场景：动画效果	源码位置：光盘\Code\09\077

实例说明

本实例将实现在页面中显示一个旋转的明日 Logo，该实例使用自定义渐变动画，实现图像沿手机纵轴旋转的效果。运行实例，界面的初始效果如图 9.7 所示；单击“开始动画”按钮，明日 Logo 将沿手机纵轴旋转，效果如图 9.8 所示。

图 9.7　初始效果

图 9.8　动画执行中

关键技术

关键技术

本实例主要使用 Animation 类的 applyTransformation() 方法，实现图像沿手机纵轴旋转的效果。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“CustomRotate”，公司域名为“mingrisoft. com”。

（2）导入资源文件：复制“光盘\Code\09\077\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）在 MainActivity 类中创建一个内部类，命名为 MyAnimation，并继承 Animation 类，再重写 initialize() 方法与 applyTransformation() 方法；在 initialize() 方法中获取一些参数，在 applyTransformation()

方法中设置动画的效果。该自定义动画类的具体代码如下：

代码位置：CustomRotate\app\src\main\java\com\mingrisoft\customrotate\MainActivity.java

```
/**
 * 自定义动画
 */
class MyAnimation extends Animation {
    private int x;                                          //图像的宽度
    private int y;                                          //图像的高度
    private Camera camera;
   @Override
    public void initialize(int width, int height, int parentWidth, int parentHeight) {
        super.initialize(width, height, parentWidth, parentHeight);
        this.x = width;
        this.y = height;
        this.setDuration(5000);                             //设置动画时间
        camera = new Camera();                              //获取Camera对象
    }
    @Override
    protected void applyTransformation(float interpolatedTime, Transformation t) {
        super.applyTransformation(interpolatedTime, t);
        final Matrix matrix = t.getMatrix();                //获取矩阵对象
        camera.save();                                      //保存当前视角
        camera.rotateY(360 * interpolatedTime);             //沿y轴旋转
        camera.getMatrix(matrix);                           //设置矩阵
        //图形变换中心
        matrix.preTranslate(-x / 2, -y);
        matrix.postTranslate(x / 2, y);
        camera.restore();                                   //恢复视角
    }
}
```

在 MainActivity 类的 onCreate() 方法中实例化控件与自定义动画，并为控件设置动画。代码如下：

```
image = (ImageView) findViewById(R.id.animation);           //实例化控件
myAnimation = new MyAnimation();                            //实例化动画对象
myAnimation.setAnimationListener(animationListener);        //给动画设置监听
myAnimation.setRepeatCount(0);                              //不重复播放
```

在 MainActivity 类中添加 start() 方法，在该方法中设置在单击按钮时播放动画。当动画开启时，重复单击按钮无效果。代码如下：

```
public void start(View view) {
    if (!isRunning){
        image.startAnimation(myAnimation);                  //播放动画
        isRunning = true;
}
```

明日科技

扩展学习

请尝试通过对 Marix 的设置，使动画效果从沿 y 轴旋转改为沿 x 轴旋转。

实例 078	会变色的花 应用场景：动画效果	源码位置：光盘\Code\09\078 视频位置：光盘\Video\09\078

实例说明

本实例将使用属性动画实现花的颜色渐变效果——从红花渐变到紫花。运行实例，将显示如图 9.9 所示的红色花朵，花朵颜色渐渐从红色变换到紫色，如图 9.10 所示。

图 9.9　花朵初始颜色

图 9.10　渐变到紫色

关键技术

本实例主要使用 ObjectAnimator 类的 ofint() 方法来获取属性动画对象，使用 setEvaluator() 方法设置估值器，使用 setRepeatCount() 方法设置动画的执行次数，使用 setRepeatMode() 方法设置动画循环的模式，使用 setDuration() 设置动画的执行时间，使用 start() 方法开启动画。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“FlowerColor”，公司域名为“mingrisoft. com”。

（2）导入资源文件：复制“光盘\Code\09\078\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）在 MainActivity 类中初始化控件，创建属性动画对象，并设置动画为循环播放，循环播放模式为“从结束开始循环动画”，最后开启动画。代码如下：

代码位置：FlowerColor\app\src\main\java\com\mingrisoft\flowercolor\MainActivity.java

```
01  public class MainActivity extends AppCompatActivity {
02      private RelativeLayout view;                                    //声明控件
03      @Override
04      protected void onCreate(Bundle savedInstanceState) {
05          super.onCreate(savedInstanceState);
06          setContentView(R.layout.activity_main);
07          //初始化控件
08          view  = (RelativeLayout) findViewById(R.id.activity_main);
09          //属性动画
10          ObjectAnimator parentAnimator = ObjectAnimator.ofInt(
11                  view,"backgroundColor",
12                  Color.parseColor("#ff0000"),
13                  Color.parseColor("#0000ff"));
14          parentAnimator.setEvaluator(new ArgbEvaluator());           //设置颜色估值器
15          parentAnimator.setRepeatCount(ValueAnimator.INFINITE);      //播放次数：循环
16          //播放模式：循环播放时不是从头开始，而是从结尾开始
17          parentAnimator.setRepeatMode(ValueAnimator.REVERSE);
18          parentAnimator.setDuration(5000);                           //播放时间为5000ms
19          parentAnimator.start();                                     //播放动画
20      }
21  }
```

扩展学习

请尝试将 setRepeatMode() 设置为 ValueAnimator.RESTART，再查看动画的实现效果。

实例 079　以动画的方式浏览照片

应用场景：动画效果

源码位置：光盘\Code\09\079

实例说明

本实例将实现用透明、缩放动画的效果切换照片，主要使用一个渐变动画组合，其中包括透明渐变动画和缩放动画。运行实例，将显示如图 9.11 所示的界面；当单击“上一张”按钮或者“下一张”按钮时，将显示如图 9.12 所示的切换图片时的动画效果。

明日科技

图 9.11　界面的默认效果

图 9.12　切换图片时的动画效果

关键技术

关键技术

本实例主要使用 AnimationUtils 类的 loadAnimation() 方法来加载 XML 文件中设置的渐变动画，使用 ImageView 类的 setAnimation() 方法来设置控件的动画效果，使用 Animation 类的 start() 方法来开启动画。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“ChangePicture”，公司域名为“mingrisoft. com”。

（2）导入资源文件：复制“光盘\Code\09\079\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）创建 hide_out.xml 文件，用来设置渐变动画的效果。代码如下：

代码位置：ChangePicture\app\src\main\res\anim\hide_out.xml

```
<?xml version="1.0" encoding="utf-8"?>
<set xmlns:android="http://schemas.android.com/apk/res/android"
    android:interpolator="@android:anim/accelerate_interpolator">
    <!-- 透明渐变
        duration：动画时间
        fromAlpha：开始的透明度
        toAlpha：结束的透明度
    -->
    <alpha
        android:duration="2000"
        android:fromAlpha="0.0"
```

```
        android:toAlpha="1.0" />
    <!-- 缩放渐变
        duration：动画时间
        fromXScale：横向缩放开始值
        fromYScale：纵向缩放开始值
        pivotX：缩放中心x
        pivotY：缩放中心y
        toXScale：横向缩放结束值
        toYScale：纵向缩放结束值
     -->
    <scale
        android:duration="2000"
        android:fromXScale="0.1"
        android:fromYScale="0.1"
        android:pivotX="50%"
        android:pivotY="50%"
        android:toXScale="1.0"
        android:toYScale="1.0" />
</set>
```

（4）在 MainActivity 类中加载布局文件中的动画，并初始化显示图片控件和按钮控件，代码如下：
代码位置：ChangePicture\app\src\main\java\com\mingrisoft\changePicture\MainActivity.java

```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
    isInit = true;                              //设置初始化界面
    //获取动画
    showAnimation = AnimationUtils.loadAnimation(this,R.anim.show_in);
    hideAnimation = AnimationUtils.loadAnimation(this,R.anim.hide_out);
    //图片控件集合
    images = new ArrayList<>();
    //初始化显示图片的控件
    images.add((ImageView) findViewById(R.id.image_one));
    images.add((ImageView) findViewById(R.id.image_two));
    images.add((ImageView) findViewById(R.id.image_three));
    images.add((ImageView) findViewById(R.id.image_four));
    images.add((ImageView) findViewById(R.id.image_five));
    //初始化按钮控件
    lastBtn = (ImageButton) findViewById(R.id.last);
    nextBtn = (ImageButton) findViewById(R.id.next);
    //设置单击监听事件
    lastBtn.setOnClickListener(this);
    nextBtn.setOnClickListener(this);
    //初始化显示图片
```

明日科技

```
24          showImage();
25      }
```

（5）在 MainActivity 类中添加 onClick() 方法，用来设置单击按钮时切换显示图片的控件，并为显示图片的控件设置动画。代码如下：

```
01  /**
02   * 切换图片
03   *
04   */
05  @Override
06  public void onClick(View v) {
07      isInit = false;                          //设置不是初始化界面
08      lastShowIndex = showIndex;               //记录当前索引
09      switch (v.getId()){
10          case R.id.last:                      //上一张图片
11              showIndex--;                     //切换到上一张
12              break;
13          case R.id.next:                      //下一张图片
14              showIndex++;                     //切换到下一张
15              break;
16      }
17      showImage();                             //显示图片
18  }
```

扩展学习

请尝试将图片的切换动画效果更改为旋转缩放的动画。

实例 080 拍立得打印照片的动画效果

应用场景：移出消息提示

源码位置：光盘\Code\09\080

实例说明

本实例将实现打印照片的动画效果。运行实例，将播放一个平移渐变动画，具体平移动画的过程，如图 9.13 和图 9.14 所示。

关键技术

关键技术

本实例主要使用 AnimationUtils 类的 loadAnimation() 方法加载动画资源，使用 Animation 类中的 setRepeatCount() 方法设置动画的执行次数，使用 setRepeatMode() 方法设置动画的循环模式。

明日科技

图 9.13　动画开始时

图 9.14　动画结束后

实现过程

（1）打开 Android Studio，新建一个项目，命名为“PrintPicture”，公司域名为“mingrisoft. com”。

（2）导入资源文件：复制“光盘\Code\09\080\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）一般情况下，创建渐变动画的文件是在 res 文件夹下创建 anim 资源文件夹以及资源 XML 文件，然后在该文件中设置动画的参数，用来控制动画的效果。代码如下：

代码位置：PrintPicture\app\src\main\res\anim\print.xml

```
<?xml version="1.0" encoding="utf-8"?>1
<!-- 设置动画效果从相对与本身减去一个本身的高度开始动画
到图像完全显示结束动画时间为3000ms
-->
<translate xmlns:android=http://schemas.android.com/apk/res/android
    android:fromYDelta="-100%" android:toYDelta="0" android:duration="3000"/>
```

（4）在MainActivity 类中的 onCreate() 方法中初始化控件和动画对象，然后播放动画。关键代码如下：

代码位置：PrintPicture\app\src\main\java\com\mingrisoft\printpicture\MainActivity.java

```
public class MainActivity extends AppCompatActivity {

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

```
07          ImageView image = (ImageView) findViewById(R.id.picture);          //实例化控件
08          //加载动画
09          Animation animation = AnimationUtils.loadAnimation(this,R.anim.print);
10          image.setAnimation(animation);                                     //设置动画
11          animation.setRepeatCount(Animation.INFINITE);                      //设置动画循环播放
12          animation.setRepeatMode(Animation.RESTART);                        //动画在开始处播放
13          animation.start();                                                 //播放动画
14      }
15  }
```

扩展学习

请尝试为动画添加监听器，并在动画循环和开始的时候弹出 Toast 消息。

实例 081　两只会飞的鸟

应用场景：动画效果

源码位置：光盘\Code\09\081

视频位置：光盘\Video\09\081

实例说明

本实例将使用逐帧动画与属性动画联合实现鸟的飞行效果。运行实例，将显示动画开始的界面，如图 9.15 所示；在进入界面后会有两只鸟从屏幕的左边飞到屏幕的右边，并伴随着拍打着翅膀的动作，如图 9.16 所示。

图 9.15　动画开始

图 9.16　动画执行中

关键技术

本实例主要使用 ObjectAnimaton 类的 setInterpolator() 方法，通过逐帧动画与属性动画的联合作用，实现鸟飞行的动画效果。

实现过程

（1）在 drawable 文件夹下创建资源 XML 文件，然后在 XML 文件中设置逐帧动画的参数，用来控制动画的效果。代码如下：

```
<?xml version="1.0" encoding="utf-8"?>
<animation-list xmlns:android=http://schemas.android.com/apk/res/android
    android:oneshot="false">
    <item android:drawable="@mipmap/mrkj_plant_bird_01" android:duration="500"/>
    <item android:drawable="@mipmap/mrkj_plant_bird_02" android:duration="500"/>
</animation-list>
```

（2）获取屏幕的宽度，代码如下：

```
/**
 * 获取屏幕的宽度
 */
private void getWindowWidth(){
    DisplayMetrics dm = new DisplayMetrics();
    getWindowManager().getDefaultDisplay().getMetrics(dm);
    screenWidth = dm.widthPixels;
}
```

（3）初始化动画对象和控件对象后为控件添加动画。代码如下：

```
getWindowWidth();                                          //获取屏幕宽度
bird = (ImageView) findViewById(R.id.bird);
bird.setTranslationX(-screenWidth);                        //设置大雁摆放位置向左平移一个屏幕的宽
birdAnimation = (AnimationDrawable) bird.getDrawable();    //获取逐帧动画
//设置鸟的动画
birdAnimatorset = new AnimatorSet();
ObjectAnimator birdAnimatorR = ObjectAnimator
                    .ofFloat(bird,"translationX",screenWidth);
birdAnimatorR.setDuration(30*1000);                        //设置动画时间
birdAnimatorR.setInterpolator(new LinearInterpolator());   //设置插值器
birdAnimatorR.setRepeatCount(ValueAnimator.RESTART);       //设置从头开始循环
birdAnimatorR.setRepeatCount(ValueAnimator.INFINITE);      //设置循环播放
birdAnimatorset.play(birdAnimatorR);                       //向动画集合中添加动画
birdAnimation.start();                                     //开启逐帧动画
birdAnimatorset.start();                                   //开启属性动画
```

明日科技

扩展学习

该实例是属性动画与逐帧动画的结合使用，请尝试结合使用渐变动画与逐帧动画，实现动画效果。

实例 082	仿小米手机卸载应用动画效果 应用场景：粒子动画效果	源码位置：光盘\Code\09\082

实例说明

小米手机在卸载应用程序时，会有一个爆炸式的破碎效果，这是粒子动画中的一种。本实例将实现类似粒子破碎的动画效果。运行实例，效果如图9.17所示；单击执行动画按钮，细小颗粒散落开来，将显示图 9.18 所示的界面。

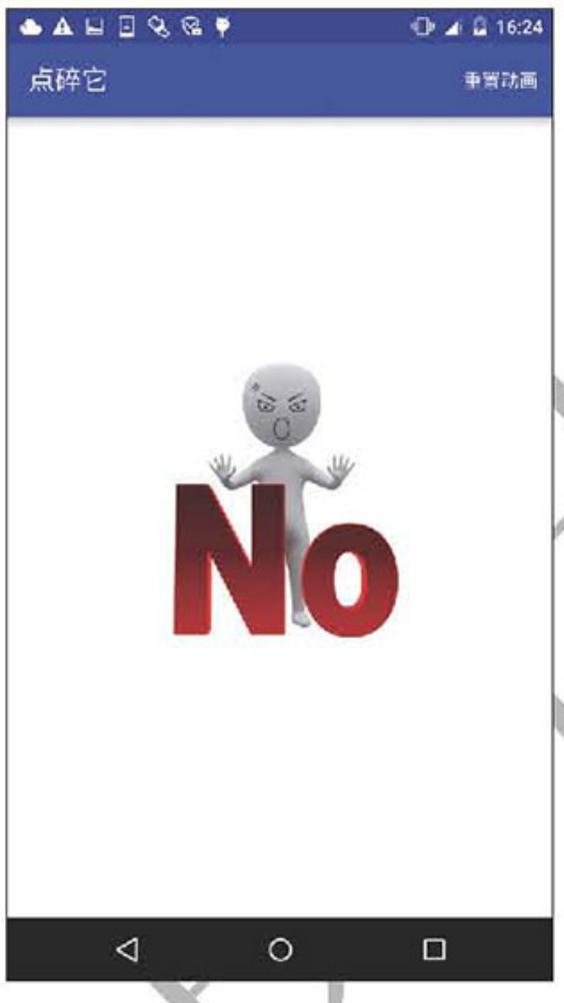


图 9.17　动画开始前

图 9.18　动画执行

关键技术

关键技术

本实例主要使用 ExplosionField 开源项目实现仿小米手机卸载应用的粒子破碎效果的动画。在实现时，需要在项目中依赖“compile 'tyrantgit:explosionfield:1.0.1'”。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“Broken”，公司域名为“mingrisoft. com”。

（2）导入资源文件：复制“光盘\Code\09\082\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）在 MainActivity 类中的 onCreate() 方法中初始化控件和 ExplosionField 对象。代码如下：

代码位置：Broken\app\src\main\java\com\mingrisoft\broken\MainActivity.java

```
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
    image = (ImageView) findViewById(R.id.picture);                    //初始化控件
    mExplosionField = ExplosionField.attach2Window(this);              //获取粒子动画对象
    addListener();                                                     //增加监听事件
}
```

（4）在 MainActivity 类中添加 addListener() 方法，用来设置 ImageView 的单击事件。代码如下：

```
private void addListener(){
    //设置控件的单击事件
    image.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            mExplosionField.explode(v);                                //执行粒子动画
            v.setOnClickListener(null);
        }
    });
}
```

（5）在 MainActivity 类中重写 onOptionsItemSelected() 方法，用来设置菜单选项的单击事件，当单击事件触发后，重置界面中控件的显示效果。代码如下：

```
@Override
public boolean onOptionsItemSelected(MenuItem item) {
    if (item.getItemId() == R.id.reset){
        image.setScaleX(1f);                                           //设置x轴缩放值
        image.setScaleY(1f);                                           //设置y轴缩放值
        image.setAlpha(1f);                                            //设置透明度
        image.setClickable(true);                                      //设置可单击
        addListener();                                                 //增加监听
        mExplosionField.clear();                                       //清除效果
    }
    return super.onOptionsItemSelected(item);
}
```

扩展学习

请尝试更换一张图片，实现动画的运行效果。

实例 083 圆形扩散效果

应用场景：动画效果

源码位置：光盘\Code\09\083

实例说明

本实例将使用属性动画的超类 Animator 实现圆形扩散效果。运行实例，动画效果如图 9.19 所示，结束时的界面如图 9.20 所示。

图 9.19　动画执行中

图 9.20　动画结束

关键技术

关键技术

本实例主要使用 ViewAnimationUtils.createCircularReveal() 方法获取 Animator 对象。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“Spread”，公司域名为“mingrisoft. com”。

（2）导入资源文件：复制“光盘\Code\09\083\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）在 MainActivity 类中初始化控件，为其中的 ImageVew 设置单击事件，并在单击事件中设置动画。代码如下：

代码位置：Spread\app\src\main\java\com\mingrisoft\spread\MainActivity.java

```
01  public class MainActivity extends AppCompatActivity {
02      private RelativeLayout home;
03      private ImageView image;
04      private LinearLayout showAnimation;
```

```
    private Animator animator;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        //初始化RelativeLayout
        home = (RelativeLayout) findViewById(R.id.home_group);
        //初始化ImageView
        image = (ImageView) findViewById(R.id.image1);
        //初始化LinearLayout
        showAnimation = (LinearLayout) findViewById(R.id.showAnimation);
        //设置ImageView的单击事件
        image.setOnClickListener(new View.OnClickListener() {
            @TargetApi(Build.VERSION_CODES.LOLLIPOP)
            @Override
            public void onClick(View v) {
                int width = home.getWidth();                                    //获取宽度
                int height = home.getHeight();                                  //获取高度
                int centerX = width>>1;                                         //获取宽度的一半
                int centerY = height>>1;                                        //获取高度的一半
                float finalRadius = (float) Math.hypot(width, height);          //获取半径
                animator = ViewAnimationUtils.createCircularReveal(             //设置动画
                        showAnimation, centerX, centerY, 0, finalRadius);
                animator.setInterpolator(new AccelerateDecelerateInterpolator());//设置插值器
                animator.setDuration(3000);                                     //设置动画时间
                animator.start();                                               //播放动画
            }
        });
    }
}
```

扩展学习

请尝试更改动画的插值器设置，查看动画的执行速度发生哪些变化。

实例 084	明日学院唱片式音乐播放器 应用场景：音乐播放器	源码位置：光盘\Code\09\084

实例说明

很多人对老式唱片情有独钟，于是出现了模拟唱片播放的网易音乐播放器。本实例将模仿网易音

明日科技

乐播放器的音乐播放效果，实现一个播放器。运行实例，播放音乐的界面如图 9.21 所示，暂停播放的界面如图 9.22 所示，切换播放音乐的界面如图 9.23 所示。

图 9.21　音乐播放中

图 9.22　音乐暂停中的界面

图 9.23　音乐切换界面

关键技术

本实例主要使用 Bitmap 类中的 creatBitmap() 方法、BitmapFactory 类中的 decodeResource() 方法，实现创建视图及加载图片。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“WangyiMusic”，公司域名为“mingrisoft.com”。

（2）导入资源文件：复制“光盘\Code\09\084\src”文件夹中的图片资源粘贴到“app\res”文件夹中的 drawable 文件夹。

（3）创建一个播放音乐的服务类，命名为“MusicService”，主要用于当界面对用户不可见时，还能继续播放音乐，关键代码如下：

```
01  /**
02   * 编写内部的广播
03   **/
04  class MusicReceiver extends BroadcastReceiver {
05      @Override
06      public void onReceive(Context context, Intent intent) {
07          String action = intent.getAction();                    //获取action
08          if (action.equals(ACTION_OPT_MUSIC_PLAY)) {            //音乐播放
```

```
09              play(mCurrentMusicIndex);
10          } else if (action.equals(ACTION_OPT_MUSIC_PAUSE)) {      //音乐暂停
11              pause();                                             //音乐暂停的方法
12          } else if (action.equals(ACTION_OPT_MUSIC_LAST)) {       //上一首
13              last();                                              //上一首音乐的方法
14          } else if (action.equals(ACTION_OPT_MUSIC_NEXT)) {       //下一首
15              next();                                              //下一首音乐的方法
16          } else if (action.equals(ACTION_OPT_MUSIC_SEEK_TO)) {    //寻找音乐
17              seekTo(intent);                                      //查找音乐的方法
18          }
19      }
20  }
```

（4）创建 MusicData 类，该类主要用来获取音乐文件，包括音乐的名称、图片和作者等，关键代码如下：

```
01  private int mMusicRes;                                          //音乐资源id
02  private int mMusicPicRes;                                       //专辑图片id
03  private String mMusicName;                                      //音乐名称
04  private String mMusicAuthor;                                    //作者
05  /**
06   * 初始化   用于其他界面调用
07   */
08  public MusicData(int mMusicRes, int mMusicPicRes, String mMusicName, String mMusicAuthor) {
09      this.mMusicRes = mMusicRes;
10      this.mMusicPicRes = mMusicPicRes;
11      this.mMusicName = mMusicName;
12      this.mMusicAuthor = mMusicAuthor;
13  }
```

（5）创建 DisplayUtil 类，该类主要用于设置唱片与唱针的比例及宽度与高度，关键代码如下：

```
01  /*唱针宽、高、距离等比例*/
02  public static final float SCALE_NEEDLE_WIDTH = (float) (276.0 / BASE_SCREEN_WIDTH);
03  public static final float SCALE_NEEDLE_MARGIN_LEFT = (float) (500.0 / BASE_SCREEN_WIDTH);
04  public static final float SCALE_NEEDLE_PIVOT_X = (float) (43.0 / BASE_SCREEN_WIDTH);
05  public static final float SCALE_NEEDLE_PIVOT_Y = (float) (43.0 / BASE_SCREEN_WIDTH);
06  public static final float SCALE_NEEDLE_HEIGHT = (float) (413.0 / BASE_SCREEN_HEIGHT);
07  public static final float SCALE_NEEDLE_MARGIN_TOP = (float) (43.0 / BASE_SCREEN_HEIGHT);
08  /*唱盘比例*/
09  public static final float SCALE_DISC_SIZE = (float) (813.0 / BASE_SCREEN_WIDTH);
10  public static final float SCALE_DISC_MARGIN_TOP = (float) (190 / BASE_SCREEN_HEIGHT);
11  /*专辑图片比例*/
12  public static final float SCALE_MUSIC_PIC_SIZE = (float) (533.0 / BASE_SCREEN_WIDTH);
```

扩展学习

本实例中唱片背景为透明颜色，尝试修改唱片背景图片。

实例 085	点赞效果 应用场景：控件动画效果	源码位置：光盘\Code\09\085

实例说明

本实例将通过使用懒人的方式来实现一个好看的点赞效果。运行实例，未点赞的界面效果如图 9.24 所示；点赞的界面效果，如图 9.25 所示。

图 9.24　未点赞的效果

图 9.25　点赞的效果

关键技术

关键技术

本实例主要使用第三方的 GoodView 库实现点赞效果，在实现时，需要在项目中添加“compile 'com.wx.goodview:goodview:1.0.0'”。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“ThumbUp”，公司域名为“mingrisoft. com”。

（2）向项目中添加依赖“compile 'com.wx.goodview:goodview:1.0.0 '”（文件位置：ThumbUp\app\build.gradle）。

（3）导入资源文件：复制“光盘\Code\09\085\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（4）在 MainActivity 类中实例化控件和 GoodView。代码如下：

代码位置：ThumbUp\app\src\main\java\com\mingrisoft\thumbup\PrizeView.java

```
        //初始化控件
        zan = (LinearLayout) findViewById(R.id.zan);
        tu = (ImageView) findViewById(R.id.tu);
        //初始化点赞效果
        mGoodView = new GoodView(this);
```

设置点赞效果的切换功能。代码如下：

```
//设置“点赞”按钮的单击事件
zan.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        if (isGood){                                    //判断是否点过赞
            tu.setImageResource(R.mipmap.three_1);
            mGoodView.reset();                          //重置
            isGood = false;                             //设置为点过赞
        }else {
            tu.setImageResource(R.mipmap.three_2);
            mGoodView.setImage(R.mipmap.three_2);
            mGoodView.show(tu);                         //设置显示的图
            isGood = true;                              //设置为未点赞
        }
    }
});
```

扩展学习

请尝试为点赞效果增加一个计数的效果，即单击一次后点赞次数加 1，再次单击的时候减 1。

实例 086　仿支付宝咻一咻

应用场景：地震波动画

源码位置：光盘\Code\09\086

实例说明

支付宝咻一咻功能是通过 ObjectAnimator 对象动画类来实现的一种地震波动画效果。运行实例，单击手机或模拟器中的触发按钮，如图 9.26 所示；显示地震波动画的界面效果，如图 9.27 所示；3 秒后将显示查找到的设备，如图 9.28 所示。

明日科技

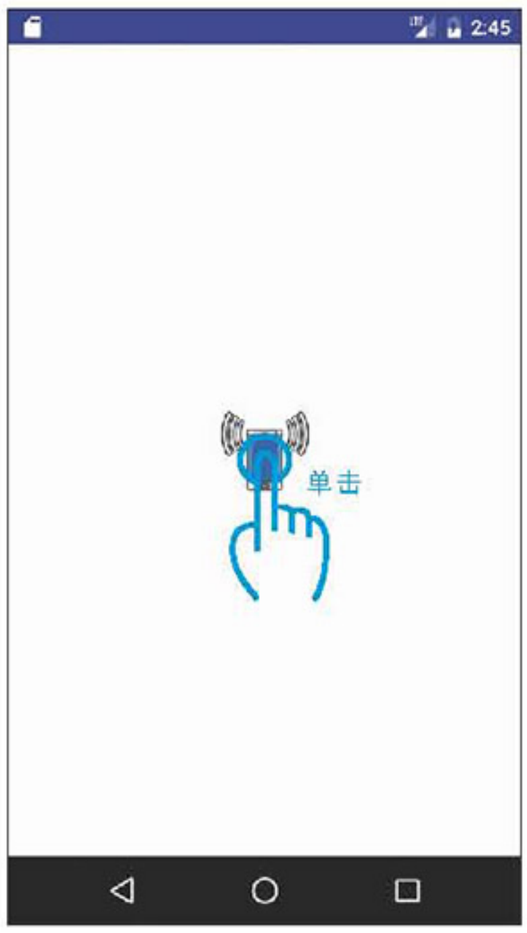


图 9.26　单击触发按钮

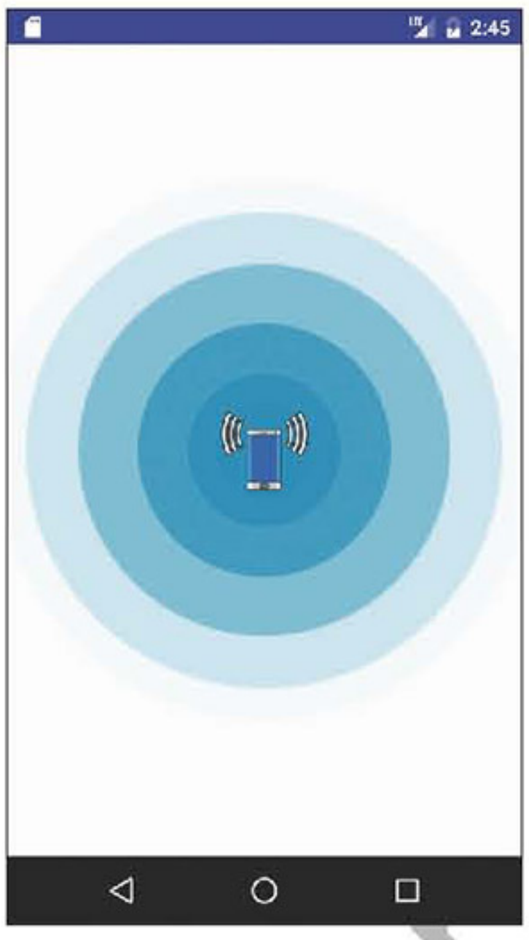


图 9.27　显示地震波动画

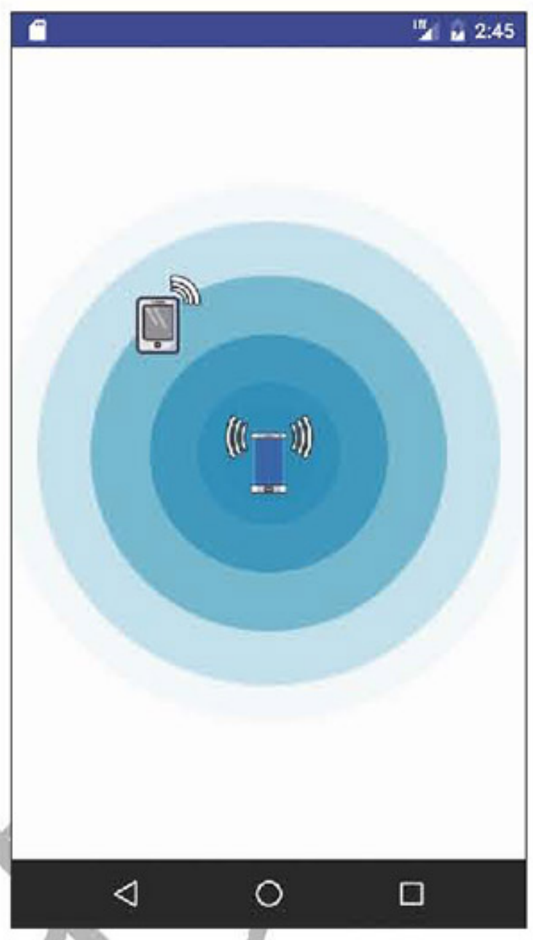


图 9.28　显示查找到的设备

关键技术

本实例主要使用 ObjectAnimator 类中的 ofFloat() 方法和 ValueAnimator 类中的 setRepeatCount() 方法、setStartDelay() 方法，实现咻一咻的运行效果。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“ShoopAShoop”，公司域名为“mingrisoft. com”。

（2）导入资源文件：复制“光盘\Code\09\086\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）在“app\src\main\java”目录中创建 view 包，在该包中创建 WaveControlLayout 类，该类继承自 RelativeLayout，用于实现自定义地震波动画的显示。在 init() 自定义的方法中，首先设置自定义控件的属性，然后设置地震波波幅横向拉伸的动画，代码如下：

```
final ObjectAnimator scaleXAnimator = ObjectAnimator.
        ofFloat(waveView, "ScaleX", 1.0f, waveScale);
scaleXAnimator.setRepeatCount(ObjectAnimator.INFINITE);          //无限重复
scaleXAnimator.setRepeatMode(ObjectAnimator.RESTART);            //动画重新启动
scaleXAnimator.setStartDelay(i * waveDelay);                     //波幅延迟的时间
scaleXAnimator.setDuration(waveDurationTime);                    //波幅持续的时间
animatorList.add(scaleXAnimator);                                //横向拉伸动画添加进动画数组
```

设置地震波波幅纵向拉伸的动画，代码如下：

```
final ObjectAnimator scaleYAnimator = ObjectAnimator.
```

```
02          ofFloat(waveView, "ScaleY", 1.0f, waveScale);
03  scaleYAnimator.setRepeatCount(ObjectAnimator.INFINITE);          //无限重复
04  scaleYAnimator.setRepeatMode(ObjectAnimator.RESTART);            //动画重新启动
05  scaleYAnimator.setStartDelay(i * waveDelay);                     //波幅延迟的时间
06  scaleYAnimator.setDuration(waveDurationTime);                    //波幅持续的时间
07  animatorList.add(scaleYAnimator);                                //纵向拉伸动画添加进动画数组
```

设置地震波波幅消失动画，代码如下：

```
01  final ObjectAnimator alphaAnimator = ObjectAnimator.
02          ofFloat(waveView, "Alpha", 1.0f, 0f);
03  alphaAnimator.setRepeatCount(ObjectAnimator.INFINITE);           //无限重复
04  alphaAnimator.setRepeatMode(ObjectAnimator.RESTART);             //动画重新启动
05  alphaAnimator.setStartDelay(i * waveDelay);                      //波幅延迟的时间
06  alphaAnimator.setDuration(waveDurationTime);                     //波幅持续的时间
07  animatorList.add(alphaAnimator);                                 //消失动画添加进动画数组
```

（4）打开 MainActivity 类，在默认生成的 onCreate() 方法中实现地震波动画的启动。代码如下：

```
01  //获取自定义波幅控件
02  final WaveControlLayout waveControlLayout =
03          (WaveControlLayout)findViewById(R.id.activity_main);
04  final Handler handler=new Handler();                             //创建handler消息机制
05  found=(ImageView)findViewById(R.id.found);                       //获取并显示找到的图片控件
06  ImageView button=(ImageView)findViewById(R.id.centerImage);      //获取触发的按钮
07  button.setOnClickListener(new View.OnClickListener() {           //设置按钮单击事件
08      @Override
09      public void onClick(View view) {
10              waveControlLayout.startWaveAnimation();              //启动波幅动画
11              handler.postDelayed(new Runnable() {
12                  @Override
13                  public void run() {
14                      found();                                     //调用显示找到图片的动画方法
15                  }
16              },3000);                                            //设置3秒后执行
17      }
18  });
```

扩展学习

本实例通过单击触发按钮显示地震波的动画，实现咻一咻的运行效果。请尝试在默认生成的 onCreate() 方法中判断地震波动画是否启动，如果启动单击触发按钮，将停止动画并隐藏找到的图片。

实例 087 数字滚动效果

应用场景：滚动数字动画

源码位置：光盘\Code\09\087

实例说明

本实例将通过 BigInteger 类和 BigDecimal 类实现整数与小数的筛选，然后通过 ValueAnimator 类实现数字滚动的效果。运行实例，效果如图 9.29 所示；单击“START”按钮，将显示数字滚动的动画效果，如图 9.30 所示。

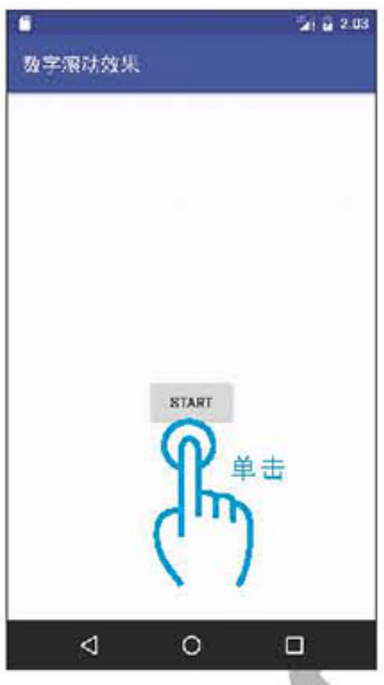


图 9.29 单击“启动”按钮 图 9.30 显示数字滚动效果

关键技术

本实例主要使用 BigInteger 类中的 BigInteger() 构造方法、Format 类中的 format() 方法、ValueAnimator 类中的 ofObject() 方法，实现数字滚动的动画效果。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“RollingNumber”，公司域名为“mingrisoft. com”。

（2）导入资源文件：复制“光盘\Code\09\087\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）在“app\src\main\java”目录中创建 view 包，在该包中创建 ScrollingDigitalAnimation 类，该类继承自 TextView，用于实现自定义数字滚动的动画。通过自定义 setNumberString() 方法实现设置起始数字与最终数字。代码如下：

```
01  public void setNumberString(String numberStart, String numberEnd) {
02      numStart = numberStart;                    //获得设置的起始数字
03      numEnd = numberEnd;                        //获得设置的最终数字
04      if (checkNumString(numberStart, numberEnd)) {
05          //如果数字合法则开始数字动画
06          start();
```

```
07      } else {
08          //如果数字不合法则直接调用setText()方法设置最终值
09          setText(prefixString + numberEnd + postfixString);
10      }
11  }
```

通过自定义 format() 方法实现整数与小数的格式化处理。代码如下：

```
01  private String format(BigDecimal bd) {
02      String pattern;
03      if (isInt) {                                    //如果是整数
04          pattern = "#,###";                          //整数格式
05      } else {
06          pattern = "#,##0.00";                       //小数格式
07      }
08      DecimalFormat df = new DecimalFormat(pattern);  //进行格式化
09      return df.format(bd);                           //返回格式化后的字符串
10  }
```

通过自定义 start() 方法实现数字滚动的动画显示功能。代码如下：

```
01  private void start() {
02      //创建数字动画，并设置起始值和最终值
03      ValueAnimator animator = ValueAnimator.ofObject(new BigDecimalEvaluator(),
04              new BigDecimal(numStart), new BigDecimal(numEnd));
05      animator.setDuration(duration);                 //设置动画持续的时间
06      //设置动画内插器
07      animator.setInterpolator(new AccelerateDecelerateInterpolator());
08      //动画监听器
09      animator.addUpdateListener(new ValueAnimator.AnimatorUpdateListener() {
10          @Override
11          public void onAnimationUpdate(ValueAnimator valueAnimator) {
12              BigDecimal value = (BigDecimal) valueAnimator.getAnimatedValue();
13              //设置显示数值
14              setText(prefixString + format(value) + postfixString);
15          }
16      });
17      animator.start();                               //启动动画
18  }
```

（4）打开 MainActivity 类，在默认生成的 onCreate() 方法中，首先获取显示滚动数字的文本控件，然后创建 start() 方法，在该方法中实现启动数字滚动的动画效果。

扩展学习

请尝试在 ScrollingDigitalAnimation 类中创建 setDuration() 方法，并在该方法中设置动画持续的时间，然后在 MainActivity 类的 start() 方法中设置每组滚动数字动画不同的持续时间。

第10章 多媒体

实例088 带音效的火山爆发效果

应用场景：短时间的音频提示

源码位置：光盘\Code\10\088

实例说明

在App中可以使用SoundPool类实现一些短的音效。SoundPool类载入音乐文件使用了独立的线程，不会阻塞UI主线程的操作。本实例将用SoundPool类播放火山爆发的音效，并调用火山爆发的背景图片。运行实例，显示默认的背景效果，如图10.1所示；触摸屏幕，播放音频，并显示火山爆发的图片，如图10.2所示。

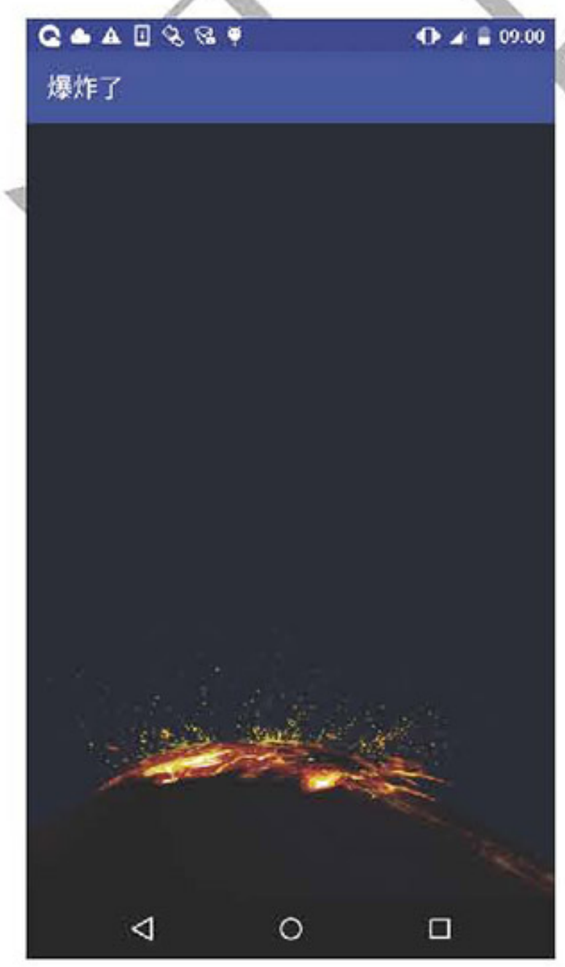


图10.1 默认背景效果

图10.2 播放音频更换背景

关键技术

本实例主要使用 SoundPool 类中的 load() 方法加载音频和 play() 方法播放音频。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“Boom”，公司域名为“mingrisoft. com”。

（2）导入资源文件：复制“光盘\Code\10\088\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）在 MainActivity 类的 onCreate() 方法中设置播放音效的属性，然后实例化 SoundPool 对象并加载音效资源。代码如下：

代码位置：Boom\app\src\main\java\com\mingrisoft\boom\MainActivity.java

```
AudioAttributes attr = new AudioAttributes.Builder()
        .setUsage(AudioAttributes.USAGE_GAME)                    //设置音效使用场景
        .setContentType(AudioAttributes.CONTENT_TYPE_MUSIC)      //设置音效的类型
        .build();
soundPool = new SoundPool.Builder()
        .setAudioAttributes(attr)                                //设置音效池的属性
        .setMaxStreams(10)                                       //设置最多可容纳10个音频流
        .build();                                                //创建soundPool对象
soundId = soundPool.load(this, R.raw.shot, 1);                   //加载音频
```

在 MainActivity 中重写 onTouchEvent() 方法，用来监听 Activity 的触摸事件，当手势抬起的时候就播放音乐。代码如下：

```
/**
 * 每一次的触摸屏幕都会开启一段音效
 * @param event
 * @return
 */
@Override
public boolean onTouchEvent(MotionEvent event) {
    switch (event.getAction()){
        case MotionEvent.ACTION_DOWN:                            //触摸屏幕
            //更改图片
            relativeLayout.setBackgroundResource(R.mipmap.background1);
            break;
        case MotionEvent.ACTION_UP:                              //触摸抬起
            soundPool.play(soundId, 1, 1, 0, 0, 1);              //播放音频
            //更改图片
            relativeLayout.setBackgroundResource(R.mipmap.background2);
            return true;
    }
    return super.onTouchEvent(event);
}
```

明日科技

扩展学习

请尝试更改 soundPool.play() 方法中的参数，以修改播放器的效果，查看会对播放中的音频产生哪些影响。

实例 089　显示歌词的音乐播放器

应用场景：播放音乐

源码位置：光盘\Code\10\089

实例说明

本实例将通过调用 MediaPlayer 类实现播放音乐的功能，并将歌词显示在页面中。运行实例，将显示如图 10.3 所示的初始界面；单击播放按钮，将播放指定的歌曲，然后将歌词显示在页面上，如图 10.4 所示。

图 10.3　音乐播放器初始界面

图 10.4　播放音乐中

关键技术

关键技术

本实例主要使用 MediaPlayer 类实现音频的播放功能。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“MusicPlayer”，公司域名为“mingrisoft. com”。

（2）导入资源文件：复制“光盘\Code\10\089\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）在 MusicPlayer 项目中的“app\src\main\java”目录下创建 utils 包，该包用于保存本项目中所使用到的工具类文件，该包中包括 3 个类文件，其类名及作用如表 10.1 所示。

表 10.1　utils 包中的三个类

类　　名	描　　述
History	用来读取轻量级数据内容
SearchFile	用来扫描歌曲文件和歌词文件
ToastUtils	弹出 Toast 的工具类

（4）在项目中的“app\src\main\java”目录下创建 activity 包，用来存放 Activity，并将 MainActivity.java 文件放到 activity 包中。然后在 MainActivity 类中添加 initialization() 方法，用来执行界面的初始化操作。代码如下：

代码位置：MusicPlayer\app\src\main\java\com\mingrisoft\musicplayer\MainActivity.java

```
01  /**
02   * 初始化操作
03   */
04  private void initialization() {
05      //扫描本地音乐
06      musics = SearchFile.searchLocaltion(this);
07      if (musicList != null) {
08          //绑定服务
09          Intent intent = new Intent(this, PlayMusicService.class);
10          intent.putParcelableArrayListExtra("MUSIC", musics);
11          bindService(intent, this, BIND_AUTO_CREATE);
12          initBottomBar();                                    //初始化底部功能栏
13          setMusicList();                                     //设置音乐列表
14          setBottomBar();                                     //设置底部功能栏
15      } else {
16          ToastUtils.Short(this, "本地没有歌曲！");
17      }
18  }
```

（5）在 SearchFile 类中扫描手机中所有的音频，每扫描出一个音频，就创建一个 Music 对象，并将音频的歌名、歌手和文件路径存入创建的 Music 对象中，然后将 Music 对象存入数据集合中。代码如下：

代码位置：MusicPlayer\app\src\main\java\com\mingrisoft\musicplayer\utils\SearchFile.java

```
01  /**
02   * 扫描本地歌曲
03   * @param context
04   * @return
05   */
06  public static ArrayList<Music> searchLocaltion(Context context){
```

```
    ArrayList<Music> musicList = null;
    //扫描本地的歌曲
    Cursor cursor = context .getApplicationContext()
                            .getContentResolver()
                            .query(MediaStore.Audio.Media.EXTERNAL_CONTENT_URI,
                                    null, null, null,null);
    if(cursor!=null){
        musicList=new ArrayList<>();                        //该集合用来存放歌曲的信息
        while(cursor.moveToNext() ){                        //游标下移
            Music music;
            music=new Music();
            //获取歌曲名称
            String music_name=cursor.getString(
                    cursor.getColumnIndex(MediaStore.Audio.Media.TITLE));
            //获取歌手名称
            String music_singer=cursor.getString(
                    cursor.getColumnIndex(MediaStore.Audio.Media.ARTIST));
            //获取歌曲路径
            String music_path=cursor.getString(
                    cursor.getColumnIndex(MediaStore.Audio.Media.DATA));
            //获取音乐ID
            long music_id = cursor.getLong(cursor
                    .getColumnIndex(MediaStore.Audio.Media._ID));
            //获取专辑
            String music_album = cursor.getString(cursor
                    .getColumnIndex(MediaStore.Audio.Media.ALBUM));
            //获取专辑ID
            int music_albumId = cursor.getInt(cursor
                    .getColumnIndex(MediaStore.Audio.Media.ALBUM_ID));
            //是否为音乐
            int isMusic = cursor.getInt(cursor
                    .getColumnIndex(MediaStore.Audio.Media.IS_MUSIC));
            if (isMusic != 0){                              //如果是音乐保存到集合中
                music.setMusic_name(music_name);
                music.setMusic_singer(music_singer);
                music.setMusic_path(music_path);
                music.setMusic_id(music_id);
                music.setMusic_album(music_album);
                music.setMusic_albumId(music_albumId);
                musicList.add(music);
            }
        }
    }
    cursor.close();                                         //关闭游标
    return musicList;
}
```

明日科技

（6）创建 PlayMusicService 类用来控制音频的播放和停止。在该类中创建 playMusic() 方法，用来准备播放的资源，然后同步资源，关键代码如下：

代码位置：MusicPlayer\app\src\main\java\com\mingrisoft\MusicPlayer\service\PlayMusicService.java

```
/**
 * 播放音乐
 */
private void playMusic(){
    try {
        mediaPlayer.reset();        //重置播放器
        //获取歌曲对象
        Music music = musicList.get(nowPlayPosition);
        //获取歌曲的路径
        String musicPath = music.getMusic_path();
        //设置播放歌曲
        mediaPlayer.setDataSource(musicPath);
        //字符串替换
        String lrcPath = musicPath.replace(".mp3",".lrc");
        //查找歌词
        lrcView.loadLrc(SearchFile.getLrcText(lrcPath));
        isPlaying = true;
        //用于更新歌词
        thread = new Thread(runnable);
        thread.start();
        //设置播放模式为播放音乐
        mediaPlayer.setAudioStreamType(AudioManager.STREAM_MUSIC);
        //同步资源
        mediaPlayer.prepare();
    } catch (IOException e) {
        e.printStackTrace();
    }
}
```

（7）调用 MediaPlayer 的 start() 方法播放音乐，代码如下：

```
/**
 * 同步资源后播放音乐
 * @param mp
 */
@Override
public void onPrepared(MediaPlayer mp) {
    mp.start();
}
```

明日科技

扩展学习

请尝试将音乐列表的分割线更改颜色，同时注意该音乐列表是使用 RecyclerView 实现的。

实例 090	录音机 应用场景：录音	源码位置：光盘\Code\10\090

实例说明

本实例将使用 MediaRecorder 类录制视频，同时为实例添加播放录制音频的功能。运行实例，将显示如图 10.5 所示的正在录音的效果；如图 10.6 所示，当单击“播放录音”后，按钮变为停止播放，同时弹出 Toast 提示框。

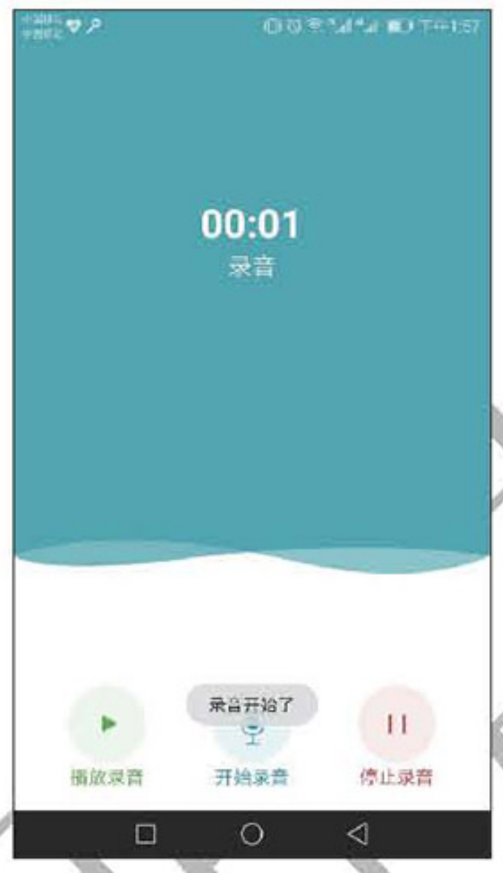


图 10.5　正在录制音频

图 10.6　播放录制的音频

关键技术

关键技术

本实例主要使用 MediaRecorder 类中的 prepare() 方法准备录制，使用 start() 方法开启录制音频，使用 stop() 方法停止录制，使用 setAudioSource() 方法设置录制源，使用 setOutputFormat() 方法设置文件格式，使用 setOutputFile() 方法设置文件输入路径，使用 setAudioEncoder() 方法设置编码等。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“Recorder”，公司域名为“mingrisoft. com”。

（2）导入资源文件：复制“光盘\Code\10\090\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）在 MainActivitystartRecord() 方法中添加设置音频的输出路径，并设置录制音频的相关参数，完成初始化操作，然后开始录制音频。代码如下：

代码位置：Recorder\app\src\main\java\com\mingrisoft\recorder\MainActivity.java

```
/**
 * 开始录音
 */
private void startRecord() {
    if (isSound && !isRecord){
        Toast.makeText(this, "录音播放没有结束", Toast.LENGTH_SHORT).show();
        return;
    }
    //设置SD卡的路径
    fileName = Environment.getExternalStorageDirectory().getAbsolutePath();
    fileName = fileName + File.separator + System.currentTimeMillis()+"mine.arm";
    chronometer.setBase(SystemClock.elapsedRealtime());                    //设置计时器的参考时间
    chronometer.start();                                                   //开启计时器
    mRecorder = new MediaRecorder();                                       //该类用来记录音频或视频
    mRecorder.setAudioSource(MediaRecorder.AudioSource.MIC);               //设置通过麦克风记录音频
    mRecorder.setOutputFormat(MediaRecorder.OutputFormat.DEFAULT);         //设置文件格式为默认格式
    mRecorder.setOutputFile(fileName);                                     //设置文件输出路径
    mRecorder.setAudioEncoder(MediaRecorder.AudioEncoder.DEFAULT);         //设置编码为默认值
    try {
        mRecorder.prepare();
    } catch (IOException e) {
        Log.e("startRecord", "prepare() failed");
    }
    mRecorder.start();                                                     //开始录制
    isRecord = true;
    Toast.makeText(this, "录音开始了", Toast.LENGTH_SHORT).show();
}
```

（4）在 MainActivity 类中添加 stopRecord() 方法，用来执行停止录制音频的操作。代码如下：

```
/**
 * 停止录音
 */
private void stopRecord() {
    if (isSound && !isRecord){
        Toast.makeText(this, "录音播放没有结束", Toast.LENGTH_SHORT).show();
        return;
    }
    if(!isRecord){
        return;
    }
```

```
        mRecorder.stop();                                          //停止录制
        mRecorder.release();                                       //释放资源
        mRecorder = null;
        chronometer.stop();                                        //停止计时器
        isRecord = false;
        Toast.makeText(this, "录音停止了", Toast.LENGTH_SHORT).show();
    }
```

（5）在 MainActivity 类中添加 startSound () 方法，实现停止录制音频后播放录制的音频，并设置切换按钮的效果。代码如下：

```
/**
 * 播放录音
 */
private void startSound() {
    if (fileName == null){
        Toast.makeText(this, "当前没有录音", Toast.LENGTH_SHORT).show();
        return;
    }
    if (isRecord && !isSound){
        Toast.makeText(this, "录音没有结束", Toast.LENGTH_SHORT).show();
        return;
    }
    chronometer.setBase(SystemClock.elapsedRealtime());
    chronometer.start();
    mPlayer = new MediaPlayer();                                   //获取播放器对象
    mPlayer.setOnCompletionListener(new MediaPlayer.OnCompletionListener() {
        @Override
        public void onCompletion(MediaPlayer mp) {
            chronometer.stop();                                    //停止计时
            Toast.makeText(MainActivity.this, "播放结束", Toast.LENGTH_SHORT).show();
        }
    });
    try {
        mPlayer.setDataSource(fileName);                           //设置播放文件
        mPlayer.prepare();                                         //同步资源
        mPlayer.start();                                           //播放音频
    } catch (IOException e) {
        Log.e("startSound", "播放失败");
    }
    isSound = true;
    play.setVisibility(View.GONE);                                 //隐藏播放按钮
    cancel.setVisibility(View.VISIBLE);                            //显示停止按钮
    Toast.makeText(this, "播放开始了", Toast.LENGTH_SHORT).show();
}
```

明日科技

扩展学习

请尝试将底部按键栏的颜色更改成与状态栏一样的视觉效果。

实例 091	分贝测试仪 应用场景：话筒监制	源码位置：光盘\Code\10\091

实例说明

本实例将通过 AudioRecord 类实现分贝测试仪的运行效果。运行实例，初始效果如图 10.7 所示；单击“START”按钮，启动分贝测试功能，当检测出声音强度的变化后，将检测到的数值显示到仪表盘中，如图 10.8 所示。

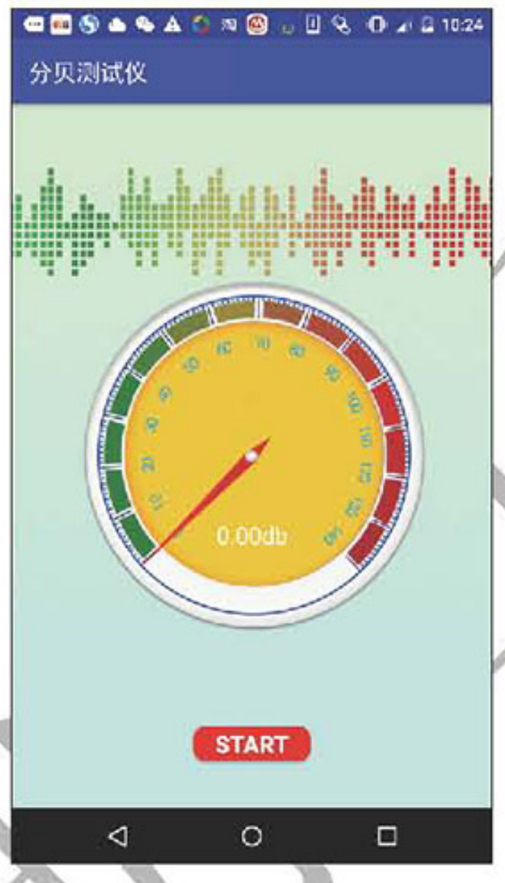


图 10.7　未开启测试分贝效果

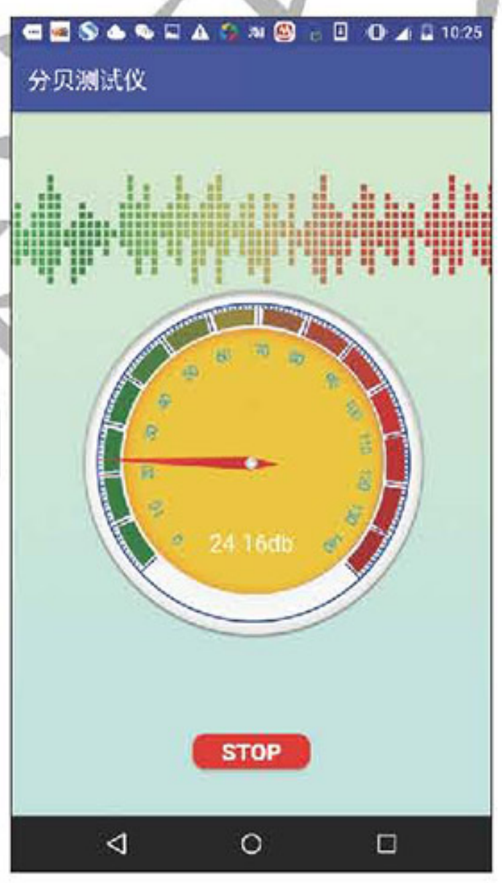


图 10.8　开启测试分贝效果

关键技术

关键技术

本实例主要使用 AudioRecord 类的 startRecording() 方法，开启声音的记录功能，并使用 read() 方法读取一段音频流。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“Decibel”，公司域名为“mingrisoft. com”。

（2）导入资源文件：复制“光盘\Code\10\091\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）创建话筒管理类 AudioRecordManger，用来开启对话筒的监听（代码位置：Decibel\app\src\

main\java\com\mingrisoft\decibel\AudioRecordManger.java）。

（4）创建自定义表盘控件 DialPlateView 类，用来显示分贝测试仪表盘的刻度（代码位置：Decibel\app\src\main\java\com\mingrisoft\decibel\DialPlateView.java）。

（5）在 MainActivity 类中添加 onClick 单击事件，实现单击按钮开启分贝测试功能。代码如下：

代码位置：Decibel\app\src\main\java\com\mingrisoft\decibel\MainActivity.java

```
@Override
public void onClick(View v) {
    //如果没有开启测试则退出方法
    if (isStart){
        AudioRecordManger.isGetVoiceRun = false;
        switchs.setText("START");
        isStart = false;
        Toast.makeText(this, "分贝测试结束了", Toast.LENGTH_SHORT).show();
        return;
    }
    //开启权限申请
    if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.M){
        requestPermission(new String[]{Manifest.permission.RECORD_AUDIO},1000);
    }else {
        startTest();                                    //开启测试
    }
}
```

（6）在 MainActivity 类中添加 startTest() 方法，用来开启测试分贝。代码如下：

```
/**
 * 开启分贝测试
 */
private void startTest(){
    audioRecordManger.getNoiseLevel();
    switchs.setText("STOP");
    isStart = true;
    Toast.makeText(this, "分贝测试开始了", Toast.LENGTH_SHORT).show();
}
```

在 MainActivity 类中添加 Hanlder 对象，用来接收分贝值，并将分贝值展示在界面上。代码如下：

```
private Handler handler = new Handler(new Handler.Callback() {
    @Override
    public boolean handleMessage(Message msg) {
        if (msg.what == 1){
            double values = (double) msg.obj;           //获取分贝值
            float db = (float) values;                  //类型转换
            dialPlateView.upDataValues(db);             //更新控件显示
        }
```

```
09            return false;
10        }
11    });
```

扩展学习

请尝试更改表盘中字体的颜色为红色。

实例 092　调用本地播放器播放视频

应用场景：播放视频

源码位置：光盘\Code\10\092

实例说明

在本实例将使用 Android 自带的控件 VideoView 实现播放视频功能。运行实例，当加载视频资源结束后自动播放视频，将显示如图 10.9 所示的界面。

图 10.9　VideoView 播放视频的效果

关键技术

关键技术

本实例使用 VideoView 类中的 setVideoURI() 方法（用于设置播放的视频）、setMediaController() 方法（用于设置控制器）、start() 方法（用于播放），以及 MediaController 类的 setMediaPlayer() 方法（用于实现绑定播放器）。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“VedioPlayer”，公司域名为“mingrisoft.com”。

（2）导入资源文件：复制“光盘\Code\10\092\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）在 MainActivity 类的 onCreate() 方法中，实例化 VideoView 控件和 MediaController 控制器对象，然后设置 VideoView 的播放资源，并让 MediaController 绑定 VideoView。最后播放视频。代码如下：

代码位置：VedioPlayer\app\src\main\java\com\mingrisoft\vedioplayer\MainActivity.java

```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    //设置全屏
    getWindow().addFlags(WindowManager.LayoutParams.FLAG_FULLSCREEN);
    setContentView(R.layout.activity_main);
    //初始化控件
    video = (VideoView) findViewById(R.id.video);
    //实例化控制器
    controller = new MediaController(this);
    //设置播放文件路径
    String uri = "android.resource://" + getPackageName() + "/" + R.raw.test;
    //设置播放资源
    video.setVideoURI(Uri.parse(uri));
    //设置控制器
    video.setMediaController(controller);
    //给控制器设置播放器控件
    controller.setMediaPlayer(video);
    video.requestFocus();                    //获取焦点
    //设置视频加载回调
    video.setOnPreparedListener(new MediaPlayer.OnPreparedListener() {
        @Override
        public void onPrepared(MediaPlayer mp) {
            //播放视频
            video.start();
        }
    });
}
```

扩展学习

请尝试通过设置界面的属性，使播放的视频全屏显示在手机屏幕中。

实例 093　带弹幕的视频播放器

应用场景：播放视频

源码位置：光盘\Code\10\093

实例说明

本实例将实现带弹幕的视频的播放器，运行实例，将显示 10.10 所示的播放效果。

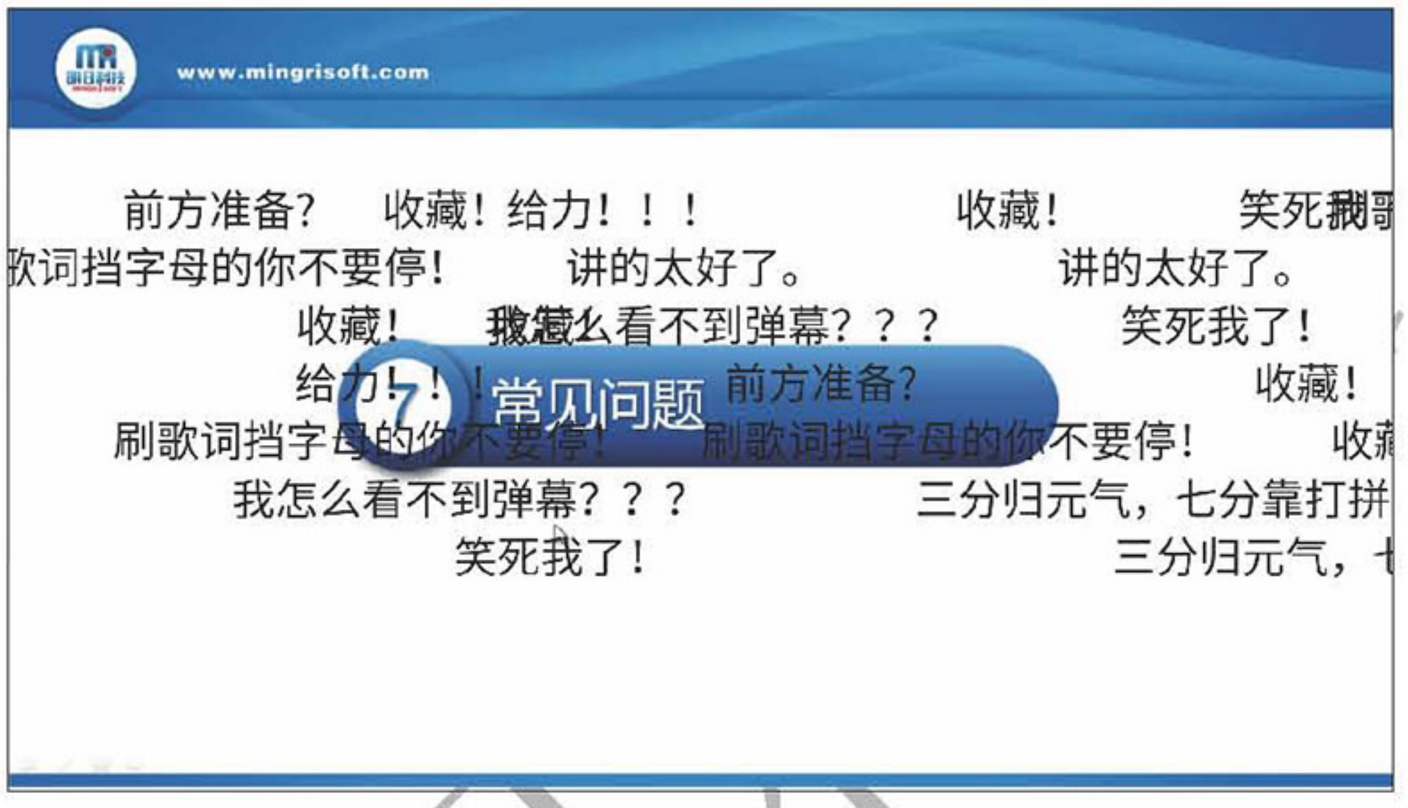


图 10.10　播放视频时的效果

关键技术

关键技术

本实例主要使用第三方视频播放器框架 Vitamio（Vitamio 官网：https://www.vitamio.org/），和哔哩哔哩的开源弹幕框架 DanmakuFlameMaster（项目地址：http://github.com/ctiao/DanmakuFlameMaster）。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“Barrage”，公司域名为“mingrisoft. com”。

（2）导入资源文件：复制“光盘\Code\10\093\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）在 MainActivity 类的 onCreate() 方法中，实例化 VideoView 控件和 MediaController 控制器对象，然后设置 VideoView 的播放资源和 VideoView 的控制器，并让 MediaController 绑定 VideoView。最后播放视频。代码如下：

代码位置：Barrage\app\src\main\java\com\mingrisoft\barrage\MainActivity.java

```
01  videoView = (VideoView) findViewById(R.id.video);
02  controllerHome = (FrameLayout) findViewById(R.id.home);
```

```
hint = (ImageView) findViewById(R.id.hint);
Log.w("生命周期", "onCreate: ");
if (!Vitamio.isInitialized(this)) {
    Log.e("Vitamio", "没有被初始化！");
    return;
}
//播放控制器
controller = new MyController(this, true, controllerHome);
barragge = new MyBarraggeSwitch();
controller.setBarrageSwitchListener(barragge);
videoView.setVideoURI(Uri.parse("http://video.mingrisoft.com/5571553ac28cb.mp4"));
videoView.setMediaController(controller);                    //设置控制器
videoView.requestFocus();
videoView.setOnPreparedListener(new MediaPlayer.OnPreparedListener() {
    @Override
    public void onPrepared(MediaPlayer mp) {
        videoView.start();
    }
});
```

（4）在 MainActivity 类中增加 addDanmaku() 方法，用来添加弹幕信息，并设置弹幕的效果。代码如下：

```
/**
 * 向弹幕View中添加一条弹幕
 */
private void addDanmaku(String content) {
    BaseDanmaku danmaku = danmakuContext.mDanmakuFactory
                                .createDanmaku(BaseDanmaku.TYPE_SCROLL_RL);
    danmaku.text = content;
    danmaku.padding = 5;                                     //弹幕内容
    danmaku.textSize = sp2px(20);                            //字体大小
    danmaku.textColor = Color.BLACK;                         //字体颜色
    danmaku.setTime(danmakuView.getCurrentTime());           //发布时间
    danmakuView.addDanmaku(danmaku);
}
```

（5）在 MainAvtivity 类中添加 changeSound() 和 changeBright() 方法，用来根据手势调节屏幕的亮度和播放视频时的音量大小。代码如下：

```
/**
 * 调节声音
 *
 * @param values
 */
private void changeSound(float values) {
    nowSoundValues = manager.getStreamVolume(AudioManager.STREAM_MUSIC);
    if (nowSoundValues < 0) {
```

```
09          nowSoundValues = 0;
10      }
11      int index = (int) (values * maxSoundValues) + nowSoundValues;
12      if (index > maxSoundValues) {
13          index = maxSoundValues;
14      } else if (index < 0) {
15          index = 0;
16      }
17      manager.setStreamVolume(AudioManager.STREAM_MUSIC, index, 0);
18  }
19
20  /**
21   * 调节亮度
22   *
23   * @param values
24   */
25  private void changeBright(float values) {
26      nowScreenBright = getWindow().getAttributes().screenBrightness;
27      if (nowScreenBright <= 0.00f || nowScreenBright < 0.01f) {
28          nowScreenBright = 0.50f;
29      }
30      WindowManager.LayoutParams lpa = getWindow().getAttributes();
31      lpa.screenBrightness = nowScreenBright + values;
32      if (lpa.screenBrightness > 1.0f)
33          lpa.screenBrightness = 1.0f;
34      else if (lpa.screenBrightness < 0.01f)
35          lpa.screenBrightness = 0.01f;
36      getWindow().setAttributes(lpa);
37  }
```

扩展学习

有时因为播放的视频太亮，导致看不清弹幕的效果，请尝试更改弹幕的字体颜色。

实例 094　手机屏幕录制

应用场景：手机游戏直播

源码位置：光盘\Code\10\094

视频位置：光盘\Video\10\094

实例说明

手机游戏直播中需要将玩家手机屏幕内的游戏进行录制，本实例将实现一个手机屏幕录制的小程序。运行实例，首先开启录制音频与存储权限，如图 10.11 所示；然后单击如图 10.12 所示的“开始录制”按钮，将弹出截取屏幕显示内容对话框，如图 10.13 所示；最后单击“立即开始”按钮后，再单击“主屏幕”按钮，即可开启其他应用程序进行屏幕的录制功能。

图 10.11　录制视频

图 10.12　视频的存储位置

图 10.13　截取屏幕内容

关键技术

本实例主要使用 MediaRecorder 类中的 setAudioSource() 方法、setVideoFrameRate() 方法、setVideoSize() 方法、setOutputFile() 方法、setVideoSource() 方法，实现录制音频并设置视频格式。

说明：由于模拟器占用内存空间较大，所以建议使用真机来运行本实例。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“ScreenRecord”，公司域名为“com.mingrisoft”。

（2）首先在 AndroidManifest 中添加一些相关的权限，主要代码如下：

```
<!--文件的写入权限-->
<uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE" />
<!--录音权限-->
<uses-permission android:name="android.permission.RECORD_AUDIO" />
```

（3）在 MainActivity 类中首先创建 ServiceConnection 对象并实现 onServiceConnected() 与 onServiceDisconnected() 方法，然后在 onServiceConnected() 方法中实现与后台录制服务的通讯。关键代码如下：

```
private ServiceConnection connection = new ServiceConnection() {
    @Override
    public void onServiceConnected(ComponentName className, IBinder service) {
        //创建该对象用于获取屏幕的宽度与高度（像素）
        DisplayMetrics metrics = new DisplayMetrics();
        //获取屏幕像素
```

```
        getWindowManager().getDefaultDisplay().getMetrics(metrics);
        //创建录制服务对象
        RecordService.RecordBinder binder = (RecordService.RecordBinder) service;
        //获取后台录制服务信息
        recordService = binder.getRecordService();
        //设置录像参数
        recordService.setConfig(metrics.widthPixels, metrics.heightPixels, metrics.densityDpi);
        startBtn.setEnabled(true);          //设置录制按钮可用状态
        //根据录屏服务状态设置按钮文字，服务开启时设置文字为停止录屏
        startBtn.setText(recordService.isRunning() ? R.string.stop_record : R.string.start_record);
    }
    @Override
    public void onServiceDisconnected(ComponentName arg0) {}
};
```

（4）在 MainActivity 类中对“开始录制”按钮进行初始化，并添加单击事件。

（5）在 RecordService 类中对录制视频时设置相关的参数，包括视频大小、视频路径、视频帧率等，主要代码如下：

```
//设置音频源（配音类的音频源主要源于文件）
mediaRecorder.setAudioSource(MediaRecorder.AudioSource.MIC);
//设置视频源：Surface和Camera 两种
mediaRecorder.setVideoSource(MediaRecorder.VideoSource.SURFACE);
//设置视频输出格式
mediaRecorder.setOutputFormat(MediaRecorder.OutputFormat.MPEG_4);
//设置视频编码格式
mediaRecorder.setVideoEncoder(MediaRecorder.VideoEncoder.H264);
//设置音频编码格式
mediaRecorder.setAudioEncoder(MediaRecorder.AudioEncoder.AMR_NB);
//设置视频输出路径
mediaRecorder.setOutputFile(getsaveDirectory() + System.currentTimeMillis() + ".mp4");
//设置视频尺寸大小
mediaRecorder.setVideoSize(width, height);
//设置视频编码的码率，码率越大 越接近真实的视频
mediaRecorder.setVideoEncodingBitRate(5 * 1024 * 1024);
//设置视频编码的帧率，帧数越高，所占内存越大
mediaRecorder.setVideoFrameRate(30);
try {
    mediaRecorder.prepare();
} catch (IOException e) {
    e.printStackTrace();
}
```

扩展学习

实现在单击“录制视频”的按钮时，弹出视频所保存的具体位置。

实例 095 仿微信录制视频

应用场景：视频录制

源码位置：光盘\Code\10\095

实例说明

仿微信录制视频是通过调用手机摄像头与自定义进度条实现的。运行实例，长按“按住拍”按钮进行视频的录制，如图 10.14 所示；手指向上滑动松开将取消视频录制，如图 10.15 所示；进度条结束后抬起手指将弹出视频录制完成提示框，如图 10.16 所示。

图 10.14　进行视频录制

图 10.15　取消视频录制

图 10.16　视频录制完成

关键技术

关键技术

本实例主要使用 File 类中的 mkdirs() 方法、createTempFile() 方法、MediaRecorder 类的 start() 方法和 stop() 方法，实现了与仿微信录制视频的功能。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“WeChatVideo”，公司域名为“mingrisoft. com”。

（2）导入资源文件：复制“光盘\Code\10\095\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）在“app\src\main\java”目录中创建 view 包，在该包中首先创建 RecordProgress 类用于实现拍摄视频时的进度条，然后创建 RecordView 类，通过该类中的 createRecordDir() 方法创建录制视频的目录与视频文件的名称。代码如下：

```
01  private void createRecordDir() {
02      //录制视频的保存文件夹
03      File sampleDir = new File(outputDirPath);                    //录制视频的保存地址
04      if (!sampleDir.exists()) {                                   //如果没有这个目录
05          sampleDir.mkdirs();                                      //创建目录
06      }
07      try {
08          //创建视频输出的目录与名称
09          vecordFile = File.createTempFile("recording", suffix, sampleDir);
10      } catch (IOException e) {
11          e.printStackTrace();
12      }
13  }
```

（4）通过 record() 方法实现视频的录制功能，代码如下：

```
01  public void record(final OnRecordStausChangeListener onRecordStausChangeListener) {
02      this.onRecordStausChangeListener = onRecordStausChangeListener;
03      createRecordDir();                                           //创建视频保存的路径
04      try {
05          if (onRecordStausChangeListener != null)
06              onRecordStausChangeListener.onRecordStart();         //开启录制回调接口
07          if (!isOpenCamera)                                       //如果未打开摄像头，则打开
08              openCamera();                                        //打开摄像头
09          initRecord();                                            //进行录制初始化
10          timeCount = 0;                                           //时间计数器重新赋值
11          timer = new Timer();                                     //创建计时器
12          timer.schedule(new TimerTask() {
13              @Override
14              public void run() {
15                  timeCount++;                                     //加1秒
16                  if (onRecordStausChangeListener != null)
17                      //设置拍摄的最大时间
18                      onRecordStausChangeListener.onRecording(timeCount, recordMaxTime);
19                  if (timeCount == recordMaxTime) {                //达到指定时间，停止拍摄
20                      stop();                                      //停止录制
21                  }
22              }
23          }, 0, 1000);                                             //1秒执行一次
24      } catch (IOException e) {
25          e.printStackTrace();
26      }
27  }
```

（5）打开 MainActivity 类，在该类中首先获取相关控件，然后通过 onRecrodFinish() 方法实现视频录制完成的工作。代码如下：

```
01  public void onRecrodFinish() {
```

```
02      runOnUiThread(new Runnable() {                    //在UI线程上运行指定的操作
03          @Override
04          public void run() {
05              //隐藏上滑取消提示文字与松开取消提示文字
06              tvUp.setVisibility(View.GONE);
07              tvRelease.setVisibility(View.GONE);
08              resetVideoRecord();                       //停止录制（释放相机后重新打开相机）
09              Toast.makeText(MainActivity.this, "录制成功！",
10                      Toast.LENGTH_SHORT).show();
11          }
12      });
13  }
```

扩展学习

本实例实现了仿微信录制视频的功能。请尝试在“app\src\main\java\com.mingrisoft.wechatvideo”包中创建 VideoPlayActivity 类，显示视频播放的界面，然后通过在 MainActivity 类的 onRecrodFinish() 方法中设置录制界面与播放界面的跳转，实现录制完成后进行视频播放的运行效果。

实例 096　播放视频对话框

应用场景：弹出视频广告

源码位置：光盘\Code\10\096

实例说明

在对话框中播放视频可以应用在播放视频广告的应用中。本实例将实现一个播放宣传片的视频对话框。运行实例，初始效果如图 10.17 所示；单击显示视频对话框按钮，将显示对话框播放视频的界面，如图 10.18 所示。

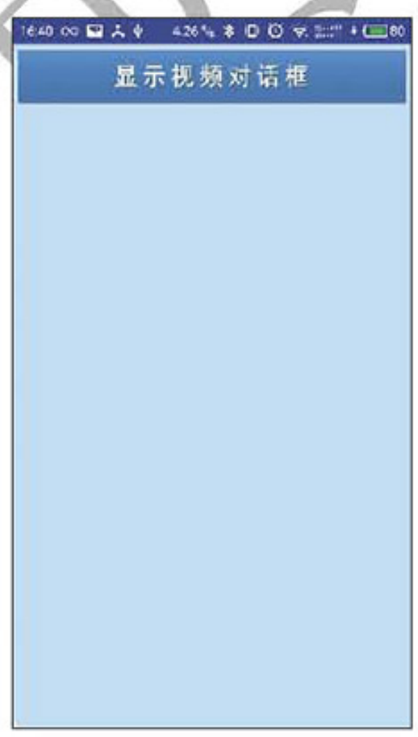


图 10.17　单击显示对话框按钮

图 10.18　显示视频对话框

关键技术

本实例主要使用 SurfaceView 类中的 getHolder() 方法、SurfaceHolder 类中的 setKeepScreenOn() 方法和 MediaPlayer 类中的 setDataSource() 方法，实现视频对话框的效果。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“VideoDialog”，公司域名为“mingrisoft. com”。

（2）导入资源文件：复制“光盘\Code\10\096\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）在 app\res\layout 目录中创建一个名称为“video_play_dialog_layout.xml”的布局文件，并在该文件中添加 SurfaceView 控件用于显示视频的图像。（代码位置：app\res\layout\video_play_dialog_layout.xml）。

（4）在 java 目录下的 com.mingrisoft.videodialog 包中创建名称为“VideoDialog”的 Java 类，该类继承自 Dialog，用于创建自定义的对话框。（代码位置：app\java\com.mingrisoft.videodialog\VideoDialog）

（5）打开 MainActivity 类，首先在该类中创建 playVideo() 方法，然后在该方法中进行播放视频的初始化工作。代码如下：

```
01  //创建播放器
02  mediaPlayer = new MediaPlayer();
03  //获取surfaceHolder
04  surfaceHolder = surfaceView.getHolder();
05  //设置播放时打开屏幕
06  surfaceHolder.setKeepScreenOn(true);
07  //设置音频类型
08  mediaPlayer.setAudioStreamType(AudioManager.STREAM_MUSIC);
09  try {
10      //设置需要播放的视频路径，本实例中将视频保存在手机SD卡的根目录下，video.mp4为视频名称
11      mediaPlayer.setDataSource(Environment.getExternalStorageDirectory()+"/video.mp4");
12  } catch (IOException e) {
13      e.printStackTrace();
14  }
15  try {
16      //准备
17      mediaPlayer.prepare();
18  } catch (IllegalStateException e1) {
19      e1.printStackTrace();
20  } catch (IOException e1) {
21      e1.printStackTrace();
22  }
```

通过 setOnPreparedListener() 方法实现视频显示的位置与视频的播放功能。代码如下：

```
mediaPlayer.setOnPreparedListener(new MediaPlayer.OnPreparedListener() {
    @Override
    public void onPrepared(MediaPlayer mp) {
        //把视频画面输出到SurfaceView
        mediaPlayer.setDisplay(surfaceHolder);
        mediaPlayer.start();                          //播放视频
    }
});
```

通过 setOnCompletionListener() 方法实现视频播放完成后的操作。代码如下：

```
mediaPlayer.setOnCompletionListener(new MediaPlayer.OnCompletionListener() {
    @Override
    public void onCompletion(MediaPlayer mp) {
        if (mediaPlayer != null)
            mediaPlayer.release();                    //重置mediaplayer等待下一次播放
        if (videoDialog.isShowing())
            videoDialog.dismiss();                    //关闭对话框
    }
});
```

（6）将需要播放的视频资源文件复制在手机或者模拟器的 SD 卡的根目录下。注意视频名称需要与代码中设置的视频名称相同。

（7）首先打开“app\manifests”节点下的 AndroidManifest.xml 文件，在该文件中添加 SD 卡的读写权限，然后在 MainActivity 类中填写获取 SD 卡读写权限的代码。

扩展学习

本实例实现了在对话框中视频播放。尝试通过修改对话框布局文件的布局方式，以及添加 TextView 控件来为视频对话框添加标题。

明日科技

第11章 语音识别

实例097 会聊天的机器人

应用场景：智能家居

源码位置：光盘\Code\11\097

视频位置：光盘\Video\11\097

视频讲解

实例说明

本实例将实现一款会聊天的机器人。这款机器人可以模拟人脑的思维与用户进行对话，即通过默认打开网页，显示相应的内容。运行实例，将显示正在初始化的机器人，如图11.1所示；初始化完毕后，将显示正在等待指令的机器人，如图11.2所示；指令识别成功后机器人开始说话，如图11.3所示。

图11.1 正在初始化机器人

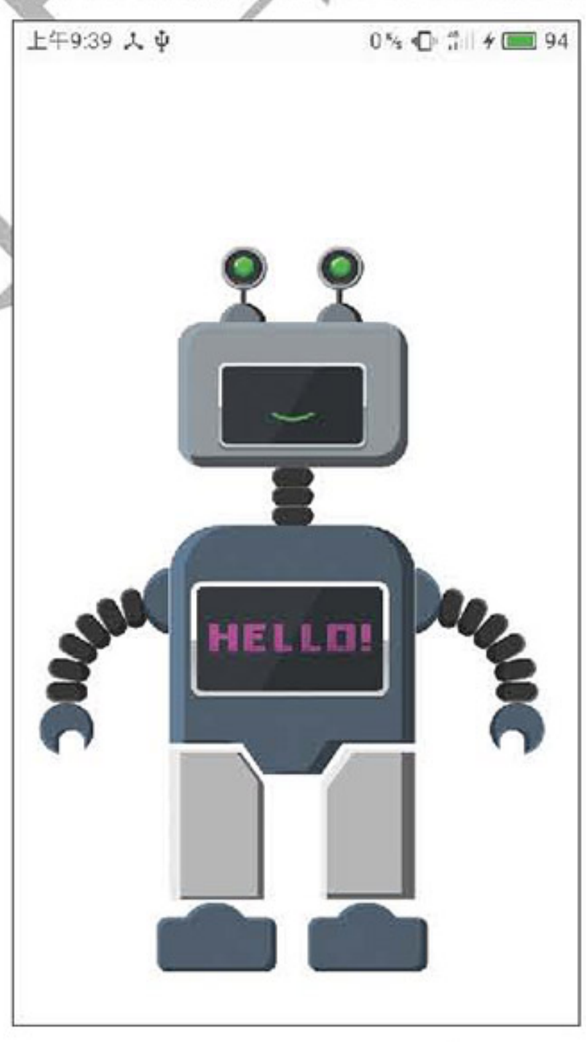


图11.2 正在等待指令的机器人

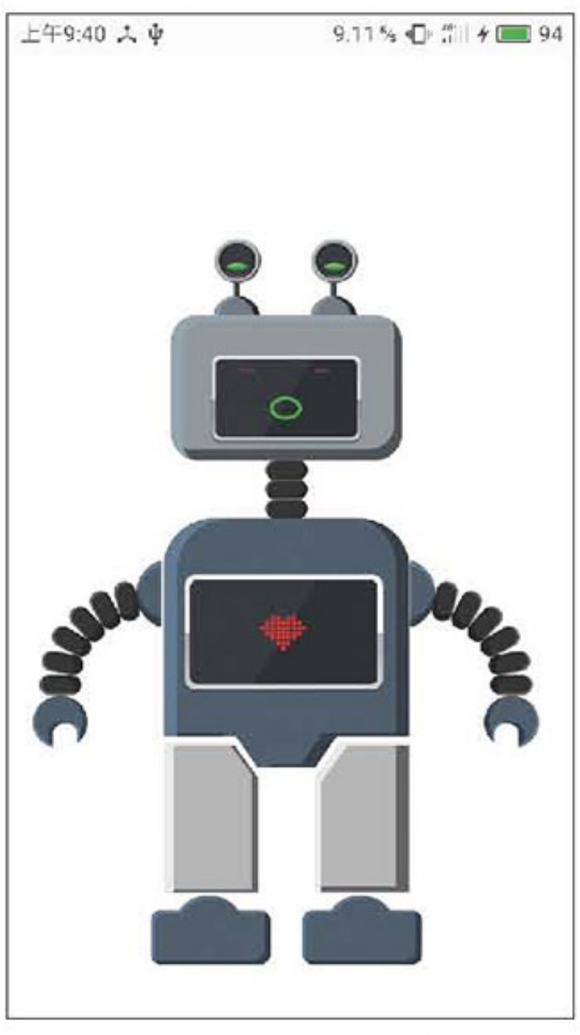


图11.3 正在说话的机器人

关键技术

本实例主要使用 WebView 类中的 loadUrl() 方法，RequestManager 类中的 load() 方法，实现加载指定的网址以及加载动态图。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“QRobot”，公司域名为“mingrisoft.com”。

（2）在“app\libs”文件夹中导入相关的 Jar 包（资源位置：光盘\Code\11\097\src），用于获取与解析网络数据。

（3）在“app\res\drawable”文件夹中导入相关的动态图，用于展现机器人说话与思考时的状态。

（4）在“app\res\values\styles”文件中将“DarkActionBar”修改为“NoActionBar”，用于去掉系统默认的标题栏。

（5）在 MainActivity 类中，实现首次进入界面的时候，加载两张动态图片，一张是等待指令的机器人动态图片，一张是说话时候的动态图片，关键代码如下：

```
//加载微笑机器人的图片
robotImOne = (ImageView) findViewById(R.id.robot_image_one);
//加载说话机器人的图片
robotImTwo = (ImageView) findViewById(R.id.robot_image_two);
//把微笑动态图加载到图片上
Glide.with(MainActivity.this).load(R.drawable.robot_a).into(robotImOne);
//把说话的动态图加载到图片上
Glide.with(MainActivity.this).load(R.drawable.robot_b).into(robotImTwo);
```

（6）加载机器人的动态图片是一个较为耗时的操作，所以在刚进入这个界面的时候，弹出一个加载等待条，显示时间为 4 秒，关键代码如下：

```
pd.setProgressStyle(R.style.progress);                    //设置等待条的风格
pd.setMessage("正在初始化机器人....");                      //等待动画的标题
pd.show();                                                //显示等待动画

new Thread(new Runnable() {
    public void run() {
        //在此处休眠4秒
        try {
            Thread.sleep(4000);                           //在此处休眠4秒
        } catch (InterruptedException e) {
        }
        /**
         * 4秒之后
         **/
        pd.dismiss();                                     //等待条消失
    }
}).start();                                               //开启线程
```

明日科技

（7）当用户与机器人交流时，向聚合数据进行请求，并解析返回的数据，再根据返回的数据类型进行相应的操作。关键代码如下：

```
String path = url + msg + key;                                          //进行数据请求
StringRequest request = new StringRequest(path, new Response.Listener<String>() {
    @Override
    public void onResponse(Request<String> request, String s) {
        Log.e("-------", s);
        Gson gson = new Gson();                                         //Gson解析数据
        UrlEntity urlEntity = gson.fromJson(s, UrlEntity.class);        //解析数据的实体类
        if (urlEntity.getResult() == null ||
                urlEntity.getResult().getUrl() == null) {               //如果返回的数据没有网址
            Log.e("-----", "没有网址的");
            myHandler.obtainMessage(SPEECH_START,
                                    urlEntity.getResult().getText()).sendToTarget();
        } else {                                                        //返回的数据有网址
            Log.e("-----", "有网址的");
            myHandler.obtainMessage(SPEECH_START,
                                    urlEntity.getResult().getText()).sendToTarget();
            Intent intent = new Intent(MainActivity.this, WebViewActivity.class);//打开WebView
            intent.putExtra("url", urlEntity.getResult().getUrl());     //将网址传递过去
            startActivity(intent);                                      //执行
        }
    }
}, new Response.ErrorListener() {                                       //请求错误执行的操作
    @Override
    public void onErrorResponse(Request request, VolleyError volleyError) {
        Toast.makeText(MainActivity.this, "网络错误", Toast.LENGTH_SHORT).show();
    }
});
```

（8）若要使机器人显示网页，则可以使用 WebView 类，WebView 类主要是在手机上显示网页的控件，关键代码如下：

```
webView = (WebView) findViewById(R.id.web_view);                        //绑定id
Intent intent = getIntent();
String url = intent.getStringExtra("url");                              //接受跳转界面传递过来的网址
webView.getSettings().setJavaScriptEnabled(true);
webView.setWebViewClient(new WebViewClient());
webView.loadUrl(url);                                                   //显示网页
```

扩展学习

请尝试显示与机器人对话返回的结果。

实例 098 语音打电话

应用场景：语音控制拨号给某人

源码位置：光盘\Code\11\098

视频位置：光盘\Video\11\098

实例说明

本实例将实现通过语音打电话，比如说出“打电话给 ×××”，只要通讯录里面存在该联系人，就会拨号给该联系人。运行实例，将显示如图 11.4 所示的通过语音打电话界面；按下屏幕中间的电话按钮，将显示如图 11.5 所示的“正在等待指令”界面；说出“打电话给老爸”，将显示图 11.6 所示的“电话已拨出”界面。

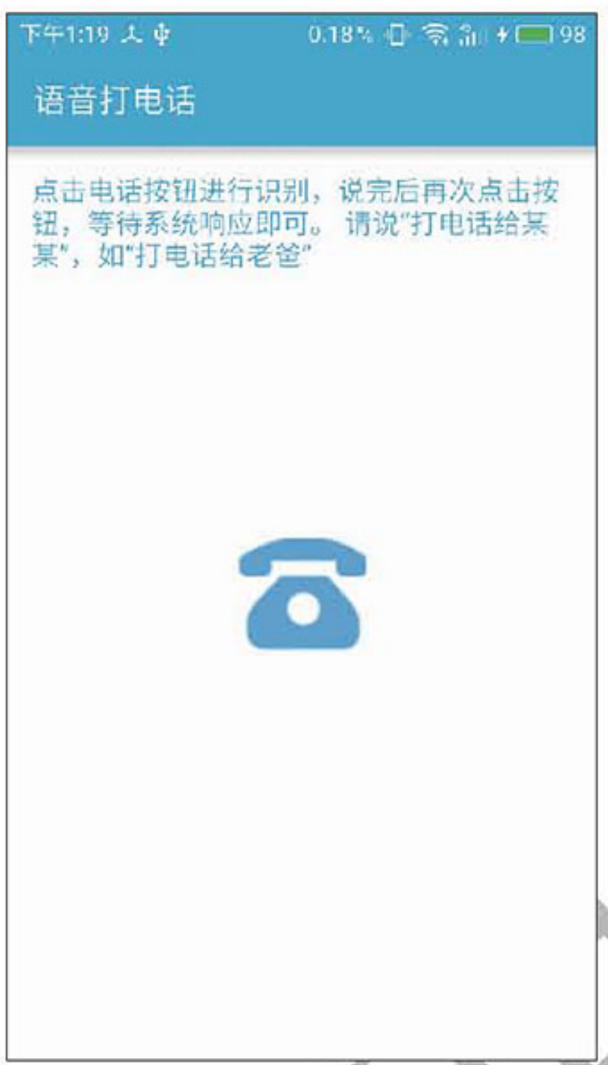


图 11.4 原始界面

图 11.5 等待指令

图 11.6 电话已拨出

关键技术

本实例主要使用 TextUtils 类中的 isEmpty() 方法、Cursor 接口类中的 moveToNext() 方法，实现判断字段是否为空并查询数据库。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“VoicPhone”，公司域名为“mingrisoft.com”。

（2）在科大讯飞官网注册账号，并申请 Appkey。

（3）在“app\libs”文件夹中拷入语音识别的 Jar 包（资源位置：光盘\Code\11\098\src），用于与后台进行交互。

（4）在 MainActivity 类中的 onCreate() 方法中添加申请的 Appkey，代码如下：

```
//appid换成自己申请的
SpeechUser.getUser().login(MainActivity.this, null, null, "appid=573a7bf0", listener);
```

（5）在 MainActivity 类中获取手机中联系人数据，关键代码如下：

```
ContentResolver resolver = getContentResolver();
try {
    //用游标读取手机中的联系人
    Cursor phoneCursor = resolver.query(Phone.CONTENT_URI, PHONES_PROJECTION,
                                        null, null, null);
    if (phoneCursor != null) {
        while (phoneCursor.moveToNext()) {
            //获取手机号码
            String phoneNumber = phoneCursor.getString(1);
            //当手机号码为空跳过当前循环
            if (TextUtils.isEmpty(phoneNumber))
                continue;
            //获取联系人名称
            String contactName = phoneCursor .getString(0);
            PersionEntity mContact = new PersionEntity(contactName, phoneNumber);
            list.add(mContact);                                    //将联系人添加到list集合中
        }
        phoneCursor.close();                                       //关闭流标
    }
} catch (Exception e) {
    e.printStackTrace();
}
```

（6）在 MainActivity 类中设置科大讯飞的通用回调监听接口，关键代码如下：

```
List<SortModel> filterDateList = new ArrayList<SortModel>();      //初始化集合
if (TextUtils.isEmpty(filterStr)) {                                //判断返回的数据是不是为空
    filterDateList = SourceDateList;
} else {
    filterDateList.clear();                                        //清除集合
    for (SortModel sortModel : SourceDateList) {                   //在联系人中遍历
        String name = sortModel.getName();
        if (name.indexOf(filterStr.toString()) != -1 ||
                characterParser.getSelling(name).startsWith(filterStr.toString())) {
            filterDateList.add(sortModel);
        }
    }
}
Collections.sort(filterDateList, pinyinComparator);
if (filterDateList != null && filterDateList.size() > 0) {         //如果集合不为空
    callPhone(filterDateList.get(0).getName());                    //将对应的联系人获取出来
} else {                                                           //如果没有对应的联系人则弹出提示
```

明日科技

```
18        Toast.makeText(this, "没有对应的联系人", Toast.LENGTH_SHORT).show();
19    }
```

（7）当单击话筒时，将获取出来的电话号码进行呼出、调用系统拨打电话，关键代码如下：

```
01  Intent intent = new Intent(Intent.ACTION_CALL);
02  Uri data = Uri.parse("tel:" + list.get(i).getNum());          //获取与联系人对应的电话号
03  intent.setData(data);                                         //设置电话号码
04  startActivity(intent);                                        //开启拨打电话
```

扩展学习

尝试实现在拨打电话时，弹出询问用户是否拨打电话给联系人的对话框。

实例 099　语音电子书

应用场景：App 阅读器

源码位置：光盘\Code\11\099

实例说明

语音电子书 App 阅读器是将文字传递给后台，等待后台处理完成后再返回语音，手机将直接读出语音，从而实现语音电子书的功能。运行实例，将打开语音电子书界面，如图 11.7 所示；单击“朗读电子书”按钮，将弹出“正在缓冲”弹窗，如图 11.8 所示；缓冲结束后，将开始朗读，并显示“正在朗读”的界面，如图 11.9 所示。

图 11.7　原始界面　　图 11.8　正在缓冲　　图 11.9　正在朗读

关键技术

本实例主要使用 TextView 类中的 getText() 方法，获取电子书的内容。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“VoiceBook”，公司域名为“mingrisoft.com”。

（2）在科大讯飞官网注册账号，并申请 Appkey。

（3）在“app\libs”文件夹中粘贴语音识别的 Jar 包（资源位置：光盘 \Code\11\099\src），用于与后台进行交互。

（4）在“app\main”文件夹中创建 assets 文件夹与 jniLibs 文件夹，粘贴科大讯飞相关的文件。

（5）在 MainActivity 类的 onCreate() 方法中添加申请的 Appkey，代码如下：

```
01  //换成自己申请的appid
02  SpeechUser.getUser().login(MainActivity.this, null, null, "appid=573a7bf0", listener);
```

（6）在 MainActivity 类中设置科大讯飞的通用回调监听接口，关键代码如下：

```
01  /**
02   * 通用回调接口
03   */
04  private SpeechListener listener = new SpeechListener() {
05      //消息回调
06      @Override
07      public void onEvent(int arg0, Bundle arg1) {
08          //TODO Auto-generated method stub
09      }
10      //数据回调
11      @Override
12      public void onData(byte[] arg0) {
13          //TODO Auto-generated method stub
14      }
15      //结束回调（没有错误）
16      @Override
17      public void onCompleted(SpeechError arg0) {
18          //TODO Auto-generated method stub
19      }
20  };
```

（7）单击“朗读电子书”按钮时，显示等待条并且把数据传递给后台，关键代码如下：

```
01  /**
02   * 单击开始阅读时弹出等待条
03   **/
04  @Override
05  public void onClick(View v) {
```

明日科技

```
06      //TODO Auto-generated method stub
07      switch (v.getId()) {
08          case R.id.button:
09              dialog = ProgressDialog.show(this,null,"正在缓冲.....");      //显示等待条
10              String text = textView.getText().toString();                  //将文字获取出来
11              speechSynthesizer.startSpeaking(text, this);                  //数据传递给后台
12              break;
13      }
14  }
```

扩展学习

本实例中的电子书内容是设定好的，尝试实现动态修改电子书内容并进行朗读。

实例 100　语音发短信

应用场景：开车时发送短信

源码位置：光盘\Code\11\100
视频位置：光盘\Video\11\100

实例说明

本实例将实现语音发短信功能，如果用户在开车的时候，要给某人发短信，只需要打开此软件，然后说出短信内容，比如说 Jerry 给 Tom 发短信，Jerry 只需对着软件说：“发短信给 Tom（停顿一秒）短信内容是（停顿一秒）你的 Jerry 在找你”。运行实例，将显示如图 11.10 所示的“正在等待指令”界面，说出指令后将显示图 11.11 所示的“短信已送达”界面。

图 11.10　等待指令

图 11.11　短信已送达

关键技术

本实例主要使用 String 类中的 substring() 方法、SmsManager 类中的 sendTextMessage() 方法，实现截取字符串并发送短信。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“VoiceSendSms”，公司域名为“mingrisoft.com”。

（2）在科大讯飞官网注册账号，并申请 Appkey。

（3）在“app\libs”文件夹中粘贴语音识别的 Jar 包（资源位置：光盘\Code\11\100\src），用于与后台进行交互。

（4）导入汉字转拼音的关键类。

（5）在 SMSActivity 类中设置汉字转拼音的工具类，关键代码如下：

```
//将源数据按照a-z的顺序排序
Collections.sort(SourceDateList, pinyinComparator);
//实例化汉字转拼音类
characterParser = CharacterParser.getInstance();
pinyinComparator = new PinyinComparator();
```

（6）在语音识别回调后，将返回的用户名，转换成拼音。具体代码如下：

```
String namePY = string.substring(4, string.indexOf(", "));
//汉字转换成拼音
String pinyin = characterParser.getSelling(namePY);
filterData(pinyin);
```

（7）通过转换的拼音去手机通讯录里获取联系人，并获取手机号，具体代码如下：

```
//获取手机联系人
Cursor phoneCursor = resolver.query(Phone.CONTENT_URI,
        PHONES_PROJECTION, null, null, null);
if (phoneCursor != null) {

    while (phoneCursor.moveToNext()) {
        //获取手机号码
        String phoneNumber = phoneCursor.getString(1);
    }
}
```

（8）给获取到的短信接收人发送短信，发送短信的相关代码如下：

```
//获取短信管理器
SmsManager smsManager = SmsManager.getDefault();
//如果汉字字数大于70个
if (smsText.length() > 70) {
//返回多条短信
```

```
06      List<String> contents = smsManager.divideMessage(string);
07      for (String sms : contents) {
08          //1.目标地址：电话号码；2.原地址：短信中心服号码；3.短信内容；4.意图
09          smsManager.sendTextMessage(num, null, sms, null, null);
10      }
11  } else {
12      smsManager.sendTextMessage(num, null, smsText, null, null);
13  }
14  Toast.makeText(this, "短息发送成功", Toast.LENGTH_SHORT).show();
15  string = "";
```

扩展学习

在实例中直接发送短信给某人，为防止错发短信，在发送短信时，弹出对话框让用户确认短信内容及收信人。

实例 101　语音唤醒

应用场景：手机找不到的时候

源码位置：光盘\Code\11\101
视频位置：光盘\Code\11\101

实例说明

视频讲解

当找不到手机的时候，你只需要说“你在哪里”，如果手机听到之后，会开启 SOS 闪光灯，并且播放音乐。当找到手机之后，对手机说“打个哈哈”，手机将关闭 SOS 闪光灯和音乐（注意：音乐的声音可能会影响到你说话的声音）。用户说“你在哪里”时的效果，如图 11.12 所示；用户说“出来聊聊”时的效果，图 11.13 所示，如果已安装“会聊天的机器人”（实例 097），手机将启动该 App 应用。

图 11.12　唤醒词“你在哪里”

图 11.13　唤醒词“出来聊聊”

关键技术

本实例主要用到了 MediaPlayer 类中的 start() 方法、stop() 方法、release() 方法，以及 String 类中的 equals() 方法，实现语音唤醒功能。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“ApplicationLock”，公司域名为“mingrisoft.com”。

（2）去百度语音的官网申请 Appkey，并下载语音唤醒的 SDK。

（3）在“app\libs”文件夹中粘贴百度语音识别的 Jar 包（资源位置：光盘 \Code\11\101\src），用于语音唤醒。

（4）在“app\main”文件夹中创建 assets 文件夹与 jniLibs 文件夹，粘贴百度官网相关的文件。

（5）修改显示界面的主题样式，去除系统默认添加的 ActionBar 标题栏。实现方法是将“app\res\values\styles”文件中的“DarkActionBar”修改为“NoActionBar”。

（6）创建名称为 PowerLED 的操作闪光灯的类文件，用于实现闪光灯的打开与关闭。关键代码如下：

```
01  /**
02   * 打开闪光灯
03   */
04  public void turnOn() {
05      if (!m_isOn) {                                          //如果闪光灯没有打开
06          m_isOn = true;
07          try {
08              Camera.Parameters mParameters;
09              mParameters = m_Camera.getParameters();
10              mParameters.setFlashMode(Camera.Parameters.FLASH_MODE_TORCH);//开启闪光灯
11              m_Camera.setParameters(mParameters);                //设置相机参数
12          } catch (Exception ex) {
13          }
14      }
15  }
16  /**
17   * 关闭闪光
18   */
19  public void turnOff() {
20      if (m_isOn) {                                           //如果闪光灯已打开
21          m_isOn = false;
22          try {
23              Camera.Parameters mParameters;
24              mParameters = m_Camera.getParameters();
25              mParameters.setFlashMode(Camera.Parameters.FLASH_MODE_OFF);  //关闭闪光灯
26              m_Camera.setParameters(mParameters);                //设置相机参数
27          } catch (Exception ex) {
28          }
29      }
30  }
```

明日科技

（7）在 MainActivity 类文件中实现唤醒成功时，播放音乐，并开启 SOS 闪光灯。要实现 SOS 效果，需要设置闪光灯每隔 0.8s 执行一次打开或关闭。具体代码如下：

```
mWakelock.acquire();                                    //唤醒屏幕
mediaPlayer1.seekTo(0);                                 //音乐从头开始
mediaPlayer1.start();                                   //播放音乐
timer = new Timer();                                    //初始化计时器
vibrator = (Vibrator) getSystemService(VIBRATOR_SERVICE);
long [] pattern = {100,400,100,400};                    //闪光灯开启、关闭的时间
/**
 * 计时器
 * 1000标示这个计时器最先打开后延迟1s执行
 * 100标示每隔0.8秒执行一次
 **/
timer.schedule(new TimerTask() {
    @Override
    public void run() {
        runOnUiThread(new Runnable() {
            @Override
            public void run() {
                if (powerLED.m_isOn) {                  //如果闪光灯是开启的
                    powerLED.turnOff();                 //关闭闪光灯
                } else {                                //否则
                    powerLED.turnOn();                  //打开闪光灯
                }
            }
        });
    }
}, 1000, 800);
```

（8）在 MainActivity 类的 onDestroy() 方法中，释放计时器、音乐播放器以及百度语音唤醒，具体代码如下：

```
//停止唤醒监听
mWpEventManager.send("wp.stop", null, null, 0, 0);
mediaPlayer1.stop();
mediaPlayer1.release();                                 //释放mediaPlayer
mWakelock.release();                                    //释放语音唤醒
timer.cancel();                                         //结束计时
```

（9）在 AndroidManifest.xml 文件中注册百度语音的相关权限以及百度的 Appkey。

扩展学习

本实例中实现了语音唤醒手机，即手机会播放音乐与开启闪烁闪光灯，尝试实现手机唤醒成功后设置闪关灯常开。

实例 102　仿微信输入

应用场景：语音输入法

源码位置：光盘\Code\11\102
视频位置：光盘\Video\11\102

实例说明

在打字的时候，为了节省用户时间，使用语音输入即可。本实例将模仿微信输入实现将语音转换为对应的文字的功能，运行程序后，正对话筒说出语音内容，程序即可实现将语音内容转换为文本。运行实例，语音识别输入界面如图 11.14 所示，手动输入文本界面如图 11.15 所示。

图 11.14　语音识别输入界面

图 11.15　手动输入文本界面

关键技术

本实例要使用 View 类中的 setVisibility() 方法、setOnTouchListener() 方法以及 Thread 类中的 sleep() 方法，实现显示或隐藏视图，并设置单击监听事件。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“VoiceText”，公司域名为“mingrisoft.com”。

（2）在科大讯飞官网申请 Appkey 并下载语音唤醒的 SDK。

（3）在“app\libs”文件夹中粘贴科大讯飞的 Jar 包（资源位置：光盘\Code\11\102\src）。

（4）在“app\main”文件夹中创建 jniLibs 文件夹，并导入相关的 .SO 文件。

（5）在 MainActivity 的类文件中，设置通用的回调接口，包括消息、数据和识别的回调方法，关键代码如下：

```
/**
 * 通用回调接口
 */
private SpeechListener listener = new SpeechListener() {
    //消息回调
    @Override
    public void onEvent(int arg0, Bundle arg1) {
        //TODO Auto-generated method stub
    }
    //数据回调
    @Override
    public void onData(byte[] arg0) {
        //TODO Auto-generated method stub
    }
    //结束回调（没有错误）
    @Override
    public void onCompleted(SpeechError arg0) {
        //TODO Auto-generated method stub
    }
};
```

（6）在 MainActivity 类中，设置设备的回调监听，用于获取识别后返回的数据。具体代码如下：

```
/**
 * 识别回调监听器
 */
private RecognizerDialogListener dialogListener = new RecognizerDialogListener() {
    //识别结果回调
    @Override
    public void onResult(RecognizerResult arg0, boolean arg1) {
        //TODO Auto-generated method stub
        String text = JsonParser.parseIatResult(arg0.getResultString());  //获取返回的数据
        editText.append(text);
        editText.setSelection(editText.length());                          //获取输入框的长度
    }

    //识别结束回调
    @Override
    public void onError(SpeechError arg0) {
        //TODO Auto-generated method stub
    }
};
```

（7）在 onCreate() 方法中添加以下代码，用来绑定科大讯飞语音的 Appkey。

```
01  //换成自己申请的appid
02  SpeechUser.getUser().login(MainActivity.this, null, null, "appid=573a7bf0", listener);
```

（8）在 Android Manifests.xml 中添加网络及获取话筒的相关权限。

扩展学习

仿照微信、QQ，尝试实现单击语音输入框右边的添加按钮后，下方显示选项卡，选项卡内容可以为图片、视频、表情、位置等。

明日科技

第12章 传感器

实例 103 计步器

应用场景：计步软件

源码位置：光盘\Code\12\103

视频位置：光盘\Video\12\103

视频讲解

实例说明

加速度传感器不仅可以用来制作摇晃手机一类的 App，还可以用来实现计步器功能。实例运行效果如图 12.1 所示，默认目标步数为 6000 步；修改目标步数后根据走步的速度显示当前的步数与进度，如图 12.2 所示。

图 12.1 默认目标步数

图 12.2 显示步数与进度

关键技术

本实例主要使用 CircularRingPercentageView 自定义类中的 setTarGet() 自定义方法、SensorEventListener 接口中的 onSensorChanged() 方法、SaveKeyValues 自定义类中的 putIntValues() 方法，实现计步器功能。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“Pedometer”，公司域名为“mingrisoft. com”。

（2）导入资源文件：复制“光盘\Code\12\103\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）在“Resource\app\src\main\java”目录中创建 utils 包，在该包中创建 SaveKeyValues 类。该类为保存数据值的工具类。

（4）打开 MainActivity 类，实现 SensorEventListener 传感器监听接口。首先在 onSensorChanged() 方法中获取 x 轴、y 轴、z 轴的平均值。代码如下：

```
01  //判断传感器的类型是否为重力传感器（加速度传感器）
02  int j = (sensor.getType() == Sensor.TYPE_ACCELEROMETER) ? 1 : 0;
03  if (j == 1) {
04      float vSum = 0;
05      //获取x轴、y轴、z轴的加速度
06      for (int i = 0; i < 3; i++) {
07          final float v = mYOffset + event.values[i] * mScale[j];
08          vSum += v;
09      }
10      int k = 0;
11      float v = vSum / 3;                                    //获取x轴、y轴、z轴加速度的平均值
```

根据两次运动的间隔时间判断是否走步。代码如下：

```
01  if (mEnd - mStart > 500) {
02      CURRENT_SETP++;                                        //加一步
03      mLastMatch = extType;
04      mStart = mEnd;                                         //结束时间为开始时间
05      if (CURRENT_SETP <= tarGet) {
06          progressCircle.setProgress(CURRENT_SETP);          //设置步数的进度与进度条结合
07          progressCircle.postInvalidate();                   //刷新
08      }
09      //存储当前的步数
10      SaveKeyValues.putIntValues("sport_steps", CURRENT_SETP);
11      stepNumber.setText(CURRENT_SETP + " ");                //显示当前步数
12  }
```

（5）在“Resource\app\src\main\java”目录中创建 view 包，在该包中创建 CircularRingPercentage View 类。在 onDraw() 方法中实现渐变色计步进度条的绘制。代码如下：

```
protected void onDraw(Canvas canvas) {
    super.onDraw(canvas);
    float share = 360 / tarGet;                                    //根据步数计算进度
    int scale = (int) tarGet;                                      //根据步数显示刻度值
    paint.setShader(sweepGradient);                                //背景渐变颜色
    //绘制渐变色
    canvas.drawArc(oval, -90, (progress * share), false, paint);
    paint.setShader(null);
    //是否是线条模式
    if (!isLine) {
        float start = -90f;                                        //起始角度
        float p = ((float) maxColorNumber / tarGet);               //计算一步的进度
        p = (int) (progress * p);                                  //当前步数的进度
        for (int i = 0; i < p; i++) {
            paint.setColor(roundBackgroundColor);                  //设置画笔颜色
            canvas.drawArc(oval, start + singlPoint - lineWidth,
                    lineWidth, false, paint);                      //绘制间隔块
            start = (start + singlPoint);                          //下一个起始位置
            postInvalidate();                                      //刷新
        }
    }
    //绘制剩下的空白区域
    paint.setColor(roundBackgroundColor);
    canvas.drawArc(oval, -90, (-(tarGet - progress) * share), false, paint);
    //绘制文字刻度
    for (int i = 1; i <= 10; i++) {
        canvas.save();                                             //保存当前画布
        canvas.rotate(360 / 10 * i, circleCenter, circleCenter);
        canvas.drawText(i * (scale / 10) + "", circleCenter,
                circleCenter - radius + roundWidth / 2 +
                        getDpValue(4) + textSize, mPaintText);
        canvas.restore();                                          //恢复画布
    }
}
```

扩展学习

本实例通过加速度传感器已经实现了计步器的功能，请尝试实现改变计步进度条的分割数量。首先在 CircularRingPercentageView 类中创建 setMaxColorNumber() 方法，用于设置进度条分割的数量，然后在 MainActivity 类的 onCreate() 方法中，调用 setMaxColorNumber() 方法。

实例 104　仿微信摇一摇

应用场景：摇一摇功能

源码位置：光盘\Code\12\104
视频位置：光盘\Video\12\104

视频讲解

实例说明

本实例将通过加速度传感器来实现一个仿微信摇一摇功能。运行实例，上下摇动手机，如图 12.3 所示；手机屏幕将显示摇一摇动画效果，如图 12.4 所示；动画结束后将显示提示信息，如图 12.5 所示。

图 12.3　上下摇动手机

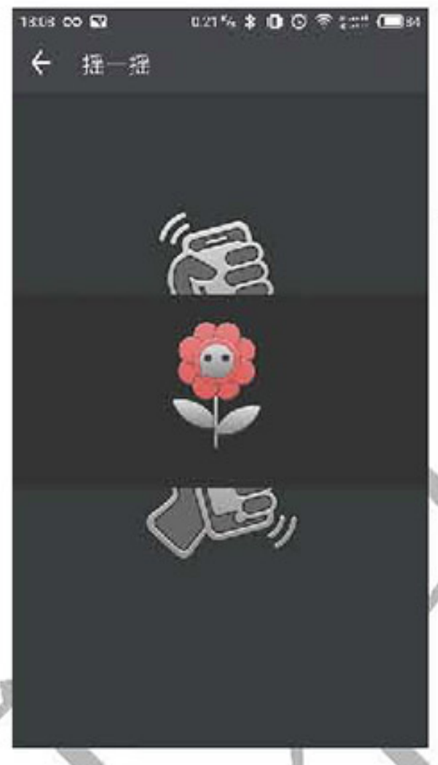


图 12.4　显示摇一摇动画

图 12.5　显示提示信息

关键技术

关键技术

本实例主要使用 Context 类中的 getSystemService() 方法、TranslateAnimation 类中的 TranslateAnimation() 构造方法、SensorManager 类中的 registerListener() 方法，实现仿微信摇一摇的运行效果。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“WeiShake”，公司域名为“mingrisoft. com”。

（2）导入资源文件：复制“光盘\Code\12\104\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）在“WeiShake\app\src\main\java\com\mingrisoft\weishake”目录中创建 ShakeListener 类，让该类实现 SensorEventListener 接口，并通过 onSensorChanged() 方法实现获得加速度传感器感应变化的数据。代码如下：

```
01  public void onSensorChanged(SensorEvent event) {
02      //当前检测时间
03      long currentUpdateTime = System.currentTimeMillis();
04      //两次检测的时间间隔
```

```
    long timeInterval = currentUpdateTime - lastTime;
    //判断是否达到了检测时间间隔
    if (timeInterval < INTERVAL_TIME)
       return;
    //当前的时间变成last时间
    lastTime = currentUpdateTime;
    //获得x,y,z坐标值
    float x = event.values[0];
    float y = event.values[1];
    float z = event.values[2];
    //获得x,y,z的变化值
    float deltaX = x - lastX;
    float deltaY = y - lastY;
    float deltaZ = z - lastZ;
    //最后的坐标
    lastX = x;
    lastY = y;
    lastZ = z;
    //计算速度
    double speed = Math.sqrt(deltaX * deltaX + deltaY * deltaY + deltaZ * deltaZ)
          / timeInterval * 10000;
    //达到速度阀值，发出提示
    if (speed >= VELOCIT_YVALUE) {
       onShakeListener.onShake();
    }
}
```

（4）打开 MainActivity 类，在 startAnim() 方法中实现摇一摇图片向上并返回动画。代码如下：

```
AnimationSet animup = new AnimationSet(true);
TranslateAnimation mytranslateanimup0 = new                    //创建向上动画
        TranslateAnimation(Animation.RELATIVE_TO_SELF,
        0f,Animation.RELATIVE_TO_SELF,
        0f,Animation.RELATIVE_TO_SELF,-0.5f);
mytranslateanimup0.setDuration(1000);                          //持续时间
TranslateAnimation mytranslateanimup1 = new                    //创建向下动画
        TranslateAnimation(Animation.RELATIVE_TO_SELF,
        0f,Animation.RELATIVE_TO_SELF,
        0f,Animation.RELATIVE_TO_SELF,
        0f,Animation.RELATIVE_TO_SELF,+0.5f);
mytranslateanimup1.setDuration(1000);                          //持续时间
mytranslateanimup1.setStartOffset(1000);                       //向上动画结束后执行返回动画
animup.addAnimation(mytranslateanimup0);                       //添加向上动画
```

明日科技

```
16  animup.addAnimation(mytranslateanimup1);                    //添加向下动画
17  imgUp.startAnimation(animup);                               //启动该动画
```

扩展学习

本实例通过加速度传感器实现仿微信摇一摇的效果，请尝试在 MainActivity 类中通过重写 onTouchEvent() 方法，实现触摸屏幕执行摇一摇的动画效果。

实例 105　水平仪

应用场景：生活工具

源码位置：光盘\Code\12\105

视频位置：光盘\Video\12\105

实例说明

方向传感器能够判断出手机在各个方向的旋转角度，利用这些角度就可以实现指南针、水平仪的功能。本实例将通过方向传感器来实现一个水平仪工具。运行实例，手机放在一个水平位置时小球将显示在圆盘的中心位置，如图 12.6 所示；根据手机抬起的角度，小球将向抬起方向滑动，如图 12.7 所示。

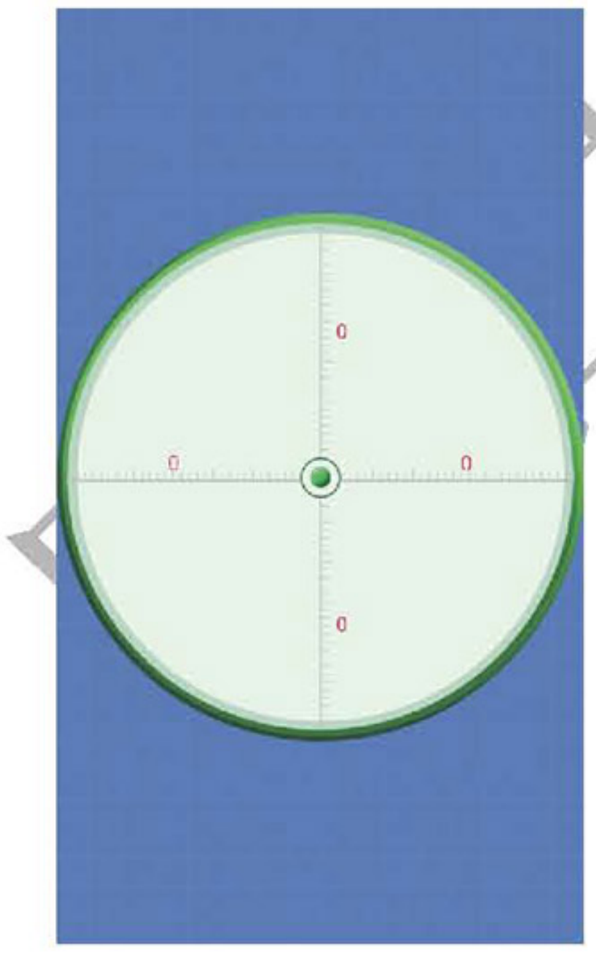


图 12.6　手机水平状态

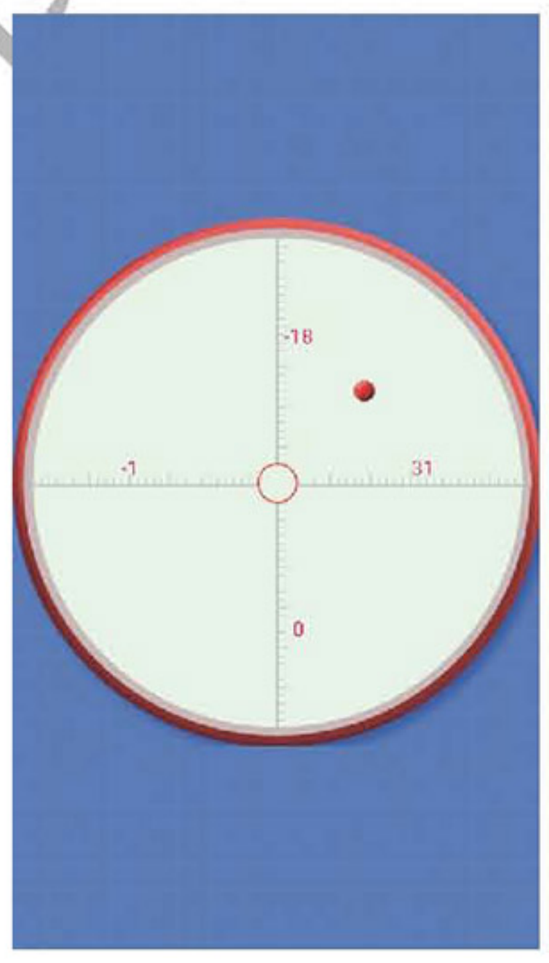


图 12.7　小球向抬起方向滑动

关键技术

本实例主要使用 SensorManager 类中的 getDefaultSensor() 方法、SensorEventListener 接口中的 onSensorChanged() 方法、View 类中的 onDraw() 方法，实现水平仪的效果。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“Level”，公司域名为“mingrisoft. com”。

（2）导入资源文件：复制“光盘\Code\12\105\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）打开 MainActivity 类，该类实现 SensorEventListener 传感器监听接口，并在 onSensorChanged() 方法中首先根据 z 轴与 y 轴的值计算出倾斜角度。代码如下：

```
01  //z轴转过的角度
02  int zAngle = (int) values[0];                    //后期使用获取方向
03  int yAngle = (int) values[1];
04  if (+yAngle>=0){
05      by.setText("+"+yAngle);                      //显示上方负角度值
06  }if (+yAngle<=0){
07      ty.setText("+"+yAngle);                      //显示下方正角度值
08  }
09  //y轴滚动时与x轴的角度
10  int xAngle = (int) values[2];
11  if (+xAngle>=0){
12      rx.setText("+"+xAngle);                      //显示右侧正角度值
13  }if (+xAngle<=0){
14      lx.setText("+"+xAngle);                      //显示左侧负角度值
15  }
```

然后根据 z 轴与 y 轴倾斜角度计算出小球显示的位置。代码如下：

```
01  //如果沿x轴的倾斜角在最大角度之内则计算出其相应坐标值
02  if (Math.abs(xAngle) <= MAXIMUM_ANGLE) {
03      //根据与x轴的倾斜角度计算x坐标的变化值（倾斜角度越大，x坐标变化越大）
04      int deltaX = (int) (ax * xAngle / MAXIMUM_ANGLE);
05      x += deltaX;
06  }
07  //如果沿x轴的倾斜角已经大于最大角度，小球在最左边
08  else if (xAngle > MAXIMUM_ANGLE) {
09      x = 0;
10  }
11  //如果与x轴的倾斜角已经小于负的最大角度，小球在最右边
12  else {
13      x = ax * 2;
14  }
15  //如果沿y轴的倾斜角还在最大角度之内
16  if (Math.abs(yAngle) <= MAXIMUM_ANGLE) {
17      //根据沿y轴的倾斜角度计算y轴坐标的变化值（倾斜角度越大，y轴坐标变化越大）
18      int deltaY = (int) (ay * yAngle / MAXIMUM_ANGLE);
```

明日科技

```
        y += deltaY;
    }
    //如果沿y轴的倾斜角已经大于最大角度，小球在最下边
    else if (yAngle > MAXIMUM_ANGLE) {
        y = ay * 2;
    }
    //如果沿y轴的倾斜角已经小于负的最大角度，小球在最上边
    else {
        y = 0;
    }
    //如果计算出来的x轴、y轴坐标还位于水平仪的仪表盘内，更新水平仪的小球坐标
    if (isContain(x, y)) {
        myview.ballX = x;
        myview.ballY = y;
    }
    //通知系统重回MyView组件重绘界面
    myview.postInvalidate();
```

（4）在“Level\app\src\main\java”目录中创建 view 包，在该包中创建 MyView 类。该类继承自 View。通过 onDraw() 方法来实现小球位置的绘制。代码如下：

```
protected void onDraw(Canvas canvas) {
    super.onDraw(canvas);
    WindowManager wm = (WindowManager) getContext()
            .getSystemService(Context.WINDOW_SERVICE);          //获取窗口服务
    int width = wm.getDefaultDisplay().getWidth();              //获取窗口宽度
    int height = wm.getDefaultDisplay().getHeight();            //获取窗口高度
    int imgHeight = compass.getHeight();                        //获取圆盘图片高度
    int cy = (height - imgHeight) / 2;                          //计算绘制圆盘y轴坐标
    if (accuracy==false){                                       //如果手机处于倾斜状态
        //绘制水平仪圆盘图片
        canvas.drawBitmap(compass1, 0, cy, null);
        //根据小球坐标绘制小球
        canvas.drawBitmap(ball, ballX, ballY, null);
    }else {                                                     //手机处于水平状态
        //绘制水平仪圆盘图片
        canvas.drawBitmap(compass, 0, cy, null);
    }
}
```

扩展学习

本实例通过方向传感器来实现水平仪的运行效果，请尝试修改 MainActivity 类的 onSensorChanged() 方法，使处于水平状态时通过 Toast.makeText() 方法进行文字提示。

实例 106 磁场指南针

应用场景：生活工具

源码位置：光盘\Code\12\106
视频位置：光盘\Video\12\106

实例说明

视频讲解

磁场传感器，主要读取的是磁场的变化，通过该传感器便可开发出指南针、罗盘等磁场应用。该传感器读取的数据同样是空间坐标系三个方向的磁场值，其数据单位为 μT（微特斯拉）。本实例将通过磁场传感器来实现一个辨别方向的指南针。运行实例，效果如图 12.8 所示，手机屏幕向上顶部指向正北时，指南针中的红色箭头将指向表盘中的正北。

图 12.8　磁场指南针

关键技术

关键技术

本实例主要使用 BitmapFactory 类中的 decodeResource() 方法、SensorManager 类中的 getDefault Sensor () 方法、SensorEventListener 接口中的 onSensorChanged() 方法，实现磁场指南针的效果。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“Compass”，公司域名为“mingrisoft. com”。

（2）导入资源文件：复制“光盘\Code\12\106\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）在“Compass\app\src\main\java”目录中创建 view 包，在该包中创建 PointerView 类，让其继承自 View 类并实现 SensorEventListener 接口。通过 onSensorChanged () 方法获取磁场传感器三轴的输出信息值。代码如下：

```
01  public void onSensorChanged(SensorEvent event) {
02      if (event.sensor.getType() == Sensor.TYPE_MAGNETIC_FIELD) {  //如果是磁场传感器
03          float value[] = event.values;                            //获取磁场传感器三轴的输出信息
```

```
04            allValue = value;                    //保存输出信息
05            super.postInvalidate();              //刷新界面
06        }
07    }
```

在 onDraw () 方法中通过获取的传感器三轴的信息值计算旋转的角度并绘制方向位图。代码如下：

```
01  protected void onDraw(Canvas canvas) {
02      super.onDraw(canvas);
03      Paint p = new Paint();                     //创建画笔
04      p.setAntiAlias(true);                      //开启抗锯齿
05      p.setDither(true);                         //开启仿抖动
06      if (allValue != null) {                    //传感器三轴输出信息不为空
07          float x = allValue[0];                 //获取x轴坐标
08          float y = allValue[1];                 //获取y轴坐标
09          canvas.save();                         //保存Canvas的状态
10          canvas.restore();                      //重置绘图对象
11          //以屏幕中心点作为旋转中心
12          canvas.translate(super.getWidth() / 2, super.getHeight() / 2);
13          //判断y轴为0时的旋转角度
14          if (y == 0 && x > 0) {
15              canvas.rotate(90);                 //旋转角度为90度
16          } else if (y == 0 && x < 0) {
17              canvas.rotate(270);                //旋转角度为270度
18          } else {
19              //通过三角函数tanh()方法计算旋转角度
20              if (y >= 0) {
21                  canvas.rotate((float) Math.tanh(x / y) * 90);
22              } else {
23                  canvas.rotate(180 + (float) Math.tanh(x / y) * 90);
24              }
25          }
26      }
27      //绘制方向位图
28      canvas.drawBitmap(this.pointer, -this.pointer.getWidth() / 2,
29              -this.pointer.getHeight() / 2, p);
30  }
```

扩展学习

本实例通过磁场传感器实现了指南针，尝试在 MainActivity 类中首先填写动态获取权限的代码，然后创建 getLocationInformation() 方法，实现获取定位信息的坐标并显示出来。

实例 107 光线控制页面背景

应用场景：光线传感器

源码位置：光盘\Code\12\107

视频位置：光盘\Video\12\107

实例说明

光线传感器，可以感应到手机周围环境的光线强度。本实例将通过光线传感器来控制屏幕的亮度，并根据该亮度让手机区分白天与夜晚。运行实例，当用户没有遮挡手机听筒旁的光线传感器时，屏幕将显示白天，如图 12.9 所示；如果用户遮挡光线传感器时，屏幕将根据亮度值变暗并显示为夜晚，如图 12.10 所示。

图 12.9　手机显示白天

图 12.10　手机显示夜晚

关键技术

本实例主要使用 Sensor 类中的 getType() 方法，Window 类中的 getAttributes() 方法、setAttributes () 方法，实现手机判断白天与夜晚的效果。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“LightSensor”，公司域名为“mingrisoft. com”。

（2）导入资源文件：复制“光盘\Code\12\107\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）打开 MainActivity 类，让该类实现 SensorEventListener 传感器监听接口，并通过 onResume() 方法实现界面获取焦点时注册光线传感器的监听器。代码如下：

```
01  public void onResume() {
02      super.onResume();
```

```
        //注册监听方光线传感器
        sensorManager.registerListener(this,
                sensorManager.getDefaultSensor(Sensor.TYPE_LIGHT),
                SensorManager.SENSOR_DELAY_GAME);
    }
```

通过 onSensorChanged() 方法实现通过光线控制屏幕亮度的效果。代码如下：

```
public void onSensorChanged(SensorEvent event) {
    //获取传感器值
    float[] values = event.values;
    //获取传感器类型
    int sensorType = event.sensor.getType();
    switch (sensorType) {
        case Sensor.TYPE_LIGHT:
            //获取窗口管理属性
            WindowManager.LayoutParams lp = this.getWindow().getAttributes();
            //计算屏幕亮度
            lp.screenBrightness = Float.valueOf(values[0]) * (1f / 255f);
            this.getWindow().setAttributes(lp);                    //设置屏幕亮度
            if (values[0] == 0) {                                  //如果光照值为0
                //更换黑夜背景图片
                relativeLayout.setBackgroundResource(R.drawable.night);
            } else {
                //否则更换白天图片
                relativeLayout.setBackgroundResource(R.drawable.day);
            }
    }
}
```

扩展学习

本实例通过光线传感器来实现控制屏幕的亮度并让手机区分白天和夜晚，尝试在 onSensorChanged() 方法中判断白天或者是夜晚，并通过 Toast.makeText() 方法进行文字提示。

实例 108　不要靠近听筒

应用场景：距离传感器

源码位置：光盘\Code\12\108
视频位置：光盘\Video\12\108

实例说明

距离传感器，可以感应手机和人体的距离。当手机屏幕贴近用户脸部时，就会自动感应出手机和人体的距离是多少。本实例将通过距离传感器的值来控制屏幕是否熄灭。运行实例，当用户没有靠近

听筒（距离传感器）的位置时，屏幕正常显示，如图 12.11 所示；如果用户的耳朵靠近听筒（距离传感器）时，屏幕将自动熄灭，并发出开枪的声音，如图 12.12 所示。

图 12.11　屏幕正常显示

图 12.12　屏幕自动熄灭

关键技术

本实例主要使用 PowerManager 类中的 newWakeLock() 方法、PowerManager.WakeLock 类中的 acquire() 方法、release() 方法，实现靠近距离传感器后锁屏的效果。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“DistanceSensor”，公司域名为“mingrisoft. com”。

（2）导入资源文件：复制“光盘\Code\12\108\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）打开 MainActivity 类，在 onCreate() 方法中获取传感器服务与电源管理服务，并进行音效的加载。代码如下：

```
01  protected void onCreate(Bundle savedInstanceState) {
02      super.onCreate(savedInstanceState);
03      setContentView(R.layout.activity_main);
04      //获取手机传感器管理服务
05      sensorManager = (SensorManager)getSystemService(SENSOR_SERVICE);
06      //获取电源管理服务
07      powerManager = (PowerManager) getSystemService(POWER_SERVICE);
08      //第一个参数为电源锁级别，第二个参数为Log日志
09      localWakeLock = this.powerManager.newWakeLock(32,"锁");
10      AudioAttributes attr = new AudioAttributes.Builder()
11              .setUsage(AudioAttributes.USAGE_GAME)          //设置音效使用场景
```

```
12              //设置音效的类型
13              .setContentType(AudioAttributes.CONTENT_TYPE_MUSIC)
14              .build();
15      soundPool = new SoundPool.Builder()
16              .setAudioAttributes(attr)                    //设置音效池的属性
17              .setMaxStreams(10)                           //设置最多可容纳10个音频流，
18              .build();
19      //加载音乐资源
20      soundId = soundPool.load(this, R.raw.music, 1);
21  }
```

通过 onResume() 方法实现界面获取焦点时，进行距离传感器的监听注册。代码如下：

```
01  public void onResume() {
02      super.onResume();
03      //注册监听距离传感器
04      sensorManager.registerListener(this,
05              sensorManager.getDefaultSensor(Sensor.TYPE_PROXIMITY),
06              SensorManager.SENSOR_DELAY_UI);
07  }
```

通过在 onSensorChanged() 方法中判断传感器的值，来实现锁屏的效果。代码如下：

```
01  public void onSensorChanged(SensorEvent event) {
02      float[] values = event.values;                       //获取距离传感器的值
03      int sensorType = event.sensor.getType();             //获取传感器类型
04      switch (sensorType) {
05          case Sensor.TYPE_PROXIMITY:
06              if (values[0] == 0.0) {                      //如果值为0，为贴近手机
07                  if (localWakeLock.isHeld()) {            //如果电源锁已被唤醒
08                  } else {                                 //如果没有唤醒
09                      localWakeLock.acquire();             //申请设备电源锁
10                  }
11                  soundPool.play(soundId, 1, 1, 0, 0, 1);  //播放音效
12              } else {                                     //远离手机
13                  localWakeLock.setReferenceCounted(false);
14                  localWakeLock.release();                 //释放设备电源锁
15              }
16              break;
17      }
18  }
```

（4）打开“DistanceSensor\app\src\main”目录中的 AndroidManifest.xml 文件，在该文件中添加锁屏权限的代码。

扩展学习

本实例中通过距离传感器实现了锁屏并播放枪声音效的运行效果，个别手机由于其传感器所具备的特质，可能会连续发出枪声。请尝试通过添加“private boolean record”标记来解决连续枪声的问题。

实例 109	话筒吹雾小游戏 应用场景：话筒传感器	源码位置：光盘\Code\12\109

实例说明

话筒传感器，学名为传声器。话筒是将声音信号转换为电信号的能量转换器。在 Android 系统 6.0 版本以上的手机中需要手动或动态获取话筒权限。本实例将通过话筒传感器实现一个吹雾并擦除的运行效果。运行实例，向手机底部的话筒吹气，屏幕将渐渐出现雾气，效果如图 12.13 所示；通过手指在雾气上滑动以实现擦除雾气的效果，如图 12.14 所示。

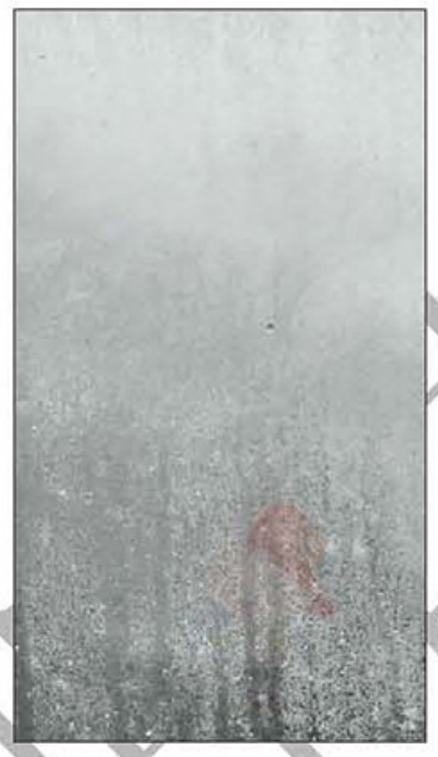


图 12.13　屏幕显示雾气

图 12.14　擦除雾气

关键技术

本实例主要使用 AudioRecord 类中的 AudioRecord() 构造方法、startRecording() 方法、read() 方法，实现吹雾的效果。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“BlowingFog”，公司域名为“mingrisoft. com”。

（2）导入资源文件：复制“光盘\Code\12\109\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）在“app\src\main\java”目录中创建 utils 包，在该包中创建 AudioRecordManger 类。在

AudioRecord 构造方法创建录音对象，并初始化对象属性。代码如下：

```
//创建录音对象，并初始化对象属性
mAudioRecord = new AudioRecord(MediaRecorder.AudioSource.MIC,
        SAMPLE_RATE_IN_HZ, AudioFormat.CHANNEL_IN_DEFAULT,
        AudioFormat.ENCODING_PCM_16BIT, BUFFER_SIZE);
```

在新线程的 run() 方法中进行向话筒吹气分贝值的计算。代码如下：

```
//r是实际读取的数据长度，一般而言r会小于buffersize
int r = mAudioRecord.read(buffer, 0, BUFFER_SIZE);
long v = 0;
//将buffer内容取出，进行平方和运算
for (int i = 0; i < buffer.length; i++) {
    v += buffer[i] * buffer[i];
}
double mean = v / (double) r;                    //平方和除以数据总长度，得到音量大小
double volume = 10 * Math.log10(mean);           //计算分贝值
```

（4）在“app\src\main\java”目录中创建 view 包，在该包中创建 DrawView 类，该类继承自 View。通过重写 onDraw() 方法来实现雾气的绘制。代码如下：

```
protected void onDraw(Canvas canvas) {
    super.onDraw(canvas);
    canvas.drawColor(Color.TRANSPARENT);
    if (!mComplete) {                                //如果未擦除完成
        mPaint.setXfermode(new PorterDuffXfermode(PorterDuff.Mode.DST_OUT)); //设定目标图模式
        canvas.drawBitmap(mBitmap, 0, 0, null);      //画图
        mCanvas.drawPath(mPath, mPaint);             //画布路径
    }
    //如果擦除干净，则进行资源释放操作
    if (mComplete) {
        if (mListener != null) {
            mListener.complete();                    //监听结束
            setEndValues();
        }
    }
}
```

通过在 Runnable 对象的 run() 方法中计算擦除气雾的百分比，实现清除图层区域。代码如下：

```
if (wipeArea > 0 && totalArea > 0) {
    //计算已经擦除的百分比
    int percent = (int) (wipeArea * 100 / totalArea);
    //如果擦除大于50%
    if (percent > 50) {
```

明日科技

```
06            mComplete = true;                    //清除掉图层区域
07            postInvalidate();                    //刷新界面
08        }
09    }
```

（5）打开 MainActivity 类，在该类中通过获取到的话筒分贝值进行起雾动画的绘制。（代码位置：BlowingFog\app\src\main\java\com\mingrisoft\blowingfog\MainActivity.java）

（6）打开“BlowingFog\app\src\main”目录中的 AndroidManifest.xml 文件，在该文件中添加录制音频权限的代码。

（7）运行该实例后会出现报错现象，需要在“设置”→“应用管理”→“吹雾”→“应用信息”→“权限管理”中开启录音权限。

扩展学习

本实例通过手动开启麦克风权限来实现吹雾的运行效果，尝试通过在 MainActivity 类中填写动态获取麦克风权限的代码来实现吹雾的运行效果。

实例 110　弹力球碰撞

应用场景：加速度传感器

源码位置：光盘\Code\12\110

视频位置：光盘\Video\12\110

实例说明

弹力球碰撞是通过摇晃手机触发手机内的加速度传感器，通过该传感器返回的值计算手机弹力球移动的坐标并进行弹力球的绘制。运行实例，弹力球将默认显示在手机屏幕的正中心，如图 12.15 所示；摇晃手机后，弹力球碰到手机屏幕边框时发生反弹，如图 12.16 所示。

图 12.15　弹力球默认显示位置

图 12.16　弹力球反弹效果

关键技术

本实例主要使用 SensorEventListener 接口中的 onSensorChanged() 方法、System 类中的 currentTimeMillis() 方法、Math 类中的 abs() 方法，实现弹力球碰撞反弹的功能。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“ElasticBallCollision”，公司域名为“mingrisoft. com”。

（2）导入资源文件：复制“光盘\Code\12\110\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）在 ElasticBallCollision 项目的“app\src\main\java”目录下创建 utils 包，该包用于保存本项目中所使用到的工具类文件，该包中包括 3 个类文件，其类名及作用如表 12.1 所示。

表 12.1　utils 包中的 3 个类

类　名	描　述
Coordinate	该类是弹力球坐标向量类
Space	弹力球的活动空间类，用于设置弹力球的移动范围
Ball	自定义弹力球类，其中包含弹力球的初速度方法、设置活动空间的方法、绘图方法、判断小球开始运动的方法等。主要通过该类中的 isCollision() 方法检测弹力球是否碰撞到手机屏幕的边框，并通过 rebound() 方法实现碰到边框后反弹的效果

（4）在 ElasticBallCollision 项目中的 java 目录下创建 view 包，然后在该包中创建 BallView 类，用于实现摇晃手机后弹力球碰到手机屏幕边框时进行反弹的效果（代码位置：ElasticBallCollision\app\src\main\java\view\BallView.java）。

通过 BallView 类的 onSensorChanged() 方法获取加速度传感器三个轴上的值，并计算手机摇晃的速度。代码如下：

```
01  public void onSensorChanged(SensorEvent event) {
02      //主要是为了清除爆发模式时人为因素增大的加速度
03      accelerationX = 0;
04      accelerationY = 0;
05      //获取现在的时间
06      long curTime = System.currentTimeMillis();
07      if ((curTime - lastTime) > 10) {
08          long diffTime = (curTime - lastTime);
09          lastTime = curTime;
10          float x = event.values[0];
11          float  y = event.values[1];
12          float z = event.values[2];
13          //计算晃动速度
14          float speed = Math.abs(x + y + z - lastX - lastY - lastZ)
15                  / diffTime * 10000;
16          //加快小球运动速度
```

```
17          if (speed > 400d) {
18              //剧烈晃动，进入疯狂模式
19              accelerationX = x*10;
20              accelerationY = y*10;
21              preTime = curTime;      //初始化时间
22          }
23          //重新计算三个轴的值
24          lastX = x;
25          lastY = y;
26          lastZ = z;
27      }
28  }
```

通过 parameter() 方法设置弹力球的向量与运动坐标。代码如下：

```
01  private void parameter() {
02      //边界碰撞检测
03      if(lBall.isCollision(0.0f, 0.0f, screenW, screenH)){
04      }
05      //设置小球的向量
06      lBall.setDirectionVector(new Coordinate(accelerationX,
07              accelerationY));
08      //小球的运动坐标
09      lBall.go(0.0f, 0.0f, screenW, screenH);
10  }
```

通过 draw() 方法进行弹力球与背景图片的绘制。代码如下：

```
01  private void draw() {
02      try {
03          canvas = this.surfaceHolder.lockCanvas();              //锁定当前画布
04          canvas.drawColor(Color.WHITE);                         //画布颜色
05          Rect rect = new Rect(0,0,getWidth(),getHeight());      //屏幕宽高
06          canvas.drawBitmap(bgBitmap,null,rect,null);            //绘制背景图片
07          lBall.draw(canvas, paint);                             //绘制小球
08
09      } catch (Exception e) {
10
11      } finally {
12          try {
13              if (canvas != null)                                //如果画布不为空
14                  surfaceHolder.unlockCanvasAndPost(canvas);     //解锁画布
15          } catch (Exception e2) {
16
17          }
18      }
19  }
```

明日科技

（5）打开 MainActivity 类，在该类中首先设置上下文全局变量，然后在 onCreate() 方法中设置显示自定义弹力球控件。代码如下：

```
01  protected void onCreate(Bundle savedInstanceState) {
02      super.onCreate(savedInstanceState);
03      lContext = this;                                    //设置上下文
04      setContentView(new BallView(lContext));             //设置自定义的弹力球控件
05  }
```

扩展学习

本实例通过 SensorEventListener 接口来实现弹力球碰撞的功能，请尝试实现小球在碰撞手机屏幕的边框时发出声音。

实例 111　使用陀螺仪控制全景视图

应用场景：陀螺仪传感器

源码位置：光盘\Code\12\111

视频位置：光盘\Video\12\111

实例说明

全景视图是通过手机中的陀螺仪传感器实现左右或上下转动手机时显示全景图片的效果。运行实例，显示默认图片位置，如图 12.17 所示；手机在水平的状态下右侧抬起向左转动手机后，显示横向移动的全景图片，如图 12.18 所示；顶部抬起向下转动手机后，显示纵向移动的全景图片，如图 12.19 所示。

图 12.17　显示默认的图片

图 12.18　显示横向移动全景

图 12.19　显示纵向移动全景

关键技术

本实例主要使用 SensorManager 类中的 getDefaultSensor() 方法、SensorEventListener 接口中的 onSensorChanged() 方法、View 类中的 onDraw() 方法，实现摇晃手机更换图片的效果。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“PanoramicView”，公司域名为“mingrisoft.com”。

（2）导入资源文件：复制“光盘\Code\12\111\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）在“PanoramicView\app\src\main\java”目录中创建 listener 包，在该包中创建 GyroSensor 类并实现 SensorEventListener 接口。在该类中，首先通过 register() 方法实现陀螺仪传感器的监听服务，然后通过在 onSensorChanged() 方法中获取的三个轴的信息来实现横向的全景显示。代码如下：

```
01  //获取三个轴的信息
02  float rotateX = Math.abs(event.values[0]);
03  float rotateY = Math.abs(event.values[1]);
04  float rotateZ = Math.abs(event.values[2]);
05  //进行横向全景实现
06  if (rotateY > rotateX + rotateZ) {          //如果y轴值大于其他两个轴的值，说明手机处于横向滚动
07      //计算事件经过的时间
08      final float dT = (event.timestamp - lastTimestamp) * nanosecond;
09      //计算沿y轴旋转的弧度
10      rotateRadianY += event.values[1] * dT;
11      //设置左侧抬起向右最大的旋转弧度
12      if (rotateRadianY > maxRotateRadian) {
13          rotateRadianY = maxRotateRadian;
14      //设置右侧抬起向左最大的旋转弧度
15      } else if (rotateRadianY < -maxRotateRadian) {
16          rotateRadianY = -maxRotateRadian;
17      } else {
18          //刷新图片显示的进度
19          for (ImageScrollView view : mViews) {
20              if (view != null && view.getOrientation()
21                              == ImageScrollView.HORIZONTAL_ORIENTATION) {
22                  view.updateProgress((float) (rotateRadianY / maxRotateRadian));
23              }
24          }
25      }
26  }
```

（4）在“PanoramicView\app\src\main\java”目录中创建 view 包，在该包中创建 ImageScrollView 类，该类继承自 ImageView。通过 onDraw() 方法实现横向与纵向全景视图的绘制工作。代码如下：

```
01  if (mOrientation == HORIZONTAL_ORIENTATION) {
02      //计算横向偏移的进度值
03      float currentOffsetX = lMaxOffset * lProgress;
04      canvas.save();                              //保存当前画布
05      canvas.translate(currentOffsetX, 0);        //使用指定的转换方式预转换当前矩阵
06      super.onDraw(canvas);
07      canvas.restore();                           //恢复画布
08  } else if (mOrientation == VERTICAL_ORIENTATION) {
09      //计算纵向偏移的进度值
10      float currentOffsetY = lMaxOffset * lProgress;
11      canvas.save();                              //保存当前画布
12      canvas.translate(0, currentOffsetY);        //使用指定的转换方式预转换当前矩阵
13      super.onDraw(canvas);
14      canvas.restore();                           //恢复画布
15  }
```

（5）打开 MainActivity 类，在该类中首先获取滚动视图控件并设置监听器，然后在 onResume() 方法中实现界面获取焦点时注册监听器，最后在 onPause() 方法实现界面停止时注销监听器。（代码位置：PanoramicView\app\src\main\java\com\mingrisoft\panoramicview\MainActivity.java）。

扩展学习

本实例实现了显示全景视图的效果。尝试通过在 ImageScrollView 类中设置自定义滚动条属性，及通过 onDraw() 方法进行滚动条的绘制，实现转动手机的同时显示滚动条。

实例 112　摇晃手机更换图片

应用场景：加速度传感器

源码位置：光盘\Code\12\112

实例说明

摇晃手机更换图片是通过对手机加速度传感器的监听，实现摇晃手机时（传感器三个轴的值达到一定的标准）更换手机图片。运行实例，将显示默认的图片，如图 12.20 所示；摇晃手机后将更换显示的图片，如图 12.21 所示。

关键技术

本实例主要使用 SensorManager 类中的 registerListener() 方法、SensorEventListener 接口中的 onSensorChanged() 方法、View 类中的 setBackgroundResource() 方法，实现摇晃手机更换图片的效果。

图 12.20　显示默认的图片

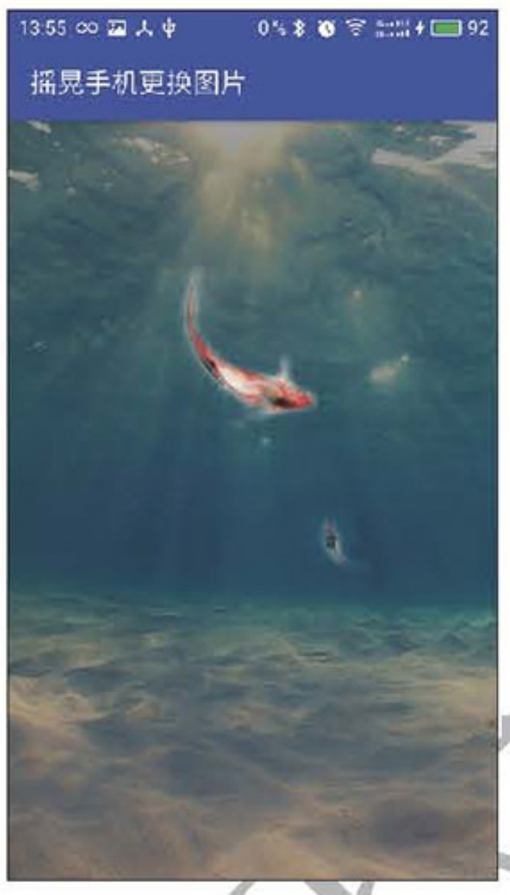


图 12.21　显示更换后的图片

实现过程

（1）打开 Android Studio，新建一个项目，命名为“ShakeForWallpaper”，公司域名为“mingrisoft.com”。

（2）导入资源文件：复制“光盘\Code\12\112\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）打开 MainActivity 类，在该类中创建 register() 方法，用于注册加速度传感器的监听。代码如下：

```
01  private void register() {
02      if (sensorManager == null) {
03          //获取传感器管理服务
04          sensorManager = (SensorManager) getSystemService(SENSOR_SERVICE);
05      }
06      //监听器
07      sensor = new MainActivity.sensorListener();
08      //加速度传感器
09      sensorManager.registerListener(sensor,
10              sensorManager.getDefaultSensor(Sensor.TYPE_ACCELEROMETER),
11              SensorManager.SENSOR_DELAY_NORMAL);
12  }
```

通过 onSensorChanged() 方法实现摇晃手机时判断传感器三个轴的值是否符合更换图片的标准，如果符合则发送更换图片的消息。代码如下：

```
01  public void onSensorChanged(SensorEvent event) {
02      //现在检测时间
03      long currentUpdateTime = System.currentTimeMillis();
04      //两次检测的时间间隔
```

```
05        long timeInterval = currentUpdateTime - lastTime;
06        //判断是否达到了检测时间间隔
07        if (timeInterval < INTERVAL_TIME)
08            return;
09        //现在的时间变成last时间
10        lastTime = currentUpdateTime;
11        int sensorType = event.sensor.getType();
12        //values[0]:x轴，values[1]：y轴，values[2]：z轴
13        float[] values = event.values;
14        if (sensorType == Sensor.TYPE_ACCELEROMETER) {
15            if ((Math.abs(values[0]) > 12 || Math.abs(values[1]) > 12 || Math
16                    .abs(values[2]) > 12)) {
17                flag++;                                  //摇晃一次，该值加1用于更换图片
18                if(flag>3) flag = 0;                     //0~3一共四张图片，该值大于3时从第一张开始
19                mHandler.sendEmptyMessage(0);            //发送更换图片的消息
20            }
21        }
22    }
```

通过 handleMessage() 方法实现根据接收摇晃手机的消息，进行图片的更换。

扩展学习

本实例实现了摇晃手机更换图片的效果。请尝试在 register() 方法中注册震动器服务（Vibrator），然后在 onSensorChanged() 方法中设置震动频率并启动震动器，以实现摇晃手机更换图片的同时手机震动的效果。

明日科技

第13章 网络编程

实例 113 多线程下载腾讯新闻客户端

应用场景：下载文件

源码位置：光盘\Code\13\113

实例说明

在本实例中，将会使用多线程断点续传方式下载来实现文件下载的功能，运行实例，默认效果如图 13.1 所示；单击“立即下载”按钮，开启下载文件功能，随后在界面中将显示出下载进度条，如图 13.2 所示。

图 13.1 默认界面效果

图 13.2 显示下载进度

关键技术

关键技术

实现多线程断点下载的原理非常简单，首先下载文件的链接要支持断点下载。实现断点下载功

能，最关键的就是 RandomAccessFile 类，使用该类才会把分段下载的数据拼接到一起。网络下载使用 HttpURLConnection 类来实现。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“ThreadDownLoad”，公司域名为“mingrisoft.com”。

（2）导入资源文件：复制“光盘\Code\13\113\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）在 MainActivity 类中设置按钮的单击事件，在单击事件被触发后执行下载。代码如下：

代码位置：ThreadDownLoad\app\src\main\java\com\mingrisoft\ThreadDownLoad\MainActivity.java

```
01  //获取下载器
02  final DownLoadManger manger = new DownLoadManger("http://www.imooc.com/mobile/tengxun.apk");
03  //开启子线程
04  new Thread(new Runnable() {
05      @Override
06      public void run() {
07          //开启下载文件功能
08          manger.downLoadFileTest();
09          handler.sendEmptyMessage(100);
10      }
11  }).start();
```

（4）创建 ThreadManger 类，在该类的 run() 方法中执行数据的断点续传下载和文件的写入操作。主要代码如下：

代码位置：ThreadDownLoad\app\src\main\java\com\mingrisoft\ThreadDownLoad\ThreadManger.java

```
01      HttpURLConnection conn = null;
02      InputStream is = null;
03      RandomAccessFile accessFile = null;
04      try {
05          URL url = new URL(urlPath);
06          conn = (HttpURLConnection) url.openConnection();
07          conn.setConnectTimeout(3000);                    //超时时间
08          conn.setRequestMethod("GET");                    //GET请求
09          //断点下载的位置
10          conn.setRequestProperty("Range", "bytes=" + startPosition + "-" + endPosition);
11
12          Log.w("Range", "s :" + startPosition + "-" + endPosition);
13          conn.connect();//打开连接
14          //获取返回值，应该为206，注意这里不是200，返回200是不能断点下载的
15          int code = conn.getResponseCode();    System.out.println("线程ID" + threadId);
16          System.out.println("返回码" + code);
17          is = conn.getInputStream();                      //获取输入流
```

明日科技

```
18          byte[] buff = new byte[1024];                    //设置缓存
19          int len;
20          accessFile = new RandomAccessFile(fileName, "rwd");
21         accessFile.seek(startPosition);                   //设置文件的写入位置
22          while ((len = is.read(buff)) != -1) {
23              accessFile.write(buff, 0, len);
24              progress.get(len, all);                      //该回调接口用来获取进度
25              Log.d("id", threadId+"");
26             }
```

扩展学习

请尝试将加载进度条的颜色更改为其他颜色。

实例 114　使用 WebView 访问明日学院

应用场景：网络数据请求

源码位置：光盘\Code\13\114

实例说明

本实例将使用 WebView 来实现一个浏览器的效果。当程序运行后将加载一个网络界面，效果如图 13.3 所示，并且通过手势可以进行页面的缩放，效果如图 13.4 所示。

图 13.3 默认打开网页

图 13.4　手势缩放

关键技术

本实例使用 WebView 的 loadUrl() 方法加载网页，使用 getSettings() 方法来设置 WebView。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“WebSite”，公司域名为“mingrisoft. com”。

（2）导入资源文件：复制“光盘\Code\13\114\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）在 MainActivity 类的 onCreate() 方法中，首先实例化 WebView 控件，并设置要加载网页的网址，然后获取 WebSettings，并通过 WebSettings 设置 WebView 的显示效果和开启 WebView 的手势缩放。代码如下：

代码位置：WebSite\app\src\main\java\com\mingrisoft\website\MainActivity.java

```
01  public class MainActivity extends AppCompatActivity {
02      private WebView webView;
03      @Override
04      protected void onCreate(Bundle savedInstanceState) {
05          super.onCreate(savedInstanceState);
06          setContentView(R.layout.activity_main);
07          //获取控件
08          webView = (WebView) findViewById(R.id.webview);
09          //设置加载网址
10          webView.loadUrl("http://www.mingrisoft.com/");
11          //设置支持JavaScript
12          webView.getSettings().setJavaScriptEnabled(true);
13          //设置启用视meta标签的支持
14          webView.getSettings().setUseWideViewPort(true);
15          //不显示控制页面缩放的按钮
16          webView.getSettings().setDisplayZoomControls(false);
17          //使用内部缩放机制
18          webView.getSettings().setBuiltInZoomControls(true);
19          //设置支持缩放手势
20          webView.getSettings().setSupportZoom(true);
21          //设置适应设备屏幕
22          webView.getSettings().setLoadWithOverviewMode(true);
23      }
24  }
```

扩展学习

尝试让设置 WebView 显示缩放按钮，看看可以实现什么效果。

实例 115 明日学院课程详情页面

应用场景：网络数据请求

源码位置：光盘\Code\13\115

视频位置：光盘\Video\13\115

视频讲解

实例说明

本实例将实现从网络上获取数据，然后将获取到的数据展示在界面中的功能。运行实例，图 13.5 为运行时的列表展示效果图；图 13.6 为单击列表某数据时显示的效果图，所有数据均为从网上获取。

图 13.5 数据列表

图 13.6 详情界面

关键技术

关键技术

本实例实现网络数据请求时使用 Volley（网络请求框架），并通过使用 Gson 来解析 JSON 数据。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“Course”，公司域名为“mingrisoft. com”。

（2）导入资源文件：复制“光盘\Code\13\115\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）MainActivity 类的 onCreate() 方法中初始化控件和数据，代码如下：

代码位置：Course\app\src\main\java\com\mingrisoft\course\activity\MainActivity.java

```
01    setTitleName(R.string.main_title);                          //设置标题
02    setTitleBarBackgroundColor(R.color.colorWhite);             //设置标题栏背景颜色
03    setTitleNameColor(Color.BLACK);                             //设置标题颜色
04    //初始化控件
05    classList = (ListView) findViewById(R.id.class_list);
```

```
06    //设置Item单击事件
07    classList.setOnItemClickListener(this);
```

创建 StringRequest 网络请求，然后获取 ListView 要加载的数据，在网络请求成功之后，使用 Gson 解析 JSON 数据，最后将 StringRequest 加入到网络请求队列，这样一次普通的 GET 请求方完成了。代码如下：

```
01  //创建网络请求
02  StringRequest request = new StringRequest(Service.first_url,
03              new Response.Listener<String>() {
04      @Override
05      public void onResponse(String response) {
06          //使用Gson解析JSON数据
07          Clazz clazz = new Gson().fromJson(response, Clazz.class);
08          list = clazz.getResult();          //获取数据集
09          //创建适配器绑定数据
10          ClazzAdapter adapter = new ClazzAdapter(list, MainActivity.this);
11          //绑定适配器
12          classList.setAdapter(adapter);
13
14      }
15  }, new Response.ErrorListener() {
16      @Override
17      public void onErrorResponse(VolleyError error) {
18
19      }
20  });
21  requestQueue.add(request);                 //将请求添加到队列中
```

扩展学习

尝试使用 Glide 图片加载库来加载图片，使用 Glide 在加载图片时增加加载等待图片。

实例 116　获取网络菜谱

应用场景：网络数据请求

源码位置：光盘\Code\13\116
视频位置：光盘\Video\13\116

实例说明

本实例将会使用 Retrofit 结合 RxJava 来执行网络数据的下载，同时使用 Gson 解析请求回来的 JSON 数据。运行实例，首先进行网络数据的请求，然后将数据展示到界面中，效果如图 13.7 所示；单击其中一项后跳转界面，并在新的界面中展示获取的详请信息，效果如图 13.8 所示。

明日科技

图 13.7　默认界面效果

图 13.8　菜谱详情界面

关键技术

通过使用 RxJava 的线程调度器与 Retrofit 来实现网络数据的下载。实现这个功能的主要方法就是使用 Observable 类的 subscribeOn() 方法设置事件处理的工作线程，然后再通过 observeOn() 方法切换接收事件的工作线程来实现一个异步操作，最后在 subscribe() 方法中处理订阅事件。限于篇幅，Retrofit 的配置在此不作介绍，具体内容请参考源码。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“CookBook”，公司域名为“mingrisoft. com”。

（2）导入资源文件：复制“光盘\Code\13\116\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）创建 MineApplication 类，在该类中添加 getRetrofit() 方法来配置 Retrofit。代码如下：

代码位置：CookBook\app\src\main\java\com\mingrisoft\cookbook\application\MineApplication.java

```
01  /**
02   * 获取Retrofit对象
03   */
04  public static Retrofit getRetrofit() {
05      if (retrofit == null) {
06          retrofit = new Retrofit.Builder()
07                  .baseUrl(MyService.HTTP_URL)                    //设置网址
08                  //gson转换器
09                  .addConverterFactory(GsonConverterFactory.create())
10                  //rxJava转换器
11                  .addCallAdapterFactory(RxJavaCallAdapterFactory.create())
12                  .build();                                       //创建网络请求对象
```

```
13      }
14      return retrofit;
15  }
```

（4）创建 MyService 接口，用来设置网络请求的模式和网络数据接口。代码如下：

代码位置：CookBook\app\src\main\java\com\mingrisoft\cookbook\service\MyService.java

```
01  public interface MyService {
02      String APP_KEY = "380dfd6393bc4cd0b609b27e0ae0f8e8";          //AppKey
03      String HTTP_URL = "http://api.avatardata.cn/Cook/";           //接口地址
04      String LIST_URL = "List";                                     //菜谱列表
05      String SHOW_URL = "Show";                                     //菜谱信息
06      @POST(MyService.LIST_URL)                                     //用于获取菜谱分类
07      Observable<MineList> getList(@QueryMap Map<String,Object> map);
08      @POST(MyService.SHOW_URL)                                     //用于获取菜谱详情
09      Observable<MineShow> getShow(@QueryMap Map<String,Object> map);
10  }
```

（5）在 MainActivity 类的 onCreate() 方法中，执行网络数据的请求，并将数据展示在列表中。代码如下：

代码位置：CookBook\app\src\main\java\com\mingrisoft\cookbook\MainActivity.java

```
01  //获取网络下载数据对象
02  myService = MineApplication.getMyService();
03  //列表的Item点击事件
04  lists.setOnItemClickListener((parent, view, position, id) ->
05          //跳转界面
06          startActivity(new Intent(this,SecondActivity.class)
07                  .putExtra("show", reasult.get(position))));
08  //进行网络请求
09  myService.getList(initRequest())
10          //网络请求在子线程中执行
11          .subscribeOn(Schedulers.newThread())
12          //线程切换到主线程
13          .observeOn(AndroidSchedulers.mainThread())
14          .subscribe(mineList -> {
15              ult = mineList.getResult();
16              //创建适配器
17              adapter = new MineAdapter(reasult,this);
18              //绑定数据
19              lists.setAdapter(adapter);
20          });
```

扩展学习

尝试将列表的分割线更改为黑色。

实例 117　下载网络图片

应用场景：网络数据请求

源码位置：光盘\Code\13\117

实例说明

本实例使用 HttpUrlConnection 来进行网络数据的请求，以实现下载网络图片的功能。正在下载网络图片的效果如图 13.9 所示；下载完成后的图片效果如图 13.10 所示。

图 13.9　下载图片中

图 13.10　展示下载结束的图片

关键技术

关键技术

本实例主要使用 HttpUrlConnection 来完成网络数据的下载，其中 setRequestMethod() 方法用于设置网络请求模式，getInputStream() 方法用于获取输入流。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“DownLoadPicture”，公司域名为“mingrisoft. com”。

（2）导入资源文件：复制“光盘\Code\13\117\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）在 MainActivity 类的 onCreate() 方法中，初始化控件和弹窗，代码如下：

代码位置：DownloadPicture\app\src\main\java\com\mingrisoft\downloadpicture\MainActivity.java

```
01  @Override
02  protected void onCreate(Bundle savedInstanceState) {
03      super.onCreate(savedInstanceState);
```

```
04        setContentView(R.layout.activity_main);
05        dialog = new ProgressDialog(this);                    //创建弹窗
06        dialog.setTitle("提示");                               //设置弹窗的标题
07        dialog.setMessage("下载图片中,请稍等！");               //设置弹窗提示内容
08        image = (ImageView) findViewById(R.id.image1);
09    }
```

创建 getImageFromNet() 方法，用来执行下载图片的操作。关键代码如下：

```
01    /**
02     * 下载图片
03     * @param downUrl
04     * @return
05     * @throws IOException
06     */
07    private Bitmap getImageFromNet(String downUrl) throws IOException {
08        Bitmap bitmap = null;
09        URL url = new URL(downUrl);                           //设置下载的Url
10        //通过Url开启一个连接
11        HttpURLConnection conn = (HttpURLConnection) url.openConnection();
12        conn.setConnectTimeout(5000);                         //设置超时时间
13        conn.setRequestMethod("GET");                         //设置请求模式
14        conn.setDoInput(true);                                //开启输入流
15        conn.connect();                                       //打开连接
16        if (conn.getResponseCode() == 200){                   //判断返回码
17            InputStream is = conn.getInputStream();           //获取输入流
18            bitmap = BitmapFactory.decodeStream(is);          //将流转换成Bitmap对象
19            conn.disconnect();                                //断开连接
20            is.close();                                       //关闭输入流
21        }
22        return bitmap;                                        //返回Bitmap对象
23    }
```

创建 Handler 对象，用来更新 UI 界面，将下载的图片显示在界面中。代码如下：

```
01    private Handler handler = new Handler(new Handler.Callback() {
02        @Override
03        public boolean handleMessage(Message msg) {
04            if (msg.what == 100){
05                dialog.dismiss();                             //关闭弹窗
06                Bitmap bitmap = (Bitmap) msg.obj;
07                //将下载的图片显示在ImageView中
08                image.setImageBitmap(bitmap);
09            }
10            return false;
11        }
12    });
```

明日科技

扩展学习

尝试更改加载等待弹窗中进度条的颜色。

实例 118	调用系统下载文件 应用场景：网络数据请求	源码位置：光盘\Code\13\118

实例说明

在本实例中将使用系统自带的下载管理器进行文件的下载。运行实例，将显示如图所示 13.11 的界面；单击“立即下载”按钮后，在通知栏中就会收到一个通知，通知名称为“调用系统下载文件”，效果如图 13.12 所示。

图 13.11　默认界面效果

图 13.12　在通知栏中显示正在下载

关键技术

关键技术

本实例主要是通过使用 DownloadManager 对象的 Reguest() 来获取一个下载请求，然后通过 DownloadManager.Request() 方法的设置来配置要执行的文件下载，最后通过 DownloadManager 的 enqueue() 方法来执行文件的下载。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“DownLoadManager”，公司域名为“mingrisoft.com”。

（2）导入资源文件：复制“光盘\Code\13\118\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）在 MainActivity 类中添加 dowLoadApk() 方法，用来设置请求，然后调用系统下载管理执行下载文件的操作。代码如下：

代码位置：DownLoadManager\app\src\main\java\com\mingrisoft\downloadmanager\MainActivity.java

```
/**
 * 该方法是调用了系统的下载管理器
 */
public void downLoadApk(String url) {
    if (isFinish) {
        Toast.makeText(this, "文件正在下载中", Toast.LENGTH_SHORT).show();
        return;
    }
    Uri uri = Uri.parse(url);                                    //下载连接
    //得到系统的下载管理
    manager = (DownloadManager) getSystemService(Context.DOWNLOAD_SERVICE);
    requestApk = new DownloadManager.Request(uri);               //获得连接请求对象
    requestApk.setAllowedNetworkTypes(DownloadManager.Request.NETWORK_MOBILE
            | DownloadManager.Request.NETWORK_WIFI);             //在指定网络下进行下载
    requestApk.setNotificationVisibility(DownloadManager
                    .Request.VISIBILITY_VISIBLE_NOTIFY_COMPLETED);
    requestApk.setDestinationInExternalPublicDir(this.getPackageName() +
            "/DownLoad", "sample.apk");                          //指定下载文件的保存路径
    requestApk.setVisibleInDownloadsUi(true);                    //设置显示下载界面
    //表示允许MediaScanner扫描到这个文件，默认不允许
    requestApk.allowScanningByMediaScanner();
    requestApk.setTitle("调用系统下载文件");                       //设置下载通知栏的提示消息
    requestApk.setDescription("下载腾讯新闻中");                   //设置下载通知栏提示的介绍
    id = manager.enqueue(requestApk);
    registerReceiver(receiver, new IntentFilter(DownloadManager.ACTION_DOWNLOAD_COMPLETE));
    isFinish = true;
}
```

在 MainActivity 类中添加广播接收器，用于监测文件何时完成下载。代码如下：

```
/**
 * 广播接收器
 */
private BroadcastReceiver receiver = new BroadcastReceiver() {
    @Override
    public void onReceive(Context context, Intent intent) {
        if (intent.getAction().equals(DownloadManager.ACTION_DOWNLOAD_COMPLETE)) {
            boolean finish = id == intent.getLongExtra(DownloadManager.EXTRA_DOWNLOAD_ID, 0);
            String result = finish == true ? "下载完成" : "下载未完成";
            isFinish = false;
            Toast.makeText(context, result, Toast.LENGTH_SHORT).show();
        }
    }
};
```

扩展学习

尝试将下载文件的通知栏的标题更改为正在下载 apk 文件。

实例 119	天气预报 应用场景：网络数据请求	源码位置：光盘\Code\13\119 视频位置：光盘\Video\13\119

实例说明

在本实例中将会实现一个天气预报的功能，界面效果如图 13.13 所示；在界面中将会展示设备所在的城市、日期、温度和天气状况，同时还能查看未来 5 天的天气预报，当单击未来 5 天按钮后，将会弹出如图 13.14 所示的窗口，通过滑动来查看未来 5 天的天气预报。

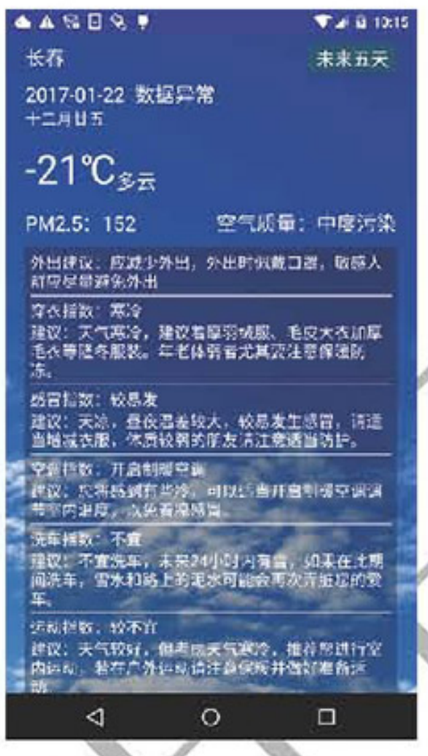
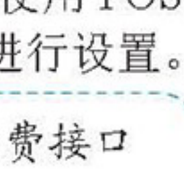


图 13.13　设备所在地理位置天气预报

图 13.14　查看未来 5 天的天气预报

关键技术



本实例使用 Volley 框架进行网络请求，主要使用 Volley 框架来执行 POST 请求操作，使用 POST 请求时需要向后台传递参数，这个参数可以通过重写 StringRequest 类的 getParams() 方法来进行设置。

由于接口是免费提供的，可能出现接口停用或维护等情况。聚合数据天气预报免费接口地址：https://www.juhe.cn/docs/api/id/73。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“Weather”，公司域名为“mingrisoft. com”。

（2）导入资源文件：复制“光盘\Code\13\119\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）将源码文件中的资源文件复制到新创建的项目中（资源目录为“app\res”，该目录中的所有文件复制粘贴到新建项目中）。

（4）在 MianActivity 类中添加 findID() 方法，用来初始化控件。在 onCreate() 方法中调用 findID() 方法，用来获取 Volley 的请求队列并开启定位功能。由于实例中的定位功能使用了百度 SDK，所以需要单独申请百度 key。代码如下：

代码位置：Weather\app\src\main\java\com\mingrisoft\weather\activity\MainActivity.java

```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
    //网络请求队列
    requestQueue = MyApplication.getRequestQueue(this);
    //开启定位获取城市名称
    new GetLocation(this).addCallback(this);
    findID();                                          //获取控件
    dialog = new MyWeatherDialog(this);                //弹窗
}
```

在 getCityName() 方法中获取到当前设备的所在城市定位，然后通过调用 getData() 方法获取当前城市天气预报的信息。代码如下：

```
/**
 * 获取设备所在的城市名称
 * @param cityName
 */
@Override
public void getCityName(String cityName) {
    getData(cityName);                                 //获取天气预报数据
}
```

在 getData() 方法中解析 JSON 数据，将数据存入 Bean 对象中。关键代码如下：

```
@Override
public void onResponse(String response) {
    Log.i(TAG, "onResponse: " + response);
    //解析返回的数据
    Bean = new Gson().fromJson(response,Bean.class);
```

将解析出来的数据展示在控件中。代码如下：

```
chuanyi.setText("穿衣指数：" + dataEntity.getLife().getInfo().getChuanyi().get(0));
ganmao.setText("感冒指数：" + dataEntity.getLife().getInfo().getGanmao().get(0));
kongtiao.setText("空调指数：" + dataEntity.getLife().getInfo().getKongtiao().get(0));
xiche.setText("洗车指数：" + dataEntity.getLife().getInfo().getXiche().get(0));
yundong.setText("运动指数：" + dataEntity.getLife().getInfo().getYundong().get(0));
ziwaixian.setText("紫外线指数：" + dataEntity.getLife().getInfo().getZiwaixian().get(0));
chuanyiJY.setText("建议：" + dataEntity.getLife().getInfo().getChuanyi().get(1));
```

```
08  ganmaoJY.setText("建议：" + dataEntity.getLife().getInfo().getGanmao().get(1));
09  kongtiaoJY.setText("建议：" + dataEntity.getLife().getInfo().getKongtiao().get(1));
10  xicheJY.setText("建议：" + dataEntity.getLife().getInfo().getXiche().get(1));
11  yundongJY.setText("建议：" + dataEntity.getLife().getInfo().getYundong().get(1));
12  ziwaixianJY.setText("建议：" + dataEntity.getLife().getInfo().getZiwaixian().get(1));
```

扩展学习

实现动态获取应用中所需要的定位权限。

实例 120　国内新闻

应用场景：网络数据请求

源码位置：光盘\Code\13\120

实例说明

在本实例中，将会使用 OkHttp 下载网络数据，并将下载的数据展示到界面中。运行实例，将显示如图 13.15 所示的效果；当单击列表中的任意一条数据后跳转界面，并在跳转的界面中显示新闻的详细内容页，效果如图 13.16 所示。

图 13.15　默认界面效果

图 13.16　新闻网页

关键技术

关键技术

在实现本实例时，需要通过 OkHttpClient 类的 newCall() 方法创建一个新的网络请求，并通过执行 enqueue() 方法来开启异步网络请求操作。另外，还需要使用 Handler 将子线程中获取到的数据，传递给主线程。

注意　由于接口是免费提供的，可能出现接口停用或者维护的情况。会导致无法使用该接口的数据。阿凡达数据国内新闻免费接口地址：http://avatardata.cn/。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“News”，公司域名为“mingrisoft. com”。

（2）导入资源文件：复制“光盘\Code\13\120\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）在 MainActivity 类的 onCreate() 方法中做初始化操作，并设置 ListView 的滑动监听事件和 item 单击事件的监听。代码如下：

代码位置：News\app\src\main\java\com\mingrisoft\news\activity\MainActivity.java

```
01  /**
02   * 初始化
03   * @param savedInstanceState
04   */
05  @Override
06  protected void onCreate(Bundle savedInstanceState) {
07      super.onCreate(savedInstanceState);
08      setContentView(R.layout.activity_main);
09      okHttpClient = new OkHttpClient();                    //网络请求对象
10      callBack = new MineCallBack();                        //网络请求回调
11      getDataFromNet(page);                                 //POST网络请求
12      //实例化控件
13      newList = (ListView) findViewById(R.id.news_list);
14      //设置列表的滑动事件
15      newList.setOnScrollListener(new MyScrollListener());
16      //设置列表的item单击事件
17      newList.setOnItemClickListener(this);
18      dialog = new ProgressDialog(this);                    //创建弹窗
19  }
```

在 MainActivity 类中添加 getDataFromNet() 方法，使用 OkHttpClient 的 POST 请求方式请求网络数据。代码如下：

```
01  /**
02   * 获取网络数据
03   */
04  private void getDataFromNet(int getPage) {
05      //请求键值对
06      Map<String, String> map = new HashMap<>();
07      map.put("page", getPage + " ");                       //页数
08      map.put("rows", row + " ");                           //个数
09      map.put("key", Contant.APPKEY);                       //key值
10      FormBody.Builder builder = new FormBody.Builder();
11      //遍历map集合
12      Iterator<Map.Entry<String, String>> iter = map.entrySet().iterator();
13      while (iter.hasNext()) {
14          Map.Entry<String, String> entry = iter.next();
15          String key = entry.getKey();
```

```
        String val = entry.getValue();
        builder.add(key, val);                                  //添加请求参数
    }
    okHttpClient.newCall(new Request.Builder().
            url(Contant.HTTPURL).                               //设置网络请求的链接
            post(builder.build()).                              //设置POST请求
            build()).enqueue(callBack);                         //异步请求
}
```

在回调接口中处理下载成功时的数据，并将数据发送给主线程。代码如下：

```
@Override
public void onResponse(Call call, Response response) throws IOException {
    try {
        Message message = Message.obtain();
        message.what = 1;                                       //设置标识
        //获取数据
        message.obj = getBean(response.body().string());
        handler.sendMessage(message);                           //传递消息
    } catch (JSONException e) {
        e.printStackTrace();
    }
}
```

使用 Handler 处理从子线程中传递出来的消息，并使用消息中的数据创建适配器，然后将适配器绑定给 ListView。关键代码如下：

```
reasults = bean.getReasults();                                  //获取数据
//将数据绑定适配器
adapter = new NewsAdapter(MainActivity.this, reasults);
//将列表绑定适配器
newList.setAdapter(adapter);
```

扩展学习

取消列表中 Item 的单击效果。

实例 121　从本地获取并上传图片

应用场景：网络请求

源码位置：光盘\Code\13\121

实例说明

大部分 App 都有用户个人信息的界面，在该界面中通常会有头像设置功能，需要上传本地头像。

明日科技

本实例将实现一个头像的选择和头像的更改功能。一般头像上传的流程就是将选择的图片传给后台，然后后台向前台返回数据，并通过返回的数据告知前台图片是否上传成功。运行实例，单击“更改头像”按钮，效果如图 13.17 所示；头像更改完成，效果如图 13.18 所示。

图 13.17　单击“更改头像”按钮

图 13.18　头像更改完成

关键技术

本实例中图片的选择，使用了第三方的图片选择库 TakePhoto。网络请求使用的是 OkHttp，用来实现图片的上传，并通过 Request.Builder() 类的 build() 方法来构建 Request 对象。使用 Request.Builder() 类的 url() 方法设置请求的链接，使用 Request.Builder() 类的 post() 方法设置请求参数。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“UpLoad”，公司域名为“mingrisoft. com”。

（2）在项目中添加 TakePhoto 和 OkHttp 的依赖，再设置布局（文件位置：UpLoad\app\build.gradle）。

（3）导入资源文件：复制“光盘\Code\13\121\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（4）在 MainActivity 类的 onCreate() 方法中初始化控件并实例化 TakePhoto 对象。代码如下：

代码位置：UpLoad\app\src\main\java\com\mingrisoft\upload\MainActivity.java

```
01  @Override
02  protected void onCreate(Bundle savedInstanceState) {
03      super.onCreate(savedInstanceState);
04      setContentView(R.layout.activity_main);
05      iamge = (CircleImageView) findViewById(R.id.head);
06      takePhoto = new TakePhoto(this);                    //实例化对象
```

```
        takePhoto.setOnPictureSelected(this);                    //设置回调
        //创建网络请求对象
        okHttpClient = new OkHttpClient.Builder().build();
    }
```

获取到图片的路径后，将图片上传给后台服务器。代码如下：

```
File le = new File(path);                                        //获取文件
RequestBody fileBody = RequestBody.create(MediaType.
                parse("application/octet-stream"), file);
RequestBody requestBody = new MultipartBody.Builder()            //表单
        .setType(MultipartBody.FORM)                             //设置类型
        .addPart(Headers.of(
                "Content-Disposition",
                "form-data; name=\"???\""),
                RequestBody.create(null, "???"))
        .addPart(Headers.of(
                "Content-Disposition",
                "form-data; name=\"???\"; filename=\"image.jpg\""), fileBody)
        .build();                                                //构建
Request request = new Request.Builder()                          //创建请求
        .url("url")                                              //添加请求链接
        .post(requestBody)                                       //添加请求体
        .build();                                                //构建请求
okHttpClient.newCall(request).enqueue(new Callback() {
    @Override
    public void onFailure(Call call, IOException e) {

    }

    @Override
    public void onResponse(Call call, Response response) throws IOException {
        //上传成功后将发送消息让界面中图片刷新
        handler.obtainMessage(1,path).sendToTarget();
    }
});
```

创建 Handler 对象，用来接收消息，然后更新界面中的头像。代码如下：

```
private Handler handler = new Handler(new Handler.Callback() {
    @Override
    public boolean handleMessage(Message msg) {
        if (msg.what == 1){
            String path = (String) msg.obj;
            //显示更改后的图像
            iamge.setImageBitmap(BitmapFactory.decodeFile(path));
```

```
08            }
09            return false;
10        }
11    });
```

扩展学习

请尝试更改状态栏的颜色，让界面的风格达到统一。

明日科技

第14章 与手机系统交互

实例122 吹蜡烛锁屏

应用场景：锁屏软件

源码位置：光盘\Code\14\122

视频位置：光盘\Video\14\122

视频讲解

实例说明

本实例将实现吹蜡烛锁屏的功能。运行实例，将显示如图14.1所示的界面；对着手机的话筒吹气，只要声音大于60分贝，屏幕上的蜡烛会熄灭，暂停一秒后手机将会自动锁屏，如图14.2所示。

图14.1 未锁屏界面

图14.2 蜡烛熄灭，即将锁屏

关键技术

本实例主要使用 AudioRecord 类中的 startRecording() 方法、read() 方法、stop() 方法及 DevicePolicyManager 类中的 lockNow() 方法，实现录音及锁屏的功能。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“OnLock”，公司域名为“mingrisoft.com”。

（2）导入资源文件：复制“光盘\Code\14\122\src”文件中的图片资源，粘贴到“app\res”路径下的 mipmap 文件中，用于蜡烛点燃与熄灭的图片切换。

（3）创建 AudioRecordManger 类文件，用来获取声音并转换成分贝值。

（4）在 AudioRecordManger 类中开启一个线程，实现一秒获取十次声音，关键代码如下：

```
01  //使用新线程
02  new Thread(new Runnable() {
03      @Override
04      public void run() {
05          mAudioRecord.startRecording();                    //录音启动
06          short[] buffer = new short[BUFFER_SIZE];          //设置缓存数组
07          while (isGetVoiceRun) {
08             //r是实际读取的数据长度，一般而言r会小于buffersize
09              int r = mAudioRecord.read(buffer, 0, BUFFER_SIZE);
10              long v = 0;
11              //将buffer内容取出，进行求平方和运算
12              for (int i = 0; i < buffer.length; i++) {
13                  v += buffer[i] * buffer[i];
14              }
15              double mean = v / (double) r;                 //平方和除以数据总长度，得到音量大小
16              double volume = 10 * Math.log10(mean);
17              //大概一秒十次，锁
18              synchronized (mLock) {
19                  try {
20                      mLock.wait(500);
21                  } catch (InterruptedException e) {
22                      e.printStackTrace();
23                  }
24              }
25              //声明消息类，用于将句柄发送消息到主窗体函数
26              Message message = Message.obtain();
27              message.what = mWhat;
28              message.obj = volume;
29              mHandler.sendMessage(message);
30          }
31          //话筒对象释放
32          if (null !=mAudioRecord) {
```

明日科技

```
                mAudioRecord.stop();
                mAudioRecord.release();
                mAudioRecord = null;
            }
        }
}).start(); //启动线程
```

（5）在 res 文件夹下创建 xml 文件夹，在该文件夹中创建名称为 lock_screen.xml 的文件，用来获取管理员权限。关键代码如下：

```
<device-admin
    xmlns:android="http://schemas.android.com/apk/res/android">
    <uses-policies>
        <!-- 强行锁定>
        <force-lock />
        <!-- 清除所有数据（恢复出厂设置） -->
        <wipe-data />
        <!-- 重置密码 -->
        <reset-password />
        <!-- 限制密码选择 -->
        <limit-password />
        <!-- 监控登录尝试 -->
        <watch-login />
    </uses-policies>
</device-admin>
```

（6）在 MainActivity 中让用户打开管理员权限，代码如下：

```
//创建一个Intent
Intent intent = new Intent(DevicePolicyManager.ACTION_ADD_DEVICE_ADMIN);
ComponentName mDeviceAdminSample = new ComponentName(this,MyAdmin.class);
intent.putExtra(DevicePolicyManager.EXTRA_DEVICE_ADMIN,mDeviceAdminSample);
//劝说用户开启管理员权限
intent.putExtra(DevicePolicyManager.EXTRA_ADD_EXPLANATION,"请打开管理员权限");
startActivity(intent);
```

（7）在 MainActivity 中获取分贝值的大小，当大于 60 分贝时，执行以下的操作，代码如下：

```
double i = (double) msg.obj;
if (i >=60){                                    //当分贝大于60时
    ComponentName who = new ComponentName(MainActivity.this, MyAdmin.class);
    if (dpm.isAdminActive(who)) {
        dpm.lockNow();                          //锁屏
        dpm.resetPassword(" ", 0);              //设置屏蔽密码

```

明日科技

```
08      } else {
09          Toast.makeText(MainActivity.this, "还没有打开管理员权限",Toast.LENGTH_SHORT).show();
10          return false;
11      }
12  }
```

扩展学习

本实例中当声音达到一定分贝时，手机自动锁屏，请尝试使手机在锁屏时弹出一个对话框，询问用户是否同意，再进行锁屏操作。

实例 123　短信发送状态

应用场景：发送短信的消息机制

源码位置：光盘\Code\14\123

实例说明

在使用手机发送短信时，短信会弹出一个提示，对方收到短信时又会弹出一个提示。本实例将实现监测短信发送状态。运行实例，将显示如图 14.3 所示的界面；输入电话号码和短信内容，如图 14.4 所示；单击“发送短信”按钮，将显示如图 14.5 所示的界面。

图 14.3　原始界面

图 14.4　等待发送指令

图 14.5　短信已发出

关键技术

本实例主要使用 SmsManager 类中的 sendTextMessage() 方法、TextView 类中的 append() 方法，实现发送短信与显示短信发送状态。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“SMSType”，公司域名为“mingrisoft.com”。

（2）导入资源文件：复制“光盘\Code\14\123\src”文件中的图片资源，粘贴到“app\res”目录下的 mipmap 文件中。

（3）在“app\res\values”文件夹的 styles 文件中编写顶部状态栏的属性。

（4）创建绘制 activity_main 布局文件，并且在该布局文件中绘制出电话号码与短信内容的输入框。

（5）在 MainActivity 中定义一个内部的广播，用来获取发送短信的状态，广播的代码如下：

```
01  /* 自定义mServiceReceiver重写BroadcastReceiver监听短信状态信息 */
02  public class mServiceReceiver extends BroadcastReceiver {
03      @Override
04      public void onReceive(Context context, Intent intent) {
05          //TODO Auto-generated method stub
06          if (intent.getAction().equals(SMS_SEND_ACTIOIN)) {
07              try {
08                  /* android.content.BroadcastReceiver.getResultCode()方法 */
09                  switch (getResultCode()) {
10                      case Activity.RESULT_OK:
11                          /* 发送短信成功 */
12                          mMakeTextToast(getResources().getText(R.string.str_sms_sent_success).
13                                  toString(),true);
14                          break;
15                      case SmsManager.RESULT_ERROR_GENERIC_FAILURE:
16                          /* 发送短信失败 */
17                          mMakeTextToast(getResources().getText(R.string.str_sms_sent_failed).
18                                  toString(), true);
19                          break;
20                  }
21              } catch (Exception e) {
22                  mTextView01.setText(e.toString()); //显示异常信息
23                  e.getStackTrace();
24              }
25          } else if (intent.getAction().equals(SMS_DELIVERED_ACTION)) {
26              try {
27                  /* android.content.BroadcastReceiver.getResultCode()方法 */
28                  switch (getResultCode()) {
29                      case Activity.RESULT_OK:
30                          /* 短信已送达 */
31                          mMakeTextToast(getResources().getText(R.string.str_sms_rec_success).
32                                  toString(), true);
```

```
                    break;
                case SmsManager.RESULT_ERROR_GENERIC_FAILURE:
                    /* 短信未送达 */
                    mMakeTextToast(getResources().getText(R.string.str_sms_rec_failed).
                            toString(), true);
                    break;
                }
            } catch (Exception e) {
                mTextView01.setText(e.toString()); //显示异常信息
                e.getStackTrace();
            }
        }
    }
}
```

（6）在 MainActivity 类中，编写“发送短信”按钮的单击事件，在单击“发送短信”按钮后执行发送短信的相关操作，并获取短信的发送状态。关键代码如下：

```
/* 发送SMS短信按钮事件处理 */
mButton1.setOnClickListener(new Button.OnClickListener() {
    @Override
    public void onClick(View arg0) {
        //TODO Auto-generated method stub
        /* 欲发送的电话号码 */
        String strDestAddress = mEditText1.getText().toString();
        /* 欲发送的短信内容 */
        String strMessage = mEditText2.getText().toString();
        /* 建立SmsManager对象 */
        SmsManager smsManager = SmsManager.getDefault();
        //TODO Auto-generated method stub
        try {
            /* 建立自定义Action常数的Intent(给PendingIntent参数之用) */
            Intent itSend = new Intent(SMS_SEND_ACTIOIN);
            Intent itDeliver = new Intent(SMS_DELIVERED_ACTION);
            /*sentIntent参数为传送后接受的广播信息PendingIntent */
            PendingIntent mSendPI = PendingIntent.getBroadcast(
                    getApplicationContext(), 0, itSend, 0);
            /* deliveryIntent参数为送达后接受的广播信息PendingIntent */
            PendingIntent mDeliverPI = PendingIntent.getBroadcast(
                    getApplicationContext(), 0, itDeliver, 0);
            /* 发送SMS短信，注意倒数的两个PendingIntent参数*/
            smsManager.sendTextMessage(strDestAddress, null,
                    strMessage, mSendPI, mDeliverPI);
            mTextView01.setText("开始获取短信的发送状态");
        } catch (Exception e) {
            mTextView01.setText(e.toString());
            e.printStackTrace();
```

```
30        }
31    }
```

（7）在 MainActivity 中对用户不可见或者销毁时，将注册的广播注销，代码如下：

```
01  /* 取消已注册的自定义Receiver */
02  unregisterReceiver(mReceiver01);
03  unregisterReceiver(mReceiver02);
```

扩展学习

本实例中信息的发送状态均显示在 TextView 上，尝试实现在短信发送的不同阶段弹出不同的提示。

实例 124　短信作弊器

应用场景：伪造短信

源码位置：光盘\Code\14\124

实例说明

短信作弊器主要功能如下：用户自定义一条短信与号码，输入准确的时间格式，单击插入短信的按钮，便可在收件箱中看到刚刚定义的短信。运行实例，编辑一条自定义的短信，如图 14.6 所示；单击“插入短信”按钮，将显示如图 14.7 所示的提示；此时，可以在手机的短信箱中查看自定义的短信，如图 14.8 所示。（注意：请合理合法使用该功能，不可使用该功能从事违法活动）

图 14.6　自定义短信内容

图 14.7　插入短信成功后提示

图 14.8　短信箱中查看短信

关键技术

本实例主要使用 TextView 类中的 addTextChangedListener() 方法，TextInputLayout 类中的 setError () 方法、setCounterEnabled() 方法，实现监听文本框的状态及错误提示。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“SMSForgeTure”，公司域名为“mingrisoft.com”。

（2）在“app\res\drawable”文件夹中创建编写按钮错误提示的属性。

（3）修改显示界面的主题样式，去除系统默认添加的 ActionBar 标题栏，在“app\res\values”文件夹中修改 styles.xml 文件，将“DarkActionBar”改为“NoActionBar”。

（4）创建 ShapeLoad 类，主要用于判断时间格式与短信内容是否合理。在该类中定义监听接口，用于监听文字的改变，关键代码如下：

```
/**
 * 定义监听的接口
 **/
public interface TextChangingListener {
    void onChange(Editable s);
}
```

（5）在 MainActivity 类中设置输入时间格式、信息内容与错误提示，并获取输入文本框内的字符的个数，关键代码如下：

```
editTimeLayout.setCounterEnabled(true);                    //获取字符的个数
editTimeLayout.setError("请输入正确的时间格式");              //错误提示
editContentLayout.setCounterEnabled(true);                 //获取字符的个数
editContentLayout.setError("请输入确保短信内容在140字符之内");  //信息提示
```

（6）当单击插入短信的按钮时，判断短信的内容和电话号码是否为空，如果格式正确，提示用户去收信箱中查看，关键代码如下：

```
if (editPhone.getText().toString().isEmpty()) {
    Toast.makeText(MainActivity.this, "请输入电话号码", Toast.LENGTH_SHORT).show();
}
if (editPhone.getText().toString().isEmpty()) {
    Toast.makeText(MainActivity.this, "请输入内容", Toast.LENGTH_SHORT).show();
}
getContentResolver().insert(Uri.parse("content://sms/inbox"), values);
Toast.makeText(MainActivity.this, "短信插入成功，请在收信箱中查看",
                                                    Toast.LENGTH_SHORT).show();
```

（7）在 Android Manifest.xml 文件中需要添加读写短信的权限，代码如下：

```
<uses-permission android:name="android.permission.INTERNET" />    <!--网络权限-->
<uses-permission android:name="android.permission.READ_SMS" />    <!--读取短信的权限-->
<uses-permission android:name="android.permission.WRITE_SMS" />   <!--写入短信的权限-->
```

扩展学习

本实例在插入短信成功后弹出提示框，尝试实现短信插入成功后直接跳转到手机的收信箱。

实例 125 获取系统相关信息

应用场景：了解手机相关配置信息

源码位置：光盘\Code\14\125

实例说明

现在有很多人买手机的时候会向销售人员询问手机的相关配置，本实例将实现获取系统相关信息的功能，让用户对手机配置一目了然。运行实例，将显示如图 14.9 所示的系统信息；向上滑动可以继续查看更多信息，如图 14.10 所示。

图 14.9 获取系统信息（1）

图 14.10 获取系统信息（2）

关键技术

本实例主要使用 StringBuffer 类中的 append() 方法，实现字符串的拼接与显示。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“GetSysMessage”，公司域名为“mingrisoft.com”。

（2）在 MainActivity 中创建一个内部的广播，用于获取手机的电池信息，关键代码如下：

```
01  int current = intent.getExtras().getInt("level");          //获得当前电量
02  int total = intent.getExtras().getInt("scale");            //获得总电量
03  int percent = current * 100 / total;                       //算出剩余电量的百分比
```

（3）在 onCreat() 方法中初始化广播，填写接收手机电量的 action。关键代码如下：

```
/**
 * 初始化获取手机电量的广播
 **/
receiver = new BatteryReceiver();
//获取手机电量的action
IntentFilter filter = new IntentFilter(Intent.ACTION_BATTERY_CHANGED);
registerReceiver(receiver, filter);                    //注册BroadcastReceiver
```

（4）在初始化方法中获取屏幕的管理类，用于获取到手机的宽度和高度以及屏幕的密度。具体代码如下：

```
/**
 * 获取手机屏幕大小
 **/
DisplayMetrics metric = new DisplayMetrics();
getWindowManager().getDefaultDisplay().getMetrics(metric);  //获取当前屏幕
width = metric.widthPixels;                                  //屏幕宽度（像素）
height = metric.heightPixels;                                //屏幕高度（像素）
density = metric.density;                                    //屏幕密度（0.75 / 1.0 / 1.5）
densityDpi = metric.densityDpi;                              //屏幕密度DPI（120 / 160 / 240）
```

（5）通过 StringBuffer 把系统的相关信息拼接到一个字符串中，并显示到 TextView 上。

扩展学习

尝试实现获取当前手机可用内存的功能。

实例 126　风车闹钟

应用场景：定时提醒

源码位置：光盘\Code\14\126

实例说明

自定义一个炫酷的闹钟，实现通过手指在界面上上下滑动来调节时间，选中相应的星期后，在星期的上方出现一个闹钟的图片，再一次单击星期，闹钟消失。运行实例，将显示如图 14.11 所示的界面；当手指滑动时，星期与按钮隐藏，如图 14.12 所示；在图 14.13 所示的界面可以查看所有设置好的闹钟，也可长按每一条的时间删除闹钟。

关键技术

本实例主要使用 Calendar 类中的 get() 方法获取当前系统时间，使用 View 类的 setRotation() 方法设置风车旋转，使用 View 类中的 setVisibility() 方法设置星期的显示与隐藏。

图 14.11　选中星期的界面

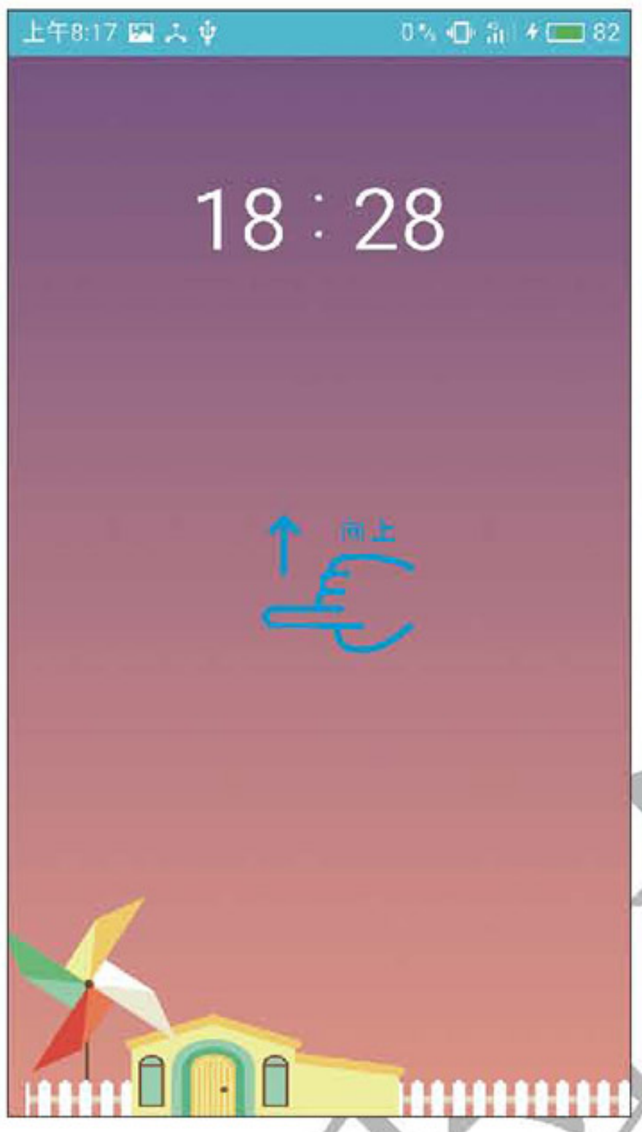


图 14.12　调节时间的界面

图 14.13　显示闹钟的列表界面

实现过程

（1）打开 Android Studio，新建一个项目，命名为“AlarmClock”，公司域名为“mingrisoft.com”。
（2）在“app\res”文件夹中创建 raw 文件夹，用于添加闹铃的音乐。
（3）在“app\res\values”文件夹中创建 attrs 文件，用于设置开关按钮的属性。
（4）在 item 类中创建一个名称为“SwitchButton”的类，用于设置闹钟的开关按钮。
（5）本实例中的 alarmclock 包有 4 个关键类，其作用如图表 14.1 所示。

表 14.1　alarmclock 包的关键类

类　名	描　述
MainActivity	进入闹钟设置时间
AlarmListActivity	显示已经设置完成的闹钟
AlarmReceiver	设置闹钟的服务类
TopActivity	闹钟提示时显示在手机屏幕最上方的界面

（6）创建 MainActivity 类，用于设置闹钟的主界面，显示时间与星期，并且在滑动时隐藏星期与确认按钮，关键代码如下：

```
01  /*
02   * 设置界面的滑动监听
03   **/
```

```
04 scrollLayout.setOnSwipListener(new MyScrollLayout.OnSwipListener() {
05     @Override
06     public void setVisible() {                          //当滑动停止时
07         checkLayout.setVisibility(View.VISIBLE);        //显示星期
08         okBtn.setVisibility(View.VISIBLE);              //显示确认按钮
09     }
10
11     @Override
12     public void setInvisible() {                        //当界面滑动时
13         checkLayout.setVisibility(View.INVISIBLE);      //隐藏星期
14         okBtn.setVisibility(View.INVISIBLE);            //隐藏确认按钮
15     }
16 });
```

（7）创建 AlarmListAdapter 类，用于存放创建好的闹钟。

（8）创建 TopActivity 类，用于在闹钟到时间进行提醒时，弹出的提示界面，关键代码如下：

```
01 mediaPlayer1 = MediaPlayer.create(this, R.raw.congcongnanian);   //初始化mediaPlayer
02 mWakelock.acquire();                                             //开启闹钟
03 mediaPlayer1.seekTo(0);                                          //音乐从头开始
04 mediaPlayer1.start();                                            //播放音乐
```

扩展学习

尝试实现让风车向相反方向旋转的功能。

实例 127　群发拜年短信

应用场景：发送拜年短信

源码位置：光盘\Code\14\127

实例说明

每逢过年的时候，人们都会忙着发拜年短信，本实例将实现短信群发功能，这里可以从模板中选取想要发送的新年祝辞，也可以自己输入新年祝辞。运行实例，打开界面，如图 14.14 所示；单击某条模板时，输入框会自动输入拜年语，如图 14.15 所示。

关键技术

本实例主要使用 Resource 类中的 obtainTypedArray() 方法、List 类中的 add() 方法、TextUtils 类中的 isEmpty() 方法，实现获取短信模板并添加到列表中的功能。

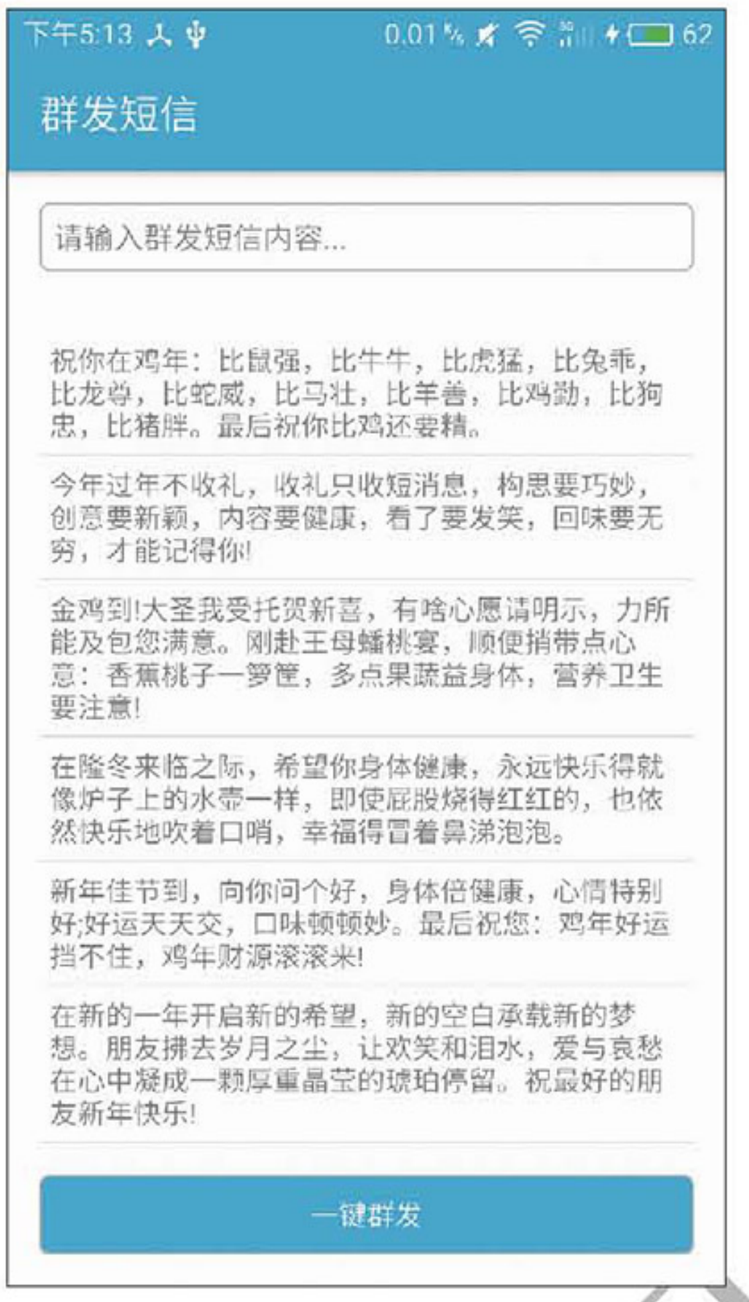


图 14.14　默认界面

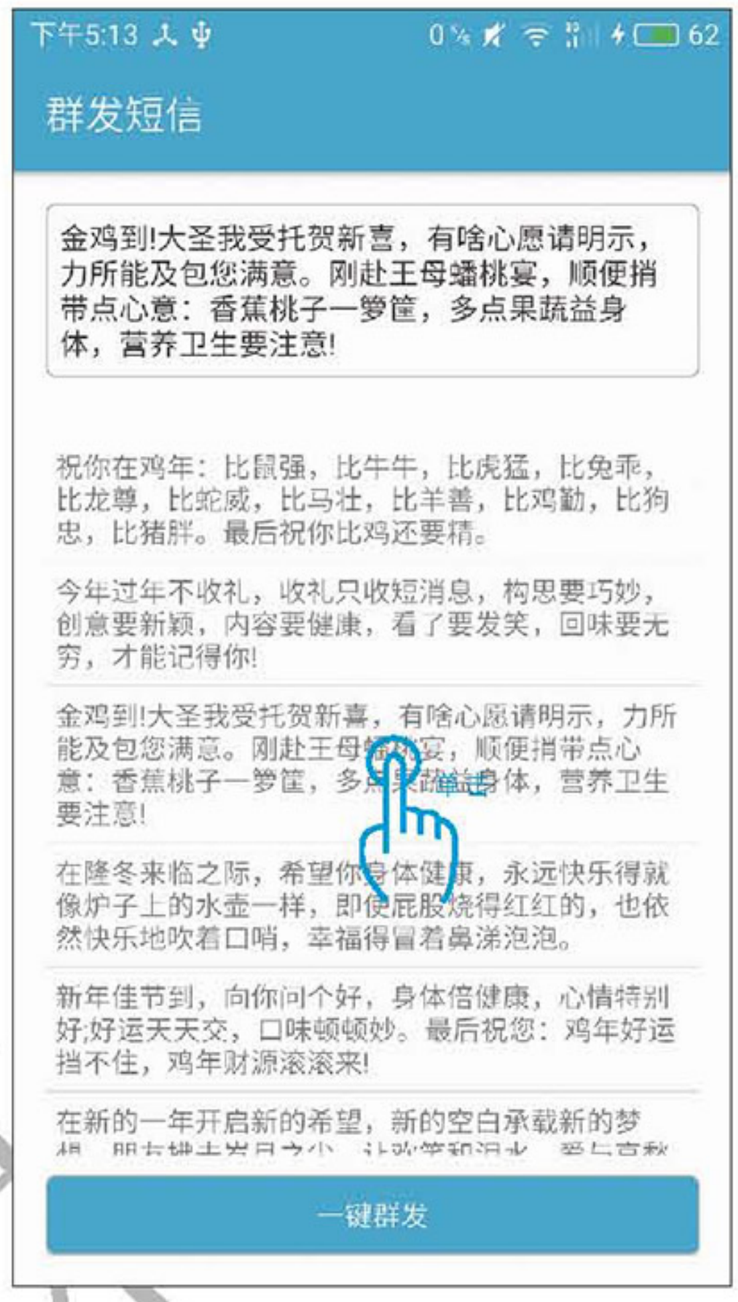


图 14.15　选择拜年语

实现过程

（1）打开 Android Studio，新建一个项目，命名为“SMSAllSend”，公司域名为“mingrisoft.com”。

（2）在“app\res”文件夹中创建名为 arrays 的文件夹，主要用于添加拜年短信的模板。

（3）在“app\res\drawable”文件夹中创建输入框与发送按钮的样式属性。

（4）创建名称为 PersionEntity 的实体类，用于从手机中获取联系人。

（5）在 MainActivity 类中添加获取 arrays 文件夹中的数组数据，并传递给适配器，显示在手机界面上，关键代码如下：

```
01  ar = getResources().obtainTypedArray(R.array.listTitle); //获取数据数组
02  len = ar.length();                                        //获取数组的长度
03  list = new ArrayList<>();                                 //初始化list
04  for (int i = 0; i < len; i++)                             //for循环添加数据
05      list.add(i, ar.getString(i));                         //向ListView中添加数据
06  adapter.addData(list);                                    //将数据传给adapter
```

（6）单击“一键群发”按钮时，判断短信的内容是否为空，如果短信的内容为空，则弹出提示；如果内容不为空，则执行获取联系人并逐条发送短信，关键代码如下：

```
01  case R.id.send_sms:                                       //单击发送按钮
```

```
02  if (TextUtils.isEmpty(smsEdit.getText().toString())) {     //如果内容为空，则弹出提示
03      Toast.makeText(this, "短信内容不能为空", Toast.LENGTH_SHORT).show();
04  } else {
05      getAllPeople();                                          //如果不是空，获取所有联系人
06  }
07  break;
```

（7）获取联系人之后执行 for 循环发送短信，首先判断联系人是否为空，如果为空则弹出提示，如果不为空则执行发送短信的操作，短信发送完成后弹出提示，关键代码如下：

```
01  if (listPersion != null && listPersion.size() > 0) {          //判断手机中是否有联系人
02      for (int i = 0, j = listPersion.size(); i < j; i++) {     //for循环发送短信
03          strings[i] = listPersion.get(i).getNum();             //获取手机号码
04          sendSMS(listPersion.get(i).getNum(), smsEdit.getText().toString()); //给该号码发短信
05      }
06      Toast.makeText(this, "短信全部发送完毕", Toast.LENGTH_SHORT).show();      //弹出提示
07  } else {
08      Toast.makeText(this, "手机联系人为空", Toast.LENGTH_SHORT).show();        //弹出提示
09  }
```

扩展学习

本实例单击“一键群发”按钮后便开始发送短信，请尝试实现单击“一键群发”按钮时询问用户是否继续进行发送的功能。

实例 128　设置黑名单

应用场景：用户不希望被推销电话打扰

源码位置：光盘\Code\14\128
视频位置：光盘\Video\14\128

实例说明

用户可以将指定的电话号设置到黑名单中，从而拒接来电。本实例将实现设置黑名单功能，通过查询数据库，确定是否将来电号码添加到数据库中。如果数据库中存在该号码，则进行挂断操作。运行实例，输入电话号码并单击按钮可将号码添加到数据库中，如图 14.16 所示；单击“查看黑名单”按钮，可打开黑名单列表如图 14.17 所示；在黑名单列表中，单击某一个电话号码，可以从黑名单中移除该号码，如图 14.18 所示。

关键技术

本实例主要使用 TextView 类中的 getText() 方法、AbstractDao 类中的 dleteByKey() 方法，获取输入框的字符串以及删除指定的号码。

图 14.16　输入添加到黑名单的电话号码

图 14.17　单击查看黑名单列表

图 14.18　将号码移出黑名单

实现过程

（1）打开 Android Studio，新建一个项目，命名为“SetBlacklist”，公司域名为“mingrisoft.com”。

（2）在“app\res\drawable”文件夹中创建输入框属性文件。

（3）在 SetBlacklist 文件夹中创建名称为 greendao 的 Mode，该文件为数据库，将黑名单的号码存入该数据库。

（4）创建 MainActivity 类，用于获取添加到黑名单的电话号码，并存到数据库中，关键代码如下：

```
01  /**
02   * 保存到数据库
03   **/
04  private void saveData(String num) {
05      //初始化实体类
06      PersonEntity personEntity = new PersonEntity(preferences.getLong("id",0),
07                                                   preferences.getLong("id",0), null, num);
08      entityDao.insert(personEntity);                          //将数据添加到数据库中
09      editor.putLong("id",preferences.getLong("id",0)+1);      //数据库自增的id
10      editor.commit();                                         //保存
11      //弹出保存完成的提示框
12      Toast.makeText(this, "成功添加到黑名单", Toast.LENGTH_SHORT).show();
13      editText.setText("");                                    //清空输入框
14  }
```

（5）当来电话的时候，遍历数据库，关键代码如下：

```
case TelephonyManager.CALL_STATE_RINGING:          //当手机来电话的时候
    Log.e("------", "响铃:来电号码" + incomingNumber);
    for (int i = 0; i < entityDao.count(); i++) {
        //判断来电的号码是否已经加入到黑名单中
        if (incomingNumber.equals(entityDao.queryBuilder().build()
                .list().get(i).getNumber())) {
            endCall(context);                      //如果来电号码在黑名单中，挂断电话
            break;
        }
    }
    break;
```

（6）如果数据库中有此次来电的号码，执行挂断电话，关键代码如下：

```
Class telephonyClass = telephonyObject.getClass();          //初始化手机管理者
Method endCallMethod = telephonyClass.getMethod("endCall");  //挂断电话
```

（7）在 AndroidManifest 中需添加相应的权限，其中最主要的两个权限如下：

```
<!-- 授予该应用控制通话的权限 -->
<uses-permission android:name="android.permission.CALL_PHONE" />
<!-- 授予该应用读取通话状态的权限 -->
<uses-permission android:name="android.permission.READ_CALL_LOG" />
```

扩展学习

由于手机号最多为 11 位，尝试设置输入框的最大输入长度为 11 个字符。

实例 129　设置免打扰

应用场景：用户不希望被打扰时

源码位置：光盘\Code\14\129
视频位置：光盘\Video\14\129

实例说明

开启免打扰是指让手机显示为静音的状态，可以收到消息，但是不进行提示。关闭免打扰模式的效果如图 14.19 所示，开启免打扰模式的效果如图 14.20 所示。

关键技术

本实例主要使用 AudioManager 类中的 setRingerMode() 方法、setVibrateSetting() 方法，设置手机铃声与震动。

图 14.19　关闭免打扰

图 14.20　开启免打扰

实现过程

（1）打开 Android Studio，新建一个项目，命名为“NoDisturb”，公司域名为“mingrisoft.com”。

（2）导入资源文件：复制“光盘\Code\14\129\src”文件中的图片资源，粘贴到“app\res”目录下的 mipmap 文件中。

（3）在 MainActivity 类中设置免打扰模式，初始化 AudioManager，关键代码如下：

```
aManger = (AudioManager) getSystemService(Service.AUDIO_SERVICE);      //获取系统的音频服务
```

（4）添加控制铃声与震动的功能，关键代码如下：

```
01  if (button.getText().equals("开启")) {
02      Toast.makeText(MainActivity.this, "已关闭免打扰", Toast.LENGTH_SHORT).show();
03      //声音模式
04      aManger.setRingerMode(AudioManager.RINGER_MODE_NORMAL);       //开启手机铃声
05      aManger.setVibrateSetting(AudioManager.VIBRATE_TYPE_NOTIFICATION,
06              AudioManager.VIBRATE_SETTING_ON);                     //开启震动模式
07      aManger.setVibrateSetting(AudioManager.VIBRATE_TYPE_RINGER,
08              AudioManager.VIBRATE_SETTING_ON);                     //开启震动模式
09  } else {
10      Toast.makeText(MainActivity.this, "已开启免打扰", Toast.LENGTH_SHORT).show();
11      //静音模式
12      aManger.setRingerMode(AudioManager.RINGER_MODE_SILENT);       //设置静音模式
```

```
13      aManger.setVibrateSetting(AudioManager.VIBRATE_TYPE_NOTIFICATION,
14              AudioManager.VIBRATE_SETTING_OFF);                    //关闭震动模式
15      aManger.setVibrateSetting(AudioManager.VIBRATE_TYPE_RINGER,
16              AudioManager.VIBRATE_SETTING_OFF);                    //关闭震动模式
17  }
```

（5）在 AndroidManifest 中添加震动的权限，关键代码如下：

```
<uses-permission android:name="android.permission.VIBRATE"/>          <!--震动的权限-->
```

扩展学习

尝试获取手机系统的时间并显示出来。

实例 130　一键截屏

应用场景：手机截屏

源码位置：光盘\Code\14\130

实例说明

用户可以在任何界面进行一键截屏，单击手机界面的悬浮窗后，截屏的图片在图库中便可找到。（本实例只针对 Android5.0 及以上版本）。运行实例。首先开启存储权限与悬浮权限，如图 14.21 所示；开启手机截屏权限，如图 14.22 所示；单击截屏的悬浮按钮进行截屏，如图 14.23 所示。

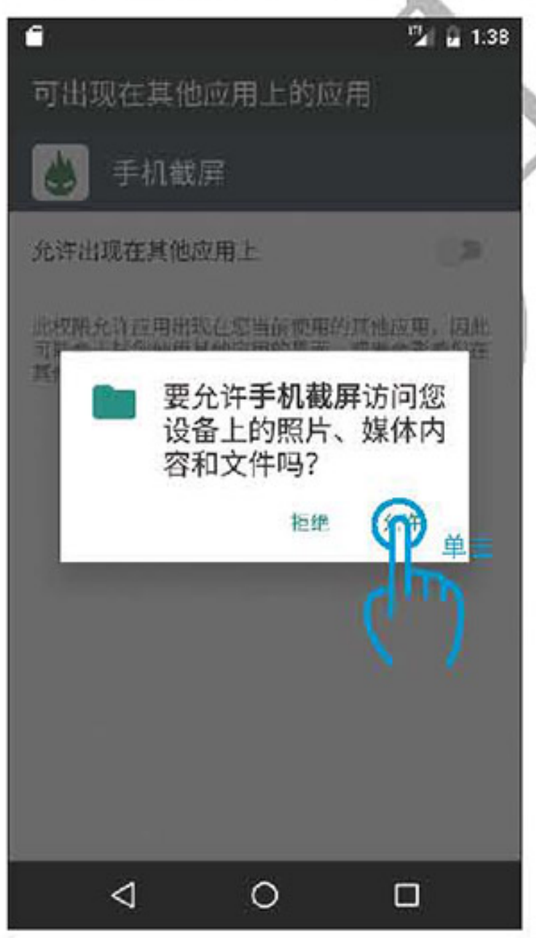


图 14.21　开启截屏权限

图 14.22　正在截屏操作

图 14.23　正在截屏操作

关键技术

本实例主要使用 StringBuffer 类中的 append() 方法、ImageView 类中的 setBirmap() 方法、ViewManager 接口类中的 removeView() 方法，实现截屏及移除悬浮窗的效果。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“PicturePendant”，公司域名为“mingrisoft.com”。

（2）导入资源文件：复制“光盘\Code\14\130\src”文件中的图片资源，粘贴到“app\res”目录下的 mipmap 文件中。

（3）在本实例中的关键类及其作用如表 14.2 所示。

表 14.2　关键类

类　　名	描　　述
GlobalScreenShort	屏幕截取的关键类
FloatWindowService	开启悬浮窗的服务类
FileUtil	截取图片后设置图片的路径及命名的工具类
BaseApplication	加载截取屏幕后的视图

（4）创建 FileUtil 类，主要用于为截取的图片添加名字，此处用到了 StringBuffer 类中的 append() 方法，关键代码如下：

```
/**
 * 设置截屏时为图片添加的名字
 * 并进行拼接
 **/
public static String getScreenShotsName(Context context) {
  SimpleDateFormat simpleDateFormat = new SimpleDateFormat("yyyy-MM-dd-hh-mm-ss");
  String date = simpleDateFormat.format(new Date());                    //初始化日期
  StringBuffer stringBuffer = new StringBuffer(getScreenShots(context)); //初始化StringBuffer
  stringBuffer.append(SCREENSHOT_NAME);
  stringBuffer.append("_");                                             //添加“_”
  stringBuffer.append(date);                                            //添加日期
  stringBuffer.append(".png");                                          //添加后缀
  return stringBuffer.toString();                                       //返回字符串
}
```

（5）在 BaseApplication 类中，设置一个全局的 BitMap 对象，用于加载截屏视图，关键代码如下：

```
private Bitmap mScreenCaptureBitmap;                                    //声明一个用于加载截图时的BitMap
```

（6）将截取的屏幕图片进行保存，首先扫描指定的目录，然后将图片保存到指定的目录下，关键代码如下：

```
01  Intent media = new Intent(Intent.ACTION_MEDIA_SCANNER_SCAN_FILE); //扫描指定的目录
02  Uri contentUri = Uri.fromFile(fileImage);
03  media.setData(contentUri);                                        //将截屏图片进行保存
04  sendBroadcast(media);                                             //保存后发送广播
```

扩展学习

本实例中需要手动设置开启悬浮窗的权限，尝试实现自动获取悬浮窗权限。

实例 131　设置手机屏幕亮度

应用场景：需要根据实际情况调节手机屏幕亮度

源码位置：光盘\Code\14\131

实例说明

在不同强度的光照下，手机需要设置相应的亮度才能使人眼感觉舒适。本实例将实现在手机关闭屏幕亮度自动调节时，显示不同亮度的手机屏幕。运行实例，图 14.24 所示的界面为亮度最暗时的屏幕效果，图 14.25 所示的界面为中等亮度时的屏幕效果，图 14.26 所示的界面为最亮时的屏幕效果。

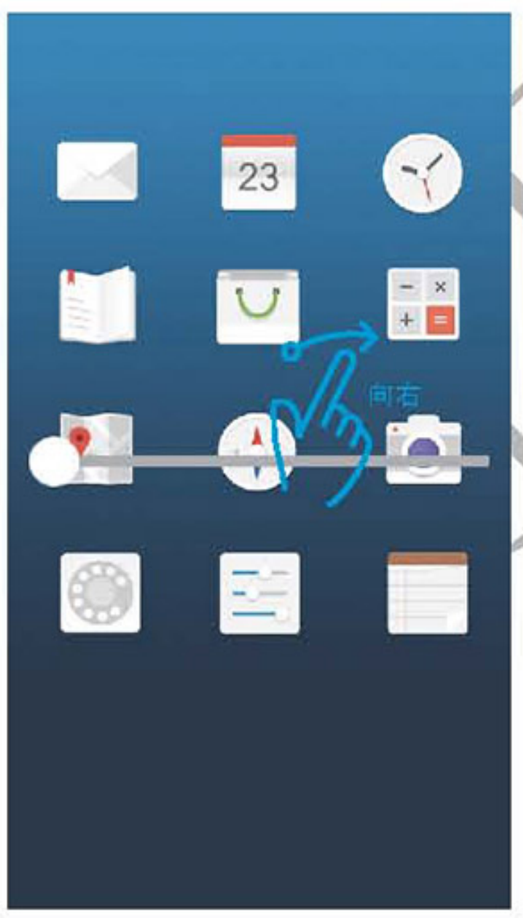


图 14.24　屏幕亮度最低

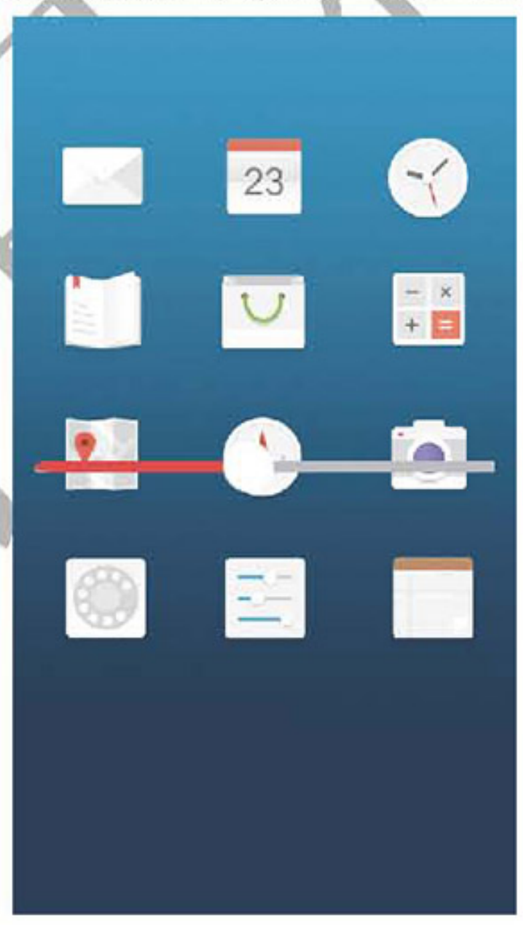


图 14.25　屏幕亮度适中

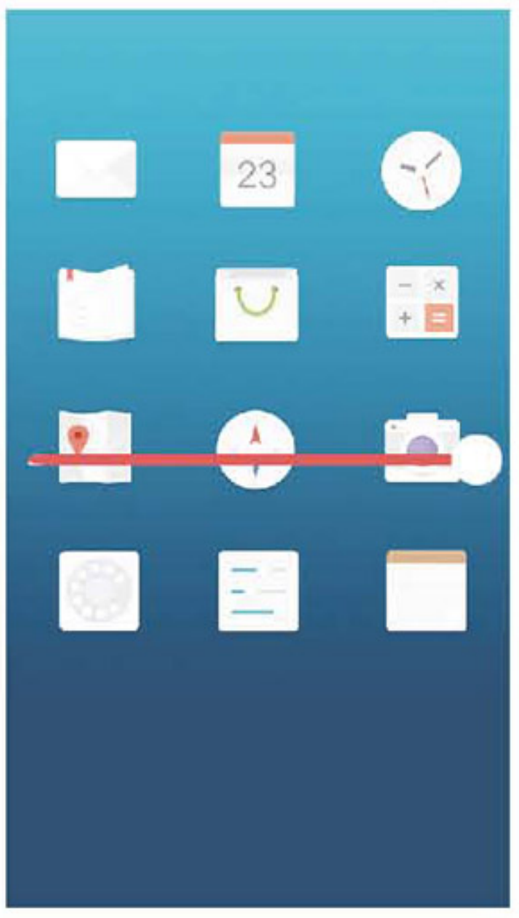


图 14.26　屏幕亮度最亮

关键技术

关键技术

本实例主要使用 SeekBar 类中的 setProgress() 方法、setMax() 方法，Setting 类中的 putInt() 方法，设置进度条的位置及屏幕亮度。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“SetScreenLight”，公司域名为“mingrisoft.com”。

（2）在“app\res\drawable”文件夹中导入相关的图片资源，并设置进度条的属性。

（3）修改界面的主题样式，去除系统默认的 ActionBar 标题栏，打开“app\res\values”目录下的 styles.xml 文件，将“DarkActionBar”修改为“NoActionBar”。

（4）关闭手机自动调节亮度按钮。

（5）创建 BrightnessUtils 类，用于调节屏幕亮度，此类中可以设置自动调节亮度，也可手动调节亮度。

（6）在工具类中介绍两种设置屏幕亮度的方法。

第一种方法：当退出软件后，设置的屏幕亮度也生效的方法，关键代码如下：

```
/**设置系统亮度
 * 程序退出之后亮度设置依然有效
 **/
public static void setSystemBrightness(Context context, int brightness) {
    //异常处理
    if (brightness < 1) {
        brightness = 1;
    }
    //异常处理
    if (brightness > 255) {
        brightness = 255;
    }
    saveBrightness(context, brightness);  //调用保存屏幕亮度的方法
}
```

第二种方法：设置调节屏幕亮度只在本实例中有效，当本实例退出后，屏幕恢复默认的亮度，关键代码如下：

```
//设置亮度
//程序退出之后亮度失效
public static void setCurWindowBrightness(Context context, int brightness) {
    //如果开启自动调节亮度，则需要关闭；否则，亮度值设置无效
    if (IsAutoBrightness(context)) {
        stopAutoBrightness(context);
    }
    //将context转换为Activity
    Activity activity = (Activity) context;
    WindowManager.LayoutParams lp = activity.getWindow().getAttributes();
    //当低于最小值时
    if (brightness < 1) {
        brightness = 1;
    }
    //当高于最大值时
```

明日科技

```
16      if (brightness > 255) {
17          brightness = 255;
18      }
19      lp.screenBrightness = Float.valueOf(brightness) * (1f / 255f);     //设置屏幕的亮度
20      activity.getWindow().setAttributes(lp);                            //屏幕亮度生效
21  }
```

（7）通过拖动条来控制屏幕的亮度，MainActivity 类中与进度条相关的关键代码如下：

```
01  seekBar = (SeekBar) findViewById(R.id.set_light);                      //绑定id
02  seekBar.setProgress(utils.getScreenBrightness(this));                  //获取当前屏幕亮度设置当前进度
03  seekBar.setMax(225);                                                   //设置进度条的最大长度为225
04  /**
05   * 对进度条进行监听
06   **/
07  seekBar.setOnSeekBarChangeListener(new SeekBar.OnSeekBarChangeListener() {
08      @Override
09      public void onProgressChanged(SeekBar seekBar, int i, boolean b) {  //改变中的监听
10          utils.setSystemBrightness(MainActivity.this,i);                //将当前的进度条的值传给系统
11      }
12      @Override
13      public void onStartTrackingTouch(SeekBar seekBar) {                //开始改变时监听
14      }
15      @Override
16      public void onStopTrackingTouch(SeekBar seekBar) {                 //结束改变时监听
17      }
18  });
```

扩展学习

尝试实现通过左右滑动手势来调节屏幕的亮度。

实例 132	通讯录 应用场景：自定义手机联系人列表	源码位置：光盘\Code\14\132

实例说明

手机自带的通讯录格式单一，可能让人感到无聊，本实例将实现一款属于自己的个性化通讯录。运行实例，进入联系人界面，如图 14.27 所示；按字母索引查看联系人信息的效果如图 14.28 所示；使用名称检索的效果如图 14.29 所示。

明日科技

图 14.27　联系人界面

图 14.28　按字母索引查看联系人信息

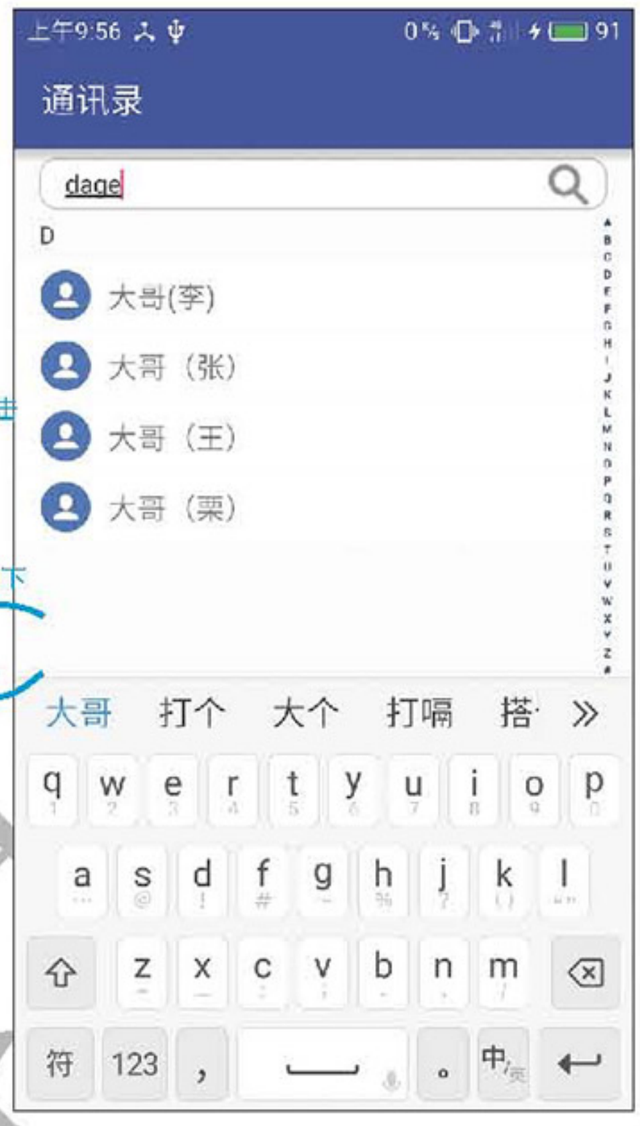


图 14.29　搜索联系人

关键技术

本实例主要使用 TextView 类中的 addTextChangedListener() 方法、List 类中的 add() 方法、BaseAdapter 类的 notifyDataSetChanged() 方法，实现监听输入框的状态及更新列表适配器的功能。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“MaiList”，公司域名为“mingrisoft.com”。

（2）导入资源文件：复制“光盘\Code\14\132\src”文件中的图片资源粘贴到“app\res”文件夹中的 mipmap 文件中。

（3）在“app\res\drawable”文件夹中创建“enter_kuang.xml”文件，用于添加搜索联系人的输入框。

（4）在默认生成的 activity_main.xml 布局文件中绘制布局。

（5）创建 tool 工具包，在该包中完善自定义的导航条与搜索栏以及汉字转拼音的工具类。

（6）创建联系人列表的适配器，在适配器中获取联系人应该出现的位置，关键代码如下：

```
01  /**
02   * 根据ListView的当前位置获取分类的首字母的Char ASCII值
03   */
04  public int getSectionForPosition(int position) {
05      return list.get(position).getSortLetters().charAt(0);
06  }
```

```
/**
 * 根据分类的首字母的Char ASCII值，获取其前字母第一次出现的位置
 */
public int getPositionForSection(int section) {
    for (int i = 0; i < getCount(); i++) {
        String sortStr = list.get(i).getSortLetters();
        char firstChar = sortStr.toUpperCase().charAt(0);
        if (firstChar == section) {
            return i;
        }
    }
    return -1;
}
```

（7）在 ContactsActivity 类中，实例化汉字转拼音的工具类，关键代码如下：

```
//实例化汉字转拼音类
characterParser = CharacterParser.getInstance();
pinyinComparator = new PinyinComparator();
mSortList = new ArrayList<SortModel>();
sideBar.setTextView(dialog);
//设置右侧触摸监听
sideBar.setOnTouchingLetterChangedListener(new SideBar.OnTouchingLetterChangedListener() {
    @Override
    public void onTouchingLetterChanged(String s) {
        //该字母首次出现的位置
        int position = adapter.getPositionForSection(s.charAt(0));
        if (position != -1) {
            sortListView.setSelection(position);
        }
    }
});
SourceDateList = new ArrayList<>();
//根据a-z的顺序将源数据排序
Collections.sort(SourceDateList, pinyinComparator);
adapter = new AdapterContact(this, SourceDateList);
sortListView.setAdapter(adapter);
```

（8）在 ContactsActivity 中获取手机联系人，主要的方法是使用游标读取，将读取出来的联系人用创建好的实体类进行解析。关键代码如下：

```
try {
    //获取手机联系人
    Cursor phoneCursor = resolver.query(ContactsContract.CommonDataKinds.Phone.CONTENT_URI,
```

明日科技

```
04              PHONES_PROJECTION, null, null, null);
05      if (phoneCursor != null) {
06          while (phoneCursor.moveToNext()) {
07              //获取手机号码
08              String phoneNumber = phoneCursor
09                      .getString(1);
10              //当手机号码为空或空字段时，跳过当前循环
11              if (TextUtils.isEmpty(phoneNumber))
12                  continue;
13              //获取联系人名称
14              String contactName = phoneCursor.getString(0);
15              //创建实体类解析联系人
16              PersonEntity mContact = new PersonEntity(contactName, phoneNumber);
17              list.add(mContact);             //添加到集合里
18          }
19          phoneCursor.close();                //关闭游标
20      }
21  } catch (Exception e) {
22      e.printStackTrace();
23  }
```

扩展学习

尝试实现单击联系人列表时，显示联系人姓名和联系人电话。

实例 133　网速监控

应用场景：用于实时监控网速

源码位置：光盘\Code\14\133

实例说明

本实例实现实时监控手机网速。运行实例，将以悬浮球的形式显示当前网速，悬浮球的水波高度会根据网速变化，如图 14.30 和图 14.31 所示；当浏览网页时，也会以悬浮球来显示当前网速，如图 14.32 所示。

关键技术

本实例主要使用 ViewManager 接口类中的 addView() 方法、updataViewLayout() 方法，Canvas 类中的 drawText() 方法、drawCircle() 方法，实现添加移动悬浮窗及绘制实时网速的文字。

明日科技

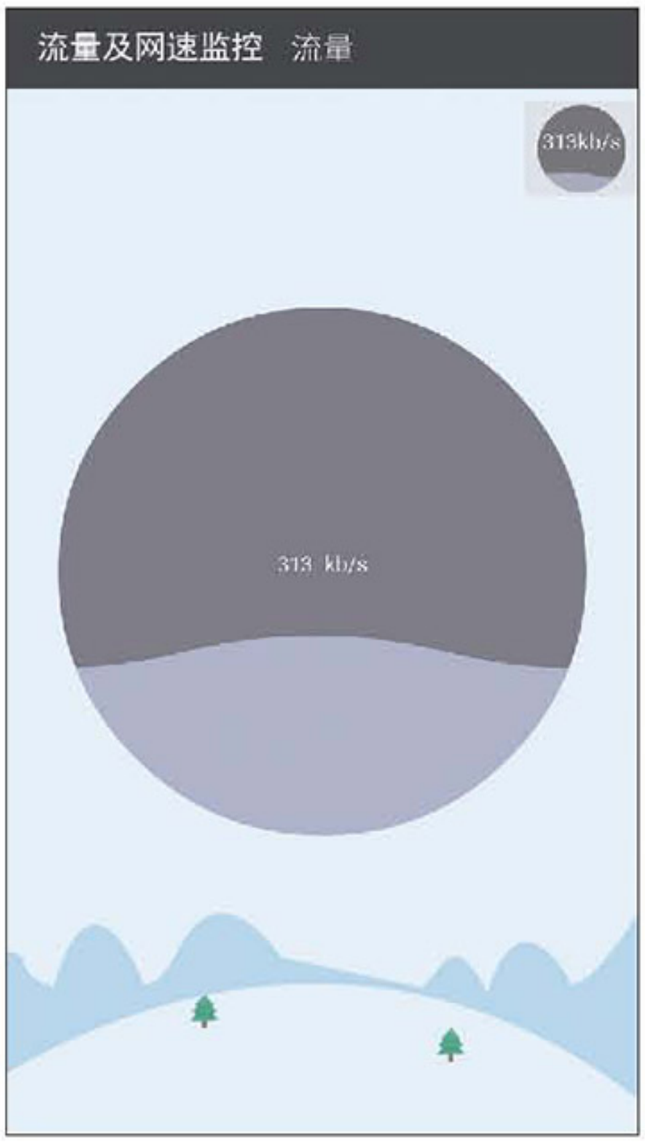


图 14.30　实时网速（一）

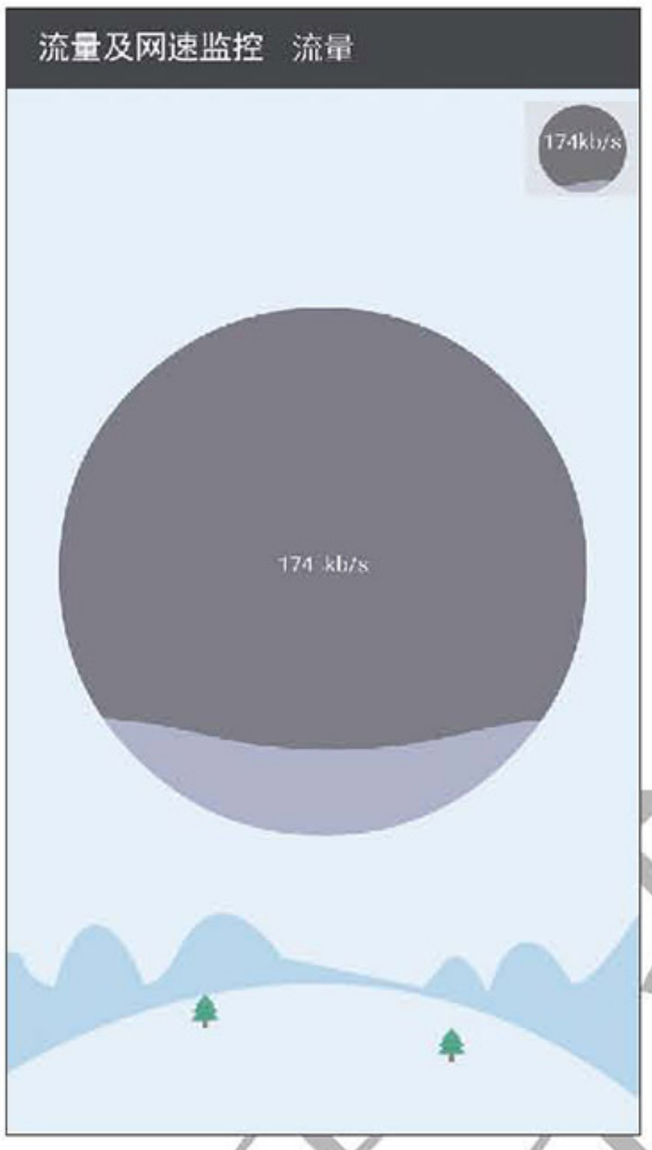


图 14.31　实时网速（二）

图 14.32　实时网速（三）

实现过程

（1）打开 Android Studio，新建一个项目，命名为“PhoneNet”，公司域名为“mingrisoft.com”。

（2）导入资源文件：复制“光盘\Code\14\133\src”文件中的图片资源，粘贴到“app\res”目录下的 mipmap 文件中。

（3）创建一个服务类，命名为“ManagerService”，用于显示悬浮窗，该服务类也可以更新悬浮窗的位置，关键代码如下：

```
01  //设置监听浮动窗口的触摸移动
02  tsp.setOnTouchListener(new OnTouchListener() {
03      @Override
04      public boolean onTouch(View v, MotionEvent event) {
05          //getRawX是触摸位置相对于屏幕的坐标，getX是相对于按钮的坐标
06          wmParams.x = (int) event.getRawX() - tsp.getMeasuredWidth() / 2 - 20;
07          //减80为状态栏的高度
08          wmParams.y = (int) event.getRawY() - tsp.getMeasuredHeight() / 2 - 80;
09          mWindowManager.updateViewLayout(mFloatLayout, wmParams); //刷新
10          return false;       //此处必须返回false，否则OnClickListener无法获取监听
11      }
12  });
```

（4）创建 TrafficInfo 类，用于获取下载与上传的流量以及查询流量的总和，关键代码如下：

```
    long rcvTraffic = UNSUPPORTED;                          //下载流量
    rcvTraffic = TrafficStats.getUidRxBytes(uid);
    if (rcvTraffic == UNSUPPORTED) {                        //不支持的查询
       return UNSUPPORTED;
    }
    RandomAccessFile rafRcv = null, rafSnd = null;          //用于访问数据记录文件
    String rcvPath = "/proc/uid_stat/" + uid + "/tcp_rcv";
    try {
       rafRcv = new RandomAccessFile(rcvPath, "r");
       rcvTraffic = Long.parseLong(rafRcv.readLine());      //读取流量统计
    } catch (FileNotFoundException e) {
       rcvTraffic = UNSUPPORTED;
    } catch (IOException e) {
       e.printStackTrace();                                 //抛出异常
    } finally {
       try {
          if (rafRcv != null)
             rafRcv.close();                                //关闭流
          if (rafSnd != null)
             rafSnd.close();                                //关闭流
       } catch (IOException e) {
       }
    }
```

（5）获取手机的当前网速，并将其 byte 类型数据转换成 int 类型，继而绘制悬浮窗中水波的高度，关键代码如下：

```
if (msg.what == 1) {
   Double a = (Double) msg.obj;
   int b = (int) (a*10);                                    //将网速转换成int类型
   tsp.setWaveHeight(b);                                    //设置水波的高度
   if(service != null)
      service.setSpeed(b);                                  //将int值传给悬浮窗
}
```

扩展学习

本实例中悬浮窗为圆形，为了使手机界面简洁，尝试只显示悬浮窗上的文本。

明日科技

实例 134　悬浮窗

应用场景：随意在桌面放置的悬浮窗

源码位置：光盘\Code\14\134

实例说明

手机安装杀毒软件后，手机的桌面会显示垃圾处理与运行状态的悬浮窗按钮，单击该按钮可以清理垃圾。本实例将实现一个悬浮窗，运行实例，将显示如图 14.33 所示的悬浮窗；按住悬浮窗拖动可以移动悬浮窗的位置，如图 14.34 所示。

图 14.33　已经开启悬浮窗

图 14.34　移动悬浮窗

关键技术

关键技术

本实例主要使用 ViewManager 接口类中的 addView()、updateViewLayout()、removeView() 方法，实现添加视图及移动视图。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“SuspensionWindow”，公司域名为“mingrisoft.com”。
（2）在默认生成的 activity_main.xml 布局文件中绘制布局。
（3）创建 MyService 类文件，继承 Service，然后重写 onBind() 方法。

（4）在 MyService 类中的 onCreat() 方法中，初始化视图和屏幕管理类 WindowParams。关键代码如下：

```
initWindowParams();                                    //初始化WindowParams
initView();                                            //初始化视图
addWindowView2Window();                                //将视图显示出来
initClick();                                           //移动悬浮窗
```

（5）在 MyService 中初始化 WindowManager 的方法中设置悬浮窗的大小，使其保持适中，每次显示的默认位置在左上方。关键代码如下：

```
wmParams.gravity = Gravity.LEFT | Gravity.TOP;
wmParams.width = WindowManager.LayoutParams.WRAP_CONTENT;
wmParams.height = WindowManager.LayoutParams.WRAP_CONTENT;
```

（6）在 MyService 的移动悬浮窗方法中，获取手势的滑动坐标，然后根据坐标点设置 WindowsManager 的位置，关键代码如下：

```
case MotionEvent.ACTION_DOWN:                          //手势按下时
    mStartX = (int) event.getRawX();
    mStartY = (int) event.getRawY();
    break;
case MotionEvent.ACTION_MOVE:                          //手势移动时
    mEndX = (int) event.getRawX();
    mEndY = (int) event.getRawY();
    if (needIntercept()) {
        //getRawX是触摸位置相对于屏幕的坐标，getX是相对于按钮的坐标
        wmParams.x = (int) event.getRawX() - mWindowView.getMeasuredWidth() / 2;
        wmParams.y = (int) event.getRawY() - mWindowView.getMeasuredHeight() / 2;
        mWindowManager.updateViewLayout(mWindowView, wmParams);
        return true;
    }
    break;
case MotionEvent.ACTION_UP:                            //手势抬起时
    if (needIntercept()) {
        return true;
    }
    break;
default:
    break;
```

扩展学习

本实例中为图片的悬浮窗，尝试换成一段文字在手机屏幕上随意移动。

实例 135　监控电池状态

应用场景：电池状态

源码位置：光盘\Code\14\135

实例说明

监控电池状态，包括监测当前手机电池的使用状态与当前的电量。本实例将通过电池管理类来实现对手机电池的监控功能。运行实例，打开应用后在没有充电的状态下显示当前手机的电量，如图 14.35 所示；如果手机充电，将显示当前手机的充电状态与充电模式，如图 14.36 所示。

图 14.35　没有充电的显示状态

图 14.36　充电后的显示状态

关键技术

本实例主要使用 BatteryManager 类、BroadcastReceiver 类中的 onReceive() 方法、ContextWrapper 类中的 registerReceiver() 方法，实现监控电池状态的功能。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“BatteryMonitor”，公司域名为“mingrisoft.com”。

（2）导入资源文件：复制“光盘\Code\14\135\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）打开 MainActivity 类，在 onCreate() 方法中，首先获取显示电池状态的相关控件，然后注册并发送电池变化的广播。代码如下：

```
01  receiver = new BatteryReceiver();          //实例化电池接收器
02  IntentFilter filter = new IntentFilter(Intent.ACTION_BATTERY_CHANGED);
03  registerReceiver(receiver, filter);        //注册BroadcastReceiver
```

（4）创建 BatteryReceiver 内部类，该类继承自 BroadcastReceiver，通过在 onReceive() 方法中获取电池当前的电量、电压、电池温度。代码如下：

```
int current = intent.getExtras().getInt("level");          //获取当前电量
int total = intent.getExtras().getInt("scale");            //获取总电量
battery_voltage.setText("当前电压为：" + intent.getIntExtra("voltage", 0) + "mV");
//获取电池温度
float temperature = intent.getExtras().getInt("temperature") * 0.1f;
//显示电池温度
battery_temperature.setText(intent.getExtras().getInt("temperature")
        * 0.1f + "度");
percent = current * 100 / total;                           //电量百分比
battery_level.setText(percent + "%");                      //显示电量
dlectricity();                                             //根据电量显示对应的电量图
```

判断电源状态，如果是充电状态则显示充电进度图片。代码如下：

```
if(BatteryManager.BATTERY_STATUS_CHARGING==2){
    if(!flag){
        //每500毫秒更换一次图片显示充电进度
        handler.postDelayed(runnable, 500);
        flag = true;                                       //修改充电状态标记
    }
}else{ //否则根据当前电量显示对应的电量图片
    isru = false;
    flag = false;
    handler.removeCallbacks(runnable);
    dlectricity();                                         //根据电量显示对应的电量图
}
```

扩展学习

本实例中已经实现了通过电池管理类监控电池状态的功能，这里将通过在 onReceive() 方法中判断电池温度的方式，实现当电池温度高于 30 摄氏度，以红色字体显示当前温度的功能。

实例 136　手机应用管理

应用场景：管理手机中的应用

源码位置：光盘\Code\14\136

实例说明

应用管理是指可以在手机中扫描出所有的应用程序，并且可以显示这些应用的具体信息。本实例将通过软件管理类来实现手机非系统应用程序的打开与卸载功能。运行实例，将自动获取手机中所有

的应用程序并显示应用的信息列表，如图 14.37 所示；找到指定的应用单击打开按钮，将显示所打开的应用程序界面，如图 14.38 所示；单击卸载按钮，将显示如图 14.39 所示的卸载界面，弹出卸载提示。

图 14.37　显示应用列表

图 14.38　打开指定的应用程序

图 14.39　卸载指定的应用程序

关键技术

本实例主要使用 Context 类中的 getPackageManager() 方法、自定义 AppInstalledAdapter 类中的 setList() 方法、BaseAdapter 类中的 notifyDataSetChanged() 方法，实现手机应用的管理功能。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“ApplicationManagement”，公司域名为“mingrisoft. com”。

（2）导入资源文件：复制“光盘\Code\14\136\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）在“ApplicationManagement\app\src\main\java”目录中创建 util 包，在该包中分别创建 AppInfo 类与 Utils 类。AppInfo 类用于保存应用程序的信息。Utils 类是工具类，在该类中通过 getAllApkList() 方法获取手机中已经安装的应用信息。代码如下：

```
01  public static List<PackageInfo> getAllApkList(Context context) {
02      //保存所有的应用包信息
03      List<PackageInfo> apps = new ArrayList<PackageInfo>();
04      //查找所有的应用包
05      PackageManager pm = context.getPackageManager();
06      //获取系统中已安装应用包的信息
07      List<PackageInfo> packageInfoList = pm.getInstalledPackages(0);
08      for (int i = 0; i < packageInfoList.size(); i++) {
09          PackageInfo packageInfo = (PackageInfo) packageInfoList.get(i);
```

```
        //不添加本应用包信息
        if (!packageInfo.packageName.equals(context.getPackageName())) {
            apps.add(packageInfo);                    //添加应用包信息
        }
    }
    return apps;
}
```

通过 openPackage() 方法实现打开应用的功能，代码如下：

```
public static boolean openPackage(Context context, String packageName) {
    try {
        Intent intent = new Intent();                 //创建intent对象
        //获取程序包的名称，实现开启功能
        intent = context.getPackageManager().getLaunchIntentForPackage(packageName);
        if (intent != null) {
            //设置以新任务的形式打开
            intent.addFlags(Intent.FLAG_ACTIVITY_NEW_TASK);
            //启动要打开的应用程序
            context.startActivity(intent);
            return true;
        }
    } catch (Exception e) {
        e.printStackTrace();
    }
    return false;
}
```

通过 uninstallApk() 方法实现卸载应用程序的功能。代码如下：

```
public static void uninstallApk(Activity context, String packageName, int requestCode) {
    //卸载程序的包名
    Uri packageURI = Uri.parse("package:" + packageName);
    //创建卸载Intent对象
    Intent uninstallIntent = new Intent(Intent.ACTION_DELETE, packageURI);
    //启动卸载
    context.startActivityForResult(uninstallIntent, requestCode);
}
```

（4）在“ApplicationManagement\app\src\main\java”目录中创建 interfaces 包，在该包中创建 InstalledInterface 接口。该接口是已安装应用单击事件的接口。

（5）在“ApplicationManagement\app\src\main\java”目录中创建 adapter 包，并且在该包中创建 AppInstalledAdapter 类，该类继承自 BaseAdapter，是已安装应用数据列表的适配器。

（6）打开 MainActivity 类，该类继承自 Activity 类，实现 InstalledInterface 接口，在该类中实现已安装应用数据列表的显示，并在列表中实现打开应用、卸载应用的功能。（代码位置：ApplicationManagement\app\src\main\java\com\mingrisoft\applicationmanagement\MainActivity.java）。

扩展学习

本实例已经通过软件管理类实现了应用程序的管理功能，尝试将获取手机内部存储的可用空间显示在屏幕右上角的位置。首先在 Utils 类中创建 getInternalAvailSize() 方法，用于获取手机内部存储的可用空间，然后在 activity_main.xml 布局文件中添加 TextView 控件，用于显示可用控件的数值。最后在 MainActivity 类的 initData() 方法中初始化 getInternalAvailSize() 方法，并在 handleMessage() 方法中实现手机内部存储的可用空间信息的显示。

实例 137　可以吸附在屏幕两侧的悬浮物

应用场景：悬浮物

源码位置：光盘\Code\14\137

实例说明

可以吸附在屏幕两侧的悬浮物是通过开启自定义悬浮窗控件后，显示在手机主屏幕中的悬浮控件。通过手指在屏幕上的触摸可以改变悬浮物的位置。本实例将通过 WindowManager 类来实现移动悬浮物的功能。运行实例，单击显示桌面悬浮物按钮，显示如图 14.40 所示；返回到手机主界面后悬浮物将默认显示在手机屏幕的左侧，如图 14.41 所示；拖曳悬浮物至手机屏幕的右侧，松开手指悬浮物将自动显示在手机屏幕的右侧位置，如图 14.42 所示。

图 14.40　单击“显示悬浮物”按钮

图 14.41　悬浮物默认显示位置

图 14.42　拖曳至手机屏幕右侧

关键技术

关键技术

本实例主要使用 ViewManager 接口中的 addView() 方法、removeView() 方法、updateViewLayout() 方法，View.OnTouchListener 接口中的 onTouch() 方法，实现移动悬浮物的功能。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“SuspendedWindow”，公司域名为“mingrisoft.com”。

（2）导入资源文件：复制“光盘\Code\14\137\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）在“SuspendedWindow\app\src\main\java\com\mingrisoft\suspendedwindow”目录中创建 WindowService 类，该类继承自 Service 类，用于实现移动悬浮物的功能。通过 Handler 消息传递机制为悬浮物添加所显示的布局与布局参数。代码如下：

```
01  private Handler mHandler = new Handler() {
02      @Override
03      public void handleMessage(Message msg) {
04          switch (msg.what) {
05              case WINDOW_SHOW:
06                  if (!isAdd) {                                    //如果悬浮物没有显示
07                      //添加悬浮物布局与布局参数
08                      windowManager.addView(float_show, layoutParams);
09                      //添加显示动画
10                      float_show.startAnimation(show_anim);
11                      isAdd = true;                                //悬浮物显示状态
12
13                  } else {
14                      float_show.startAnimation(show_anim);
15                  }
16                  break;
17              case WINDOW_HIDE:
18                  if (isAdd) {
19                      //在主窗口中删除显示的悬浮物
20                      windowManager.removeView(float_show);
21                      isAdd = false;                               //设置悬浮物没有显示
22                  }
23                  break;
24          }
25      }
26  };
```

通过 onTouch() 方法实现移动悬浮物。代码如下：

```
01  public boolean onTouch(View v, MotionEvent event) {
02      switch (event.getAction()) {
03          case MotionEvent.ACTION_DOWN:
04              //获取按下悬浮物时的坐标
05              lastX = (int) event.getRawX();
06              lastY = (int) event.getRawY();
07              //保存悬浮物在屏幕中的位置
```

```
            paramX = layoutParams.x;
            paramY = layoutParams.y;
            isMove = false;                                    //关闭移动状态
            break;
        case MotionEvent.ACTION_MOVE:
            //获取移动中的坐标
            dx = (int) event.getRawX() - lastX;
            dy = (int) event.getRawY() - lastY;
            //设置移动中的悬浮图标在屏幕中的位置
            layoutParams.x = paramX + dx;
            layoutParams.y = paramY + dy;

            showMoveView();                                    //显示移动中的悬浮物
            isMove = true;                                     //设置悬浮物移动状态
            //更新悬浮物位置
            windowManager.updateViewLayout(float_show, layoutParams);
            break;
        case MotionEvent.ACTION_UP:
            if (isMove) {                                      //如果悬浮物处于拖动状态
                //如果发生移动操作，进行松开处理，避免和单击事件冲突
                int dx1 = (int) event.getRawX();               //获取手指抬起时的坐标
                if (dx1 > (screen_width/2)){                   //横向坐标大于屏幕一半时
                    showRightView();                           //显示桌面右侧图标
                    updateDefaultLeftRight();                  //设置悬浮物的宽和高
                    layoutParams.x = default_right;            //显示在右侧
                } else {                                       //小于屏幕一半时
                    showLeftView();                            //显示左侧悬浮物
                    updateDefaultLeftRight();                  //设置悬浮物的宽和高
                    layoutParams.x = default_left;             //显示在左侧
                }
                layoutParams.y = paramY + dy;
                //更新悬浮物位置
                windowManager.updateViewLayout(float_show, layoutParams);

            }

            break;
    }
    return false;
}
```

（4）打开“SuspendedWindow\app\src\main”目录中的 AndroidManifest.xml 文件，在 application 标签中添加窗口服务标签。代码如下：

```
<service android:name=".WindowService"></service>
```

个别手机需要在应用管理中找到该应用，并在权限管理中开启悬浮物。

扩展学习

本实例通过 WindowManager 类实现了悬浮物功能，当悬浮物当前的位置在大于屏幕宽度的 1/4 且小于 3/4 的区域时，尝试实现更换移动悬浮窗的图标，离开此区域则恢复默认图标。

首先在 WindowService 类中创建 showMoveHorizontalCenterView() 方法，用于设置移动到屏幕横向中间区域时显示的图标，然后在 onTouch() 方法中判断手指移动的区域。如果移动中的悬浮物位置在大于屏幕宽度的 1/4 且小于屏幕宽度的 3/4 区域时，调用 showMoveHorizontalCenterView() 方法更换图标，如果在其他区域将使用默认图标，并计算相反的方向。

明日科技

第15章 位置服务与地图应用

实例 138 获取百度地图

应用场景：地图

源码位置：光盘\Code\15\138

实例说明

百度地图是日常生活中很常见的一种地图，无论是 PC 版和手机版都非常实用。本实例将通过百度地图官网提供的 SDK 来实现 Android 版的百度地图。运行实例，如果手机没有开启网络，那么无法获取到地图信息，将显示如图 15.1 所示的界面；如果开启网络后，将显示如图 15.2 所示的地图信息。

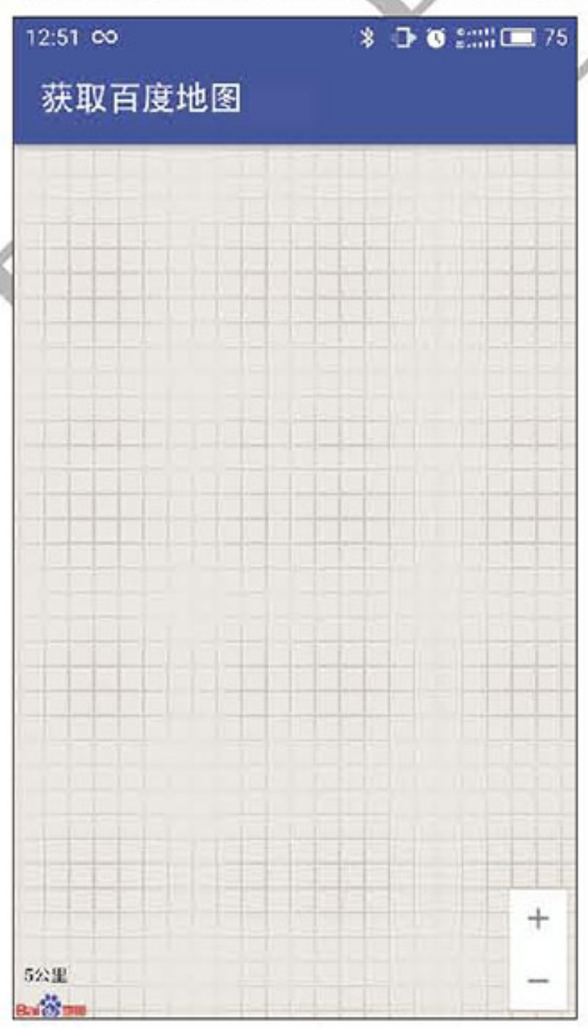


图 15.1　没有获取到地图信息

图 15.2　显示地图信息

关键技术

本实例主要使用 SDKInitializer 类的 initialize() 方法、MapView 类的 onResume() 方法、onPause() 方法、onDestroy() 方法，实现百度地图的运行效果。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“BaiDuMap”，公司域名为“mingrisoft. com”。

（2）打开百度开放平台网站（http://lbsyun.baidu.com/）并注册开放平台账号，然后申请密钥，再下载 Android 地图的 SDK，最后配置环境及发布 Android 地图 SDK。具体操作方法请在百度开发平台的官网中提供的“开发”菜单项中查询。

（3）打开“BaiDuMap\app\src\main”目录中的 AndroidManifest.xml 文件，在该文件中填写百度地图所使用到的权限。代码如下：

```
<!-- 获取运营商信息，用于支持提供运营商信息相关的接口-->
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
<!-- 访问网络，网络定位需要上网-->
<uses-permission android:name="android.permission.INTERNET" />
<!--读取launcher.db内容的权限定义-->
<uses-permission android:name="com.android.launcher.permission.READ_SETTINGS" />
<!--获取锁屏权限-->
<uses-permission android:name="android.permission.WAKE_LOCK"/>
<!--获取Wi-Fi权限，Wi-Fi信息会用来进行网络定位-->
<uses-permission android:name="android.permission.CHANGE_WIFI_STATE" />
<!-- 用于访问Wi-Fi网络信息，Wi-Fi信息会用于进行网络定位-->
<uses-permission android:name="android.permission.ACCESS_WIFI_STATE" />
<!--获取运行任务的信息-->
<uses-permission android:name="android.permission.GET_TASKS" />
<!--允许应用程序写入到外部存储器-->
<uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE" />
<!--允许应用程序读取或写入系统设置-->
<uses-permission android:name="android.permission.WRITE_SETTINGS" />
```

在 <application> 标签中添加开发密钥与百度定位服务。代码如下：

```
<!--在value中设置开发者申请的AK-->
<meta-data
    android:name="com.baidu.lbsapi.API_KEY"
    android:value="Ebkj9W1DPXo0unh8SwW810Vpp69exWs8" />
<!--设置定位服务-->
<service
    android:name="com.baidu.location.f"
    android:enabled="true"
    android:process=":remote"/>
```

（4）打开“BaiDuMap\app\src\main\res\layout”目录下默认生成的 activity_main.xml 布局文件，在该文件中添加地图控件。代码如下：

```
<!--设置显示地图控件-->
<com.baidu.mapapi.map.MapView
    android:id="@+id/bmapView"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:clickable="true" />
```

（5）打开 MainActivity 类，在该类的 setContentView() 方法的上面初始化上下文信息，然后获取地图控件并引用，最后通过 Activity 的生命周期来管理地图的生命周期。代码如下：

```
public class MainActivity extends AppCompatActivity {
    MapView mMapView = null;        //定义地图控件
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        //在使用SDK各组件之前初始化context信息，传入ApplicationContext
        //注意该方法要在setContentView方法之前实现
        SDKInitializer.initialize(getApplicationContext());
        setContentView(R.layout.activity_main);
        //获取地图控件引用
        mMapView = (MapView) findViewById(R.id.bmapView);
    }
    @Override
    protected void onDestroy() {
        super.onDestroy();
        //在Activity执行onDestroy()方法时执行mMapView.onDestroy()方法，实现地图生命周期管理
        mMapView.onDestroy();
    }
    @Override
    protected void onResume() {
        super.onResume();
        //在Activity执行onResume()方法时执行mMapView.onResume()方法，实现地图生命周期管理
        mMapView.onResume();
    }
    @Override
    protected void onPause() {
        super.onPause();
        //在Activity执行onPause()方法时执行mMapView.onPause()方法，实现地图生命周期管理
        mMapView.onPause();
    }
}
```

扩展学习

本实例实现了获取百度地图的功能，请尝试在 MainActivity 类的 onCreate() 方法中获取地图的控制器并开启热力图，实现显示百度热力地图的运行效果。

明日科技

实例 139 使用百度地图定位自己的位置

应用场景：地图定位

源码位置：光盘\Code\15\139

实例说明

地图定位可以定位自己的地理位置。本实例将通过百度地图官网提供的 SDK 来实现 Android 版百度地图的定位功能。运行实例，开启网络后获取地图，默认显示的位置为北京天安门，如图 15.3 所示；单击“定位”按钮后将显示定位后的当前位置，如图 15.4 所示。

图 15.3 默认显示的位置

图 15.4 显示定位后的当前位置

关键技术

关键技术

本实例主要使用 MapView 类中的 getMap() 方法、BaiduMap 类中的 setMyLocationEnabled() 方法、BDLocationListener 接口中的 onReceiveLocation() 方法，实现百度定位的效果。

> **注意** 百度地图开发平台要求在创建一个新的项目（包名与项目名称有变化）时重新申请新的密钥。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“BaiDuLocation”，公司域名为“mingrisoft. com”。

（2）打开百度开放平台网站（http://lbsyun.baidu.com/）并注册开放平台账号，然后申请密钥，再下载 Android 定位 SDK，最后配置环境及发布 Android 地图 SDK。具体操作方法请在百度开发平台官方网站的“开发”菜单项中查询。

（3）导入资源文件：复制“光盘\Code\15\139\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（4）打开“BaiDuLocation\app\src\main”目录中的 AndroidManifest.xml 文件，在该文件中首先填写百度地图定位所使用到的权限，然后在 application 标签中添加开发密钥与百度定位服务。

（5）打开 MainActivity 类，在该类中创建 initLocation() 方法，用于配置定位 SDK 各配置参数。代码如下：

```
01  private void initLocation() {
02      //定位参数对象
03      LocationClientOption option = new LocationClientOption();
04      //设置定位模式：高精度，低功耗
05      option.setLocationMode(LocationClientOption.LocationMode.Hight_Accuracy);
06      option.setCoorType("bd09ll");          //设置返回的定位结果坐标系
07      int span = 1000;
08      //设置发起定位请求的间隔大于或等于1000ms才有效
09      option.setScanSpan(span);
10      option.setIsNeedAddress(true);         //设置是否需要地址信息，默认不需要
11      option.setOpenGps(true);               //设置是否使用GPS，默认false
12      //设置是否当GPS有效时按照每秒1次的频率输出GPS结果，默认false
13      option.setLocationNotify(true);
14      //可以在BDLocation.getLocationDescribe里得到，结果类似于“在北京天安门附近”
15      option.setIsNeedLocationDescribe(true);
16      //设置是否需要POI结果，默认false
17      option.setIsNeedLocationPoiList(true);
18      //定位SDK内部是一个SERVICE，默认false
19      //设置是否在stop的时候杀死这个进程，默认不杀死
20      option.setIgnoreKillProcess(false);
21      //设置是否收集CRASH信息，默认收集
22      option.SetIgnoreCacheException(false);
23      //设置是否需要过滤GPS仿真结果，默认需要
24      option.setEnableSimulateGps(false);
25      mLocationClient.setLocOption(option);
26  }
```

创建 MyLocationListener() 方法并实现 BDLocationListener 接口，重写 onReceiveLocation() 方法实现获取返回的定位数据。代码如下：

```
01  public class MyLocationListener implements BDLocationListener {
02      @Override
03      public void onReceiveLocation(BDLocation location) {
04          //MapView 销毁后不再处理新接收的位置
05          if (location == null || mMapView == null)
06              return;
07          MyLocationData locData = new MyLocationData.Builder()
```

```
                .accuracy(location.getRadius())
                //此处设置开发者获取到的方向信息，顺时针旋转0°-360°
                .direction(100).latitude(location.getLatitude())
                .longitude(location.getLongitude()).build();
        mBaiduMap.setMyLocationData(locData); //设置定位数据
        if (isFirstLoc) {
            //设置定位的坐标
            LatLng ll = new LatLng(location.getLatitude(),location.getLongitude());
            //设置地图中心点以及缩放级别
            MapStatusUpdate u = MapStatusUpdateFactory.newLatLngZoom(ll, 19);
            mBaiduMap.animateMapStatus(u);
            isFirstLoc = false;                    //设置isFirstLoc的属性为false，防止无法移动地图
        }
    }
}
```

扩展学习

本实例已经实现了百度地图的定位效果，尝试通过在 initLocation() 自定义方法中添加自定义图标，实现自定义定位的图标。

实例 140　百度地图公交查询

应用场景：公交站点查询

源码位置：光盘\Code\15\140

实例说明

公交线路查询是平时生活中很常用的一个功能。本实例将通过百度地图官网提供的 SDK 实现百度地图公交线路查询的功能。运行实例，开启网络，填写需要查询的城市与公交线路默认为北京市 717 路公交车，效果如图 15.5 所示；单击“开始”按钮，在地图中将显示想要查询的公交线路与站点，如图 15.6 所示；单击“上一站”与“下一站”按钮将显示站点的站名，如图 15.7 所示。

关键技术

关键技术

本实例主要使用 PoiSearch 搜索类中的 searchInCity() 方法、OnGetPoiSearchResultListener 接口中的 onGetPoiResult() 方法、OnGetBusLineSearchResultListener 接口中的 onGetBusLineResult() 方法，实现百度公交线路查询。

注意　百度地图开发平台要求在创建一个新的项目（包名与项目名称有变化）时重新申请新的密钥。

图 15.5　输入查询的城市与公交

图 15.6　显示公交线路

图 15.7　显示站点名称

实现过程

（1）打开 Android Studio，新建一个项目，命名为“BaiDuBusLine”，公司域名为“mingrisoft.com”。

（2）打开百度开放平台网站（http://lbsyun.baidu.com/）并注册开放平台账号，然后申请密钥，再下载 Android 地图 SDK，最后配置环境及发布 Android 地图 SDK。具体操作方法请在百度开发平台官方网站的“开发”菜单项中查询。

（3）导入资源文件：复制“光盘\Code\15\140\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（4）参考实例 138“百度地图”中所使用的权限。

（5）在“BaiDuBusLine\app\src\main\java\com\mingrisoft\baidubusline”目录中创建 OverlayManager 类，用于管理公交站点中的覆盖物。

（6）创建 BusLineOverlay 类，用于显示公交站点的覆盖物。

（7）打开 MainActivity 类，让该类继承自 FragmentActivity 并实现 OnGetPoiSearchResultListener、OnGetBusLineSearchResultListener、BaiduMap.OnMapClickListener 三个接口。然后通过 searchButtonProcess() 方法发起 POI 搜索，获取被搜索城市中公交相应线路的 ID。代码如下：

```
01  public void searchButtonProcess(View v) {
02      busLineIDList.clear();                                  //清空公交ID数据
03      busLineIndex = 0;                                       //设置公交下标
04      mBtnPre.setVisibility(View.INVISIBLE);                  //隐藏按钮
05      mBtnNext.setVisibility(View.INVISIBLE);
06      EditText editCity = (EditText) findViewById(R.id.city);     //填写城市的编辑框
07      //填写公交的编辑框
08      EditText editSearchKey = (EditText) findViewById(R.id.searchkey);
```

```
09      //发起POI检索，从得到所有POI中找到公交线路类型的POI，再使用该POI的uid进行公交详情搜索
10      mSearch.searchInCity((new PoiCitySearchOption()).city(
11              editCity.getText().toString())
12                      .keyword(editSearchKey.getText().toString()));
13  }
```

通过 onGetPoiResult() 方法遍历所有 POI，找到类型为公交线路的 POI。代码如下：

```
01  public void onGetPoiResult(PoiResult result) {
02      //如果数据为空或出现查询错误，则做出提示
03      if (result == null || result.error != SearchResult.ERRORNO.NO_ERROR) {
04          Toast.makeText(BusLineSearchDemo.this, "抱歉，未找到结果", Toast.LENGTH_LONG).show();
05          return;
06      }
07      //遍历所有POI，找到类型为公交线路的POI
08      busLineIDList.clear();                                  //清除数据
09      for (PoiInfo poi : result.getAllPoi()) {
10          if (poi.type == PoiInfo.POITYPE.BUS_LINE
11                  || poi.type == PoiInfo.POITYPE.SUBWAY_LINE) {
12              busLineIDList.add(poi.uid);                     //添加公交线路的POI
13          }
14      }
15      searchNextBusline(null);                                //设置公交反方向数据空
16      route = null;                                           //公交路线数据为空
17  }
```

通过 onGetBusLineResult() 方法在地图中显示公交信息的查询结果。代码如下：

```
01  public void onGetBusLineResult(BusLineResult result) {
02      //如果数据为空或出现查询错误，则做出提示
03      if (result == null || result.error != SearchResult.ERRORNO.NO_ERROR) {
04          Toast.makeText(BusLineSearchDemo.this, "抱歉，未找到结果", Toast.LENGTH_LONG).show();
05          return;
06      }
07      mBaiduMap.clear();                                      //清空地图覆盖物
08      route = result;                                         //保存浏览数据
09      nodeIndex = -1;                                         //设置下标
10      overlay.removeFromMap();                                //删除地图覆盖物
11      overlay.setData(result);                                //设置公交数据
12      overlay.addToMap();                                     //将公交站点添加到地图上
13      overlay.zoomToSpan();                                   //缩放地图显示所有公交站点
14      mBtnPre.setVisibility(View.VISIBLE);                    //显示上一站按钮
16      mBtnNext.setVisibility(View.VISIBLE);                   //显示下一站按钮
16      //提示公交线路与终点信息
17      Toast.makeText(BusLineSearchDemo.this, result.getBusLineName(),
18              Toast.LENGTH_SHORT).show();
19  }
```

扩展学习

本实例已经实现了百度地图公交线路的搜索功能，尝试在 MainActivity 类的 nodeClick() 方法中修改代码，在显示公交站点名称时实现文字居中显示的效果。

实例 141　百度全景地图

应用场景：全景地图

源码位置：光盘\Code\15\141

实例说明

百度全景地图可以根据指定的地点查看周围的全景。本实例将通过百度地图官网提供的 SDK 来实现百度全景地图。运行实例，开启无线网络，显示当前手机定位后的全景地图，如图 15.8 所示。

图 15.8　显示定位后的全景地图

关键技术

关键技术

本实例主要使用 BMapManager 类中的 init() 方法、BDLocation 类、PanoramaView 类中的 setPanorama() 方法，显示百度全景地图。

注意

百度地图开发平台要求在创建一个新的项目（包名与项目名称有变化）时重新申请新的密钥。

实现过程

（1）打开 Android Studio，新建一个项目，将其命名为“BaiDuPanoramicMap”，公司的域名为“mingrisoft. com”。

（2）打开百度开放平台网站（http://lbsyun.baidu.com/）并注册开放平台账号，然后申请密钥，再

下载 Android 地图和 Android 全景 SDK，最后配置环境及发布 Android 地图 SDK。具体操作方法请在百度开发平台的官方网站的“开发”菜单项中查询。

（3）导入资源文件：复制“光盘\Code\15\141\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（4）参考实例 138“百度地图”中设置所使用的权限，并在“application”标签中添加开发密钥与百度定位服务。

（5）在“BaiDuPanoramicMap\app\src\main\java\com\mingrisoft\baidupanoramicmap”目录中创建 PanoDemoApplication 类，该类是全景地图引擎的管理类。然后创建 initEngineManager() 方法实现地图引擎管理的初始化。代码如下：

```
01  public void initEngineManager(Context context) {
02      if (mBMapManager == null) {
03          //创建地图引擎管理对象
04          mBMapManager = new BMapManager(context);
05      }
06      //初始化地图引擎管理
07      if (!mBMapManager.init(new MyGeneralListener())) {
08          //显示错误消息
09          Toast.makeText(PanoDemoApplication.getInstance(). getApplicationContext(),
10                  "BMapManager  初始化错误!", Toast.LENGTH_LONG).show();
11      }
12      Log.d("ljx", "initEngineManager");
13  }
```

（6）打开默认生成的 activity_main.xml 布局文件，在该文件中添加显示全景地图的控件。代码如下：

```
01  <com.baidu.lbsapi.panoramaview.PanoramaView
02      android:id="@+id/panorama"
03      android:layout_width="fill_parent"
04      android:layout_height="fill_parent"
05      android:clickable="true" />
```

（7）打开 MainActivity 类，在该类中通过百度的定位监听器获取定位坐标并根据定位后的坐标显示全景地图。代码如下：

```
01  public class MyLocationListener implements BDLocationListener {
02      @Override
03      public void onReceiveLocation(BDLocation location) {
04          //MapView销毁后不再处理新接收的位置
05          if (location == null )
06              return;
07          if (isFirstLoc) {
08              //显示定位坐标的全景
09              mPanoView.setPanorama(location.getLongitude(), location.getLatitude());
10              isFirstLoc = false;          //设置isFirstLoc的属性为false，防止无法移动地图
11          }
12      }
13  }
```

扩展学习

本实例已经实现了百度地图根据定位后的坐标显示全景地图的功能，尝试实现在百度地图定位中添加全景地图的显示功能。

实例 142　下载百度离线地图

应用场景：离线地图

源码位置：光盘\Code\15\142

实例说明

使用离线地图，可满足在无网络环境下查看地图信息的需求。本实例将通过百度地图官网提供的 SDK 实现离线地图的功能。运行实例，关闭网络后无地图显示，如图 15.9 所示；单击“离线地图”按钮，开启网络，输入需要下载离线地图的省或市，然后单击“搜索”按钮将显示需要下载的城市 ID，如图 15.10 所示；单击“下载”按钮，将显示离线地图的下载进度，如图 15.11 所示；当下载进度为 100% 时，关闭网络重新进入该应用将显示离线地图。

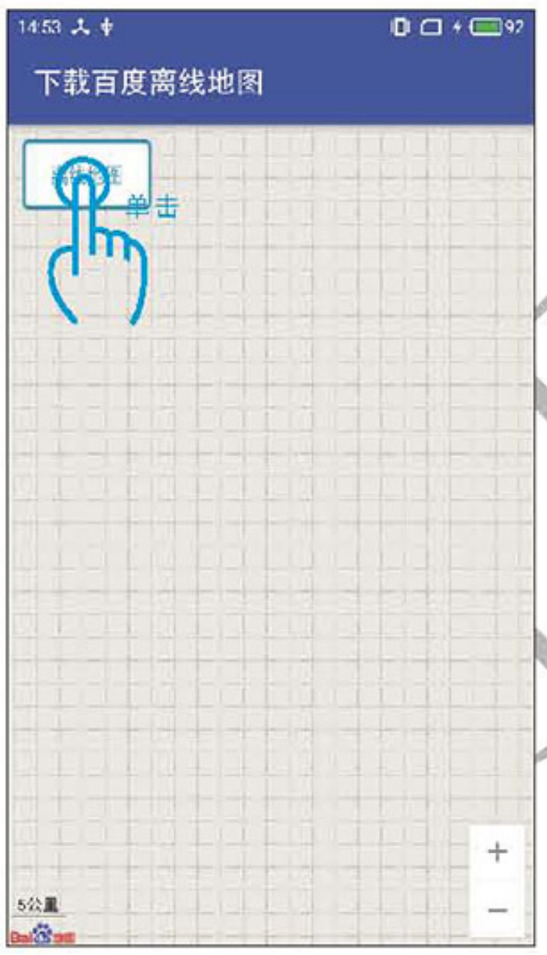


图 15.9　关闭网络无地图显示

图 15.10　搜索城市 ID

图 15.11　显示下载进度

关键技术

关键技术

本实例主要使用 MKOfflineMap 类中的 searchCity() 方法、start() 方法、getAllUpdateInfo() 方法，实现下载百度离线地图的效果。

注意

百度地图开发平台要求在创建一个新的项目（包名与项目名称有变化）时重新申请新的密钥。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“BaiDuOfflineMap”，公司域名为“mingrisoft. com”。

（2）打开百度开放平台网站（http://lbsyun.baidu.com/）并注册开放平台账号，然后申请密钥，再下载 Android 地图 SDK，最后配置环境及发布 Android 地图 SDK。具体操作方法请在百度开发平台的官方网站的“开发”菜单项中查询。

（3）导入资源文件：复制“光盘\Code\15\142\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（4）打开“BaiDuOfflineMap\app\src\main”目录中的 AndroidManifest.xml 文件，在该文件中首先填写百度地图定位所使用到的权限。然后在 application 标签中添加开发密钥与百度定位服务。

（5）在“BaiDuOfflineMap\app\src\main\java\com\mingrisoft\offlinemap”目录中创建 OfflineDemo 类，该类继承自 Activity 并实现 MKOfflineMapListener 接口。创建 search() 方法，该方法为“搜索”按钮的单击事件，用于搜索离线城市并获取该城市的 ID。代码如下：

```
01  public void search(View view) {
02      ArrayList<MKOLSearchRecord> records = mOffline.searchCity(cityNameView
03              .getText().toString());                          //获取输入城市的名称
04      if (records == null || records.size() != 1) {
05          return;                                              //返回离线城市信息
06      }
07      cidView.setText(String.valueOf(records.get(0).cityID));//显示离线城市ID
08  }
```

创建 start() 方法，该方法为“开始下载”或“暂停下载”按钮的单击事件，用于实现地图的下载与暂停功能。代码如下：

```
01  public void start(View view) {
02      if (cidView.getText().toString().equals("")) {           //搜索地区是否为空
03          Toast.makeText(this, "请先搜索需要下载的省或市的名称！", Toast.LENGTH_SHORT).show();
04      } else {
05          if (start == true) {                                 //按钮状态为“开始下载”
06              start = false;                                   //修改按钮状态
07              btn_start.setBackgroundResource(R.drawable.pause_btn);
08              //获取城市ID
09              int cityid = Integer.parseInt(cidView.getText().toString());
10              mOffline.start(cityid);                          //根据ID下载离线城市地图
11              Toast.makeText(this, "开始下载离线地图. cityid: " + cityid, Toast.LENGTH_SHORT)
12                      .show();
13              updateView();                                    //更新显示
14          } else {                                             //暂停下载
15              start = true;
16              btn_start.setBackgroundResource(R.drawable.start_btn);
17              int cityid = Integer.parseInt(cidView.getText().toString());
18              mOffline.pause(cityid);
```

```
19                Toast.makeText(this, "暂停下载离线地图. cityid: " + cityid, Toast.LENGTH_SHORT)
20                        .show();
21                updateView();
22            }
23        }
24    }
```

（6）打开 MainActivity 类，在该类中实现下载离线地图界面的跳转。（代码位置：BaiDuOfflineMap\ app\ src\main\java\com\mingrisoft\offlinemap\MainActivity.java）。

扩展学习

本实例中已经实现了下载百度的离线地图，请尝试在 onGetOfflineMapState() 方法中判断当下载进度完成时自动返回到地图界面，测试离线地图的显示需要关闭网络。

实例 143　百度地图路线规划

应用场景：路线规划

源码位置：光盘\Code\15\143

实例说明

本实例将通过百度地图官网提供的 SDK 来实现百度地图的路线规划功能。运行实例，开启无线网络，填写需要查询的地点，如图 15.12 所示；单击“驾车搜索”按钮，将显示路线对话框，如图 15.13 所示；在对话框中单击相应的路线，将在地图中显示路线的规划线路，如图 15.14 所示。

图 15.12　输入查询起点与终点

图 15.13　显示路线对话框

图 15.14　显示规划路线

关键技术

本实例主要使用 OnGetRoutePlanResultListener 接口中的 onGetDrivingRouteResult() 方法，RoutePlanSearch 类中的 drivingSearch() 方法、DrivingRouteLine 类实现百度地图的路线规划。

百度地图开发平台要求在创建一个新的项目（包名与项目名称有变化）时重新申请新的密钥。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“BaiDuRoutePlan”，公司域名为“mingrisoft.com”。

（2）打开百度开放平台网站（http://lbsyun.baidu.com/）并注册开放平台账号，然后申请密钥，再下载 Android 地图 SDK，最后配置环境及发布 Android 地图 SDK。具体操作方法请在百度开发平台的官方网站的“开发”菜单项中查询。

（3）导入资源文件：复制“光盘\Code\15\143\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（4）参考实例 138“百度地图”中所使用的权限。

（5）在“BaiDuRoutePlan\app\src\main\java”目录中创建 overlay 包，用于保存本项目中所使各种路线覆盖物的类文件，该包中包括 5 个类文件，其类名及作用如表 15.1 所示。

表 15.1　overlay 包中的 5 个类

类　名	描　述
BikingRouteOverlay	用于显示骑行路线的覆盖物
DrivingRouteOverlay	用于显示一条驾车路线的覆盖物
TransitRouteOverlay	用于显示换乘路线的覆盖物
WalkingRouteOverlay	用于显示步行路线的覆盖物
OverlayManager	提供一个能够显示和管理多个覆盖物的基类

（6）在“BaiDuRoutePlan\app\src\main\java\com\mingrisoft\baidurouteplan”目录中创建 RouteLineAdapter 类，该类是显示路线规划列表的适配器。

（7）在 RouteLineAdapter 类的 getView() 方法中获取行车路线与交通状况。代码如下：

```
01  //获取路线图数据
02  DrivingRouteLine drivingRouteLine = (DrivingRouteLine) routeLines.get(position);
03  holder.name.setText( "线路" + (position + 1));          //显示线路名称
04  //显示红绿灯数量
05  holder.lightNum.setText( "红绿灯数：" + drivingRouteLine.getLightNum());
06  //显示拥堵距离的米数
07  holder.dis.setText("拥堵距离为：" + drivingRouteLine.getCongestionDistance() + "米");
```

（8）在 MainActivity 类的 onGetDrivingRouteResult() 方法中通过返回的路况信息，弹出线路对话框并根据选择的线路绘制路线覆盖物。代码如下：

```
01  //判断返回路线数量是否大于一条
02  if (result.getRouteLines().size() > 1 ) {
03      nowResultd = result;
04      //提供路线选择的对话框
05      MyTransitDlg myTransitDlg = new MyTransitDlg(RoutePlanDemo.this,
06              result.getRouteLines(),
07              RouteLineAdapter.Type.DRIVING_ROUTE);
08      //对话框列表的单击事件
09      myTransitDlg.setOnItemInDlgClickLinster(new OnItemInDlgClickListener() {
10          public void onItemClick(int position) {
11              //获取浏览节点与路线数据
12              route = nowResultd.getRouteLines().get(position);
13              //显示驾车路线覆盖物
14              DrivingRouteOverlay overlay = new MyDrivingRouteOverlay(mBaidumap);
15              //设置覆盖物单击事件监听
16              mBaidumap.setOnMarkerClickListener(overlay);
17              routeOverlay = overlay;                    //路线覆盖物信息
18              //设置路线数据
19              overlay.setData(nowResultd.getRouteLines().get(position));
20              overlay.addToMap();                        //将覆盖物添加到地图上
21              overlay.zoomToSpan();                      //缩放地图与覆盖物显示在合适的视野内
22          }
23      });
24      myTransitDlg.show();                               //显示路线对话框
25  }
```

扩展学习

本实例实现了百度地图路线规划的功能，尝试在 activity_main.xml 布局中添加一个按钮，用于更换起点与终点的自定义图标，并在 MainActivity 类中创建自定义图标的标记，创建 changeRouteIcon() 方法作为按钮的单击事件。最后分别在 MyDrivingRouteOverlay、MyWalkingRouteOverlay、MyTransitRouteOverlay、MyBikingRouteOverlay 类中添加更换自定义图标的代码，实现更换起点与终点的自定义图标。

实例 144　高德 3D 地图

应用场景：3D 地图

源码位置：光盘\Code\15\144

实例说明

在 Android 手机的地图中，3D 地图的显示效果更加真实，使用者可以通过更加真实的场景准确判断地理位置。本实例将通过高德地图官网提供的 SDK 实现高德 3D 地图与卫星地图的模式切换。运行

实例，自动开启 3D 俯视地图，如图 15.15 所示；手指分开触摸屏幕向上滑动，将显示如图 15.16 所示切换平面视图；单击右上角的卫星地图按钮，将显示如图 15.17 所示的卫星地图效果。

图 15.15　开启 3D 俯视地图

图 15.16　切换到平面视图

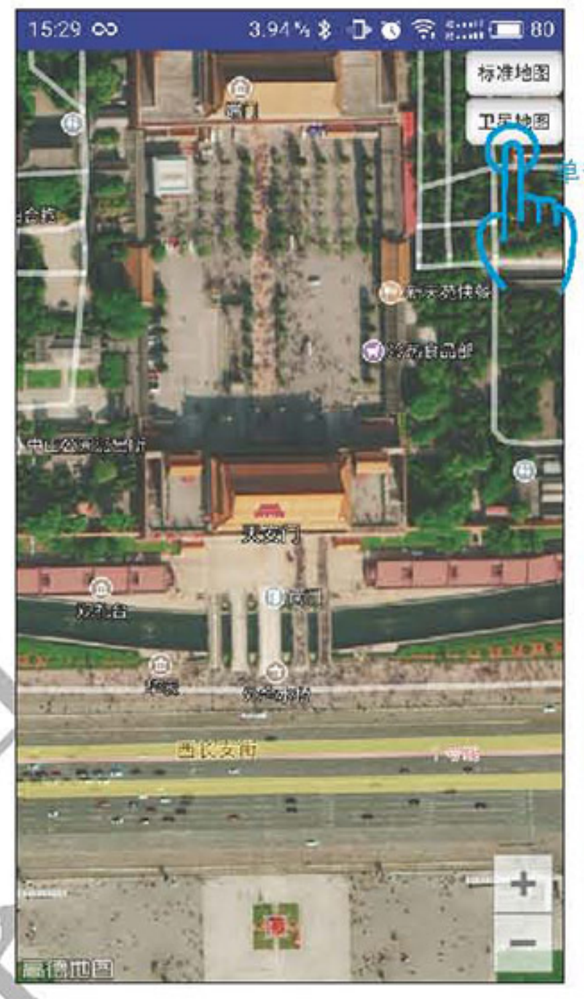


图 15.17　显示卫星地图

关键技术

关键技术

本实例主要使用 AMap 类中的 moveCamera() 方法、CameraPosition 类中的 CameraPosition() 构造方法、AMap 类中的 setMapType() 方法，实现 3D 地图与卫星地图的切换。

> **注意**　高德地图开发平台要求在创建一个新的项目（包名与项目名称有变化）时重新申请新的密钥。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“Gould3DMap”，公司域名为“mingrisoft. com”。

（2）打开高德开放平台网站（http://lbs.amap.com/）并注册开放平台账号，然后申请密钥，再下载 Android 地图 SDK，最后配置环境及发布 Android 地图 SDK。具体操作方法请根据高德开发平台的官网网站的“开发与支持”菜单项中查询。

（3）导入资源文件：复制“光盘\Code\15\144\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（4）打开“Gould3DMap\app\src\main”目录中的 AndroidManifest.xml 文件，在该文件中首先填写高德地图定位所使用到的权限。代码如下：

```
01  <!--允许程序打开网络套接字-->
02  <uses-permission android:name="android.permission.INTERNET" />
```

```
<!--允许程序设置内置SD卡的写权限-->
<uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE" />
<!--允许程序获取网络状态-->
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
<!--允许程序访问Wi-Fi网络信息-->
<uses-permission android:name="android.permission.ACCESS_WIFI_STATE" />
<!--允许程序读写手机状态和身份-->
<uses-permission android:name="android.permission.READ_PHONE_STATE" />
<!--允许程序访问CellID或Wi-Fi热点来获取粗略的位置-->
<uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION" />
```

然后在 <application> 标签中添加开发密钥。代码如下：

```
<meta-data
    android:name="com.amap.api.v2.apikey"
    android:value="此处填写开发者申请的Key" />
```

（5）在“Gould3DMap\app\src\main\java”目录中创建 utils 包，在该包中创建 Constants 类，该类是高德地图的工具类。

（6）打开 MainActivity 类，让该类实现 View.OnClickListener 接口。创建 changeCamera() 方法，用于实现地图显示位置、缩放级别、倾斜角度。代码如下：

```
private void changeCamera(CameraUpdate update) {
    aMap.moveCamera(update);                                    //根据参数改变可视区域
}
```

创建 init() 方法，用于初始化控件并设置地图显示位置、缩放级别、倾斜角度的具体参数。代码如下：

```
private void init() {
    if (aMap == null) {
        aMap = mMapView.getMap();                               //地图控制
    }
    //调用changeCamera()方法，设置地图显示位置、角度、缩放级别
    changeCamera(
            CameraUpdateFactory.newCameraPosition(new CameraPosition(
                    Constants.BEIJING, 18, 60, 0)));
    basicmap = (Button)findViewById(R.id.basicmap);         //获取标准地图按钮控件
    basicmap.setOnClickListener(this);                      //设置单击事件监听
    rsmap = (Button)findViewById(R.id.rsmap);               //获取卫星地图按钮控件
    rsmap.setOnClickListener(this);                         //设置单击事件
}
```

在 onClick() 方法中实现卫星地图与标准地图的切换。代码如下：

```
public void onClick(View v) {
    switch (v.getId()) {
```

```
03          case R.id.basicmap:
04              aMap.setMapType(AMap.MAP_TYPE_NORMAL);      //标准地图模式
05              break;
06          case R.id.rsmap:
07              aMap.setMapType(AMap.MAP_TYPE_SATELLITE);   //卫星地图模式
08              break;
09      }
10  }
```

扩展学习

本实例已经实现了高德 3D 地图与卫星地图切换的功能，请尝试 activity_main.xml 布局文件中添加一个用于切换夜景地图的按钮，然后在 MainActivity 类的 onClick() 方法中添加切换夜景地图的代码，实现夜景地图的模式切换。

实例 145　高德地图旋转方向定位

应用场景：方向定位

源码位置：光盘\Code\15\145

实例说明

在 Android 手机地图中，可以通过定位图标显示自己当前所行走的方向。本实例将通过高德地图官网提供的 SDK 实现高德旋转方向定位功能。运行实例，将定位手机当前位置及定位图标指向当前手机顶部所对应的方向，如图 15.18 所示。

图 15.18　显示当前手机定位的方向

关键技术

本实例主要使用 AMap 类中的 addMarker() 方法、SensorEventHelpe 类中的 setCurrent Marker() 方法、AMapLocationListener 接口中的 onLocationChanged() 方法，实现显示当前手机定位的方向。

高德地图开发平台要求在创建一个新的项目（包名与项目名称有变化）时重新申请新的密钥。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“RotationLocation”，公司域名为“mingrisoft. com”。

（2）打开高德开放平台网站（http://lbs.amap.com/）并注册开放平台账号，然后申请密钥，再下载 Android 地图和定位功能 SDK，最后配置环境及发布 Android 地图 SDK。具体操作方法请在高德开发平台的官网网站的“开发与支持”菜单项中查询。

（3）导入资源文件：复制“光盘\Code\15\145\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（4）参考实例 144“高德 3D 地图”，在 AndroidManifest.xml 文件中声明该项目所使用到的权限，并在 <application> 标签中设置开发者所申请的 Key。

（5）在“RotationLocation\app\src\main\java”目录中创建 util 包，在该包中创建 SensorEventHelper 类，用于根据手机的方向传感器实现地图定位图标的旋转功能。

（6）打开 MainActivity 类，让该类继承自 Activity 并实现 LocationSource 与 AMapLocationListener 接口，通过 activate() 方法实现激活定位功能。代码如下：

```
01  public void activate(OnLocationChangedListener listener) {
02      if (mlocationClient == null) {
03          mlocationClient = new AMapLocationClient(this);
04          mLocationOption = new AMapLocationClientOption();
05          mlocationClient.setLocationListener(this);                    //设置定位监听
06          //设置为高精度定位模式
07          mLocationOption.setLocationMode(AMapLocationClientOption.
08                  AMapLocationMode.Hight_Accuracy);
09          mlocationClient.setLocationOption(mLocationOption);           //设置定位参数
10          mlocationClient.startLocation();                              //启动定位并发送定位请求
11      }
12  }
```

创建 addMarker() 方法，用于实现添加定位图标的功能。代码如下：

```
01    private void addMarker(LatLng latlng) {
02        if (mLocMarker != null) {
03            return;
04        }
05        //设置定位图标
```

```
        BitmapDescriptor des = BitmapDescriptorFactory.
                fromResource(R.drawable.navi_map_gps_locked);
        MarkerOptions options = new MarkerOptions();                    //覆盖物选项
        options.icon(des);                                              //设置覆盖物定位图标
        options.anchor(0.5f, 0.5f);                                     //制定图标锚点
        options.position(latlng);                                       //设置覆盖物经纬度
        mLocMarker = aMap.addMarker(options);                           //添加地图覆盖物
    }
```

创建 onLocationChanged() 方法，实现通过定位信息完成显示定位图标与旋转的功能。代码如下：

```
public void onLocationChanged(AMapLocation amapLocation) {
        if (amapLocation != null
                && amapLocation.getErrorCode() == 0) {
            mLocationErrText.setVisibility(View.GONE);                  //定位成功隐藏显示错误的控件
            //保存经纬度
            LatLng location = new LatLng(amapLocation.getLatitude(),
                    amapLocation.getLongitude());
            if (!mFirstFix) {
                addCircle(location, amapLocation.getAccuracy());        //添加定位精度圆
                addMarker(location);                                    //添加定位图标
                mSensorHelper.setCurrentMarker(mLocMarker);             //定位图标旋转
                mFirstFix = true;
            } else {
                mCircle.setCenter(location);                            //设置圆心的经纬度坐标
                mCircle.setRadius(amapLocation.getAccuracy());          //设置圆形半径
                mLocMarker.setPosition(location);                       //设置图标经纬度位置
            }
            //更新可视区域位置
            aMap.moveCamera(CameraUpdateFactory.newLatLngZoom(location, 18));
        } else {
            //显示定位失败的原因
            String errText = "定位失败," + amapLocation.getErrorCode()
                    + ": " + amapLocation.getErrorInfo();
            Log.e("AmapErr", errText);
            mLocationErrText.setVisibility(View.VISIBLE);
            mLocationErrText.setText(errText);
        }
}
```

扩展学习

本实例已经通过高德地图与定位功能的 SDK，实现了定位并显示当前手机顶部所对应的方向，尝试在 onLocationChanged() 方法中通过 Toast 消息提示的方式，显示当前定位的具体位置与信息。

实例 146　高德地图驾车导航

应用场景：地图导航

源码位置：光盘\Code\15\146

实例说明

在开车的过程中如果找不到自己想要去的目的地时，导航就起到了至关重要的作用。本实例将通过高德地图官网提供的 SDK 来实现高德驾车导航的功能。运行实例，会自动定位当前手机所在的具体位置，如图 15.19 所示；单击“起点”按钮后在地图中设置起点位置，按照此方式再设置终点位置，单击“开始算路”按钮进行路线的规划，如图 15.20 所示；选好路线以后单击“开始导航”按钮，将显示如图 15.21 所示的导航界面。

图 15.19　显示当前手机位置

图 15.20　路线规划

图 15.21　开始导航

关键技术

关键技术

本实例主要使用 AMap.OnMapClickListener 接口中的 onMapClick() 方法、AMapNaviListener 接口中的 onCalculateRouteSuccess() 方法、AMapNavi 类中的 startNavi() 方法，实现驾车路线导航的功能。

注意

高德地图开发平台要求在创建一个新的项目（包名与项目名称有变化）时重新申请新的密钥。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“GouldNavigation”，公司域名为“mingrisoft.com”。

（2）打开高德开放平台网站（http://lbs.amap.com/）并注册开放平台账号，然后申请密钥，再下载 Android 地图和导航功能 SDK，最后配置环境及发布 Android 地图 SDK。具体操作方法请在高德开发平台的官网网站的“开发与支持”菜单项中查询。

（3）导入资源文件：复制“光盘\Code\15\147\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（4）参考实例 144“高德 3D 地图”，在 AndroidManifest.xml 文件中声明该项目所使用到的权限与开发者所申请的 Key，并添加新的权限。代码如下：

```
<!--精细定位-->
<uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" />
<!--改变Wi-Fi状态-->
<uses-permission android:name="android.permission.CHANGE_WIFI_STATE" />
<!--锁屏权限-->
<uses-permission android:name="android.permission.WAKE_LOCK" />
```

（5）在“GouldNavigation\app\src\main\java”目录中创建 util 包，在该包中创建 TTSController 类，该类用于注册语音播报的控件。

（6）在“GouldNavigation\app\src\main\java\com\mingrisoft\gouldnavigation”目录中创建 RouteNaviActivity 类，用于显示导航和语音播报的界面。

（7）在 MainActivity 类中，创建 onMapClick() 方法实现起点图标的设置。代码如下：

```
mAmap.setOnMapClickListener(new AMap.OnMapClickListener() {
    @Override
    public void onMapClick(LatLng latLng) {
        //控制选择起点
        if (mapClickStartReady) {
            //设置选中起点的坐标
            startLatlng = new NaviLatLng(latLng.latitude, latLng.longitude);
            mStartMarker.setPosition(latLng);           //在起点坐标位置绘制起点图标
            startList.clear();                          //清除起点坐标
            startList.add(startLatlng);                 //保存起点坐标
            mapClickStartReady = false;                 //设置起点标志状态
        }
        //控制选择终点
        if (mapClickEndReady) {
            //设置选中终点的坐标
            endLatlng = new NaviLatLng(latLng.latitude, latLng.longitude);
            mEndMarker.setPosition(latLng);             //在终点坐标位置绘制终点图标
            endList.clear();                            //清除终点坐标
            endList.add(endLatlng);                     //保存终点坐标
            mapClickEndReady = false;                   //设置终点标志状态
```

```
        }
    }
});
```

创建 onCalculateRouteSuccess() 方法，实现单路线计算与绘制。代码如下：

```
public void onCalculateRouteSuccess() {
    /**
     * 清空上次计算的路径列表
     */
    routeOverlays.clear();
    //获取当前计算出的路线，步行和驾车共用这一个对象
    AMapNaviPath path = mAMapNavi.getNaviPath();
    /**
     * 单路径不需要进行路径选择，直接传入-1即可
     */
    drawRoutes(-1, path);
}
```

（8）通过 RouteNaviActivity 类的 onCreate() 方法实现语音导航的功能。代码如下：

```
//初始化语音播报组件
mTtsManager = TTSController.getInstance(getApplicationContext());
mTtsManager.init();

mAMapNaviView = (AMapNaviView) findViewById(R.id.navi_view);    //获取导航控件
mAMapNaviView.onCreate(savedInstanceState);                     //导航与界面同步
//设置导航界面按钮的回调监听
mAMapNaviView.setAMapNaviViewListener(this);

mAMapNavi = AMapNavi.getInstance(getApplicationContext());      //创建导航对象
mAMapNavi.addAMapNaviListener(this);                            //添加导航事件回调监听
mAMapNavi.addAMapNaviListener(mTtsManager);                     //注册导航语音
mAMapNavi.setEmulatorNaviSpeed(60);                             //设置模拟导航的速度
boolean gps=getIntent().getBooleanExtra("gps", false);
if(gps){
    mAMapNavi.startNavi(AMapNavi.GPSNaviMode);                  //GPS导航
}
```

扩展学习

本实例已经实现了通过高德地图导航的 SDK 实现了高德地图驾车导航的功能，请尝试在 RouteNaviActivity 类中的 startNavi() 方法，实现模拟导航的功能。

明日科技

实例 147　高德地图实时显示移动轨迹

应用场景：地图移动轨迹

源码位置：光盘\Code\15\147

实例说明

通常在地图中可以根据移动轨迹的线段图，来查看自己的行走路线。本实例将通过高德地图官网提供的 SDK 来实现实时显示移动轨迹的功能。运行实例，会自动定位当前手机所在的具体位置，如图 15.22 所示；根据移动位置的变化，绘制移动的路线，如图 15.23 所示。

图 15.22　手机位置自动定位

图 15.23　手机移动的路线

关键技术

本实例主要使用 LatLng 类中的 LatLng() 构造方法、PolylineOptions 类中的 add() 方法和 AMap 类中的 addPolyline() 方法，实现移动轨迹的绘制功能。

注意　高德地图开发平台要求在创建一个新的项目（包名与项目名称有变化）时重新申请新的密钥。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“GouldTrajectory”，公司域名为“mingrisoft. com”。

（2）打开高德开放平台网站（http://lbs.amap.com/）并注册开放平台账号，然后申请密钥，再下载 Android 地图和定位功能 SDK，最后配置环境及发布 Android 地图 SDK。具体操作方法请在高德开发平台的官方网站的“开发与支持”菜单项中查询。

（3）导入资源文件：复制“光盘\Code\15\146\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（4）参考实例 144“高德 3D 地图”，在 AndroidManifest.xml 文件中声明该项目所使用到的权限与开发者所申请的 Key。

（5）在 MainActivity 类中，创建 onLocationChanged() 方法，实现获取定位后所返回的定位信息。代码如下：

```
01  public void onLocationChanged(AMapLocation amapLocation) {
02      if (mListener != null && amapLocation != null) {
03          if (amapLocation != null
04                  && amapLocation.getErrorCode() == 0) {
05              mListener.onLocationChanged(amapLocation);    //显示系统默认定位图标
06              //保存经纬度
07              LatLng newLocation = new LatLng(amapLocation.getLatitude(),
08                      amapLocation.getLongitude());
09              if (mFirstFix) {
10                  //记录第一次的定位信息
11                  oldLocation = newLocation;
12                  mFirstFix = false;
13              }
14              //位置有变化时
15              if (oldLocation != newLocation) {
16                  //设置新的位置信息
17                  setUpMap(oldLocation, newLocation);
18                  //记录新的位置信息
19                  oldLocation = newLocation;
20              }
21          } else {
22              String errText = "定位失败," + amapLocation.getErrorCode()
23                      + ": " + amapLocation.getErrorInfo();
24              Log.e("AmapErr", errText);
25              if (mFirstFix) {
26                  Toast.makeText(this, errText, Toast.LENGTH_SHORT).show();
27              }
28          }
29      }
30  }
```

（6）编写 setUpMap() 方法，根据坐标信息的变化实现在地图中绘制移动的线段。代码如下：

```
01  private void setUpMap(LatLng oldData, LatLng newData) {
02      options = new PolylineOptions();                    //创建线段选项
03      options.add(oldData, newData);                      //设置位置
04      options.width(20);                                  //设置宽度
```

明日科技

```
05        //加入对应的颜色，colorValues表示使用多颜色，color表示使用单色线
06        options.colorValues(colorList);
07        options.useGradient(true);                    //使用渐变线
08        aMap.addPolyline(options);                    //地图中添加线段
09    }
```

扩展学习

本实例已经实现了通过高德地图与定位的 SDK 实现了实时显示移动轨迹的功能，请尝试通过在“GouldTrajectory\app\src\main\java”目录中创建 offlinemap 包，该包中包含 6 个类用于实现高德离线地图的功能，然后在 MainActivity 类中创建 onOfflineMap() 方法，用于实现离线地图界面的跳转，这样可以在没有网络的情况下显示移动轨迹的路线。

实例 148　高德地图实时天气

应用场景：实时天气

源码位置：光盘\Code\15\148

实例说明

通过 Android 手机获取实时天气或预报天气的方式有很多种。本实例将通过高德地图官网提供的 SDK 实现高德实时天气的功能。运行实例，将自动获取默认城市的天气状况，如图 15.24 所示。

图 15.24　显示默认城市的天气状况

关键技术

本实例主要使用 WeatherSearchQuery 类中的 WeatherSearchQuery() 构造方法、WeatherSearch 类中的 setQuery() 方法、WeatherSearch.OnWeatherSearchListener 接口中的 onWeatherLiveSearched() 方法，实现获取高德天气状况的信息。

高德地图开发平台要求在创建一个新的项目（包名与项目名称有变化）时重新申请新的密钥。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“GouldWeather”，公司域名为“mingrisoft. com”。

（2）打开高德开放平台网站（http://lbs.amap.com/）并注册开放平台账号，然后申请密钥，再下载 Android 地图和搜索功能 SDK，最后配置环境及发布 Android 地图 SDK。具体操作方法请在高德开发平台的官方网站的“开发与支持”菜单项中查询。

（3）导入资源文件：复制“光盘\Code\15\148\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（4）参考实例 144“高德 3D 地图”，在 AndroidManifest.xml 文件中声明该项目所使用到的权限并在 application 标签中加入设置开发者所申请的 Key。

（5）在“GouldWeather\app\src\main\java”目录中创建 util 包，在该包中创建 ToastUtil 类。该类是 Toast 提示工具类。

（6）打开 MainActivity 类，让该类继承自 Activity 类，并实现 WeatherSearch.OnWeatherSearch Listener 接口。在该类中定义了显示天气状况的相关控件并设置默认搜索的城市名称，创建了 searchliveweather() 方法，用于实现搜索默认城市实时天气状况的功能。代码如下：

```
01  private void searchliveweather() {
02      //检索参数为城市和天气类型，实时天气为1
03      mquery = new WeatherSearchQuery(cityname, WeatherSearchQuery.WEATHER_TYPE_LIVE);
04      mweathersearch=new WeatherSearch(this);                          //创建天气查询对象
05      mweathersearch.setOnWeatherSearchListener(this);                 //设置天气信息查询监听
06      mweathersearch.setQuery(mquery);                                 //设置天气查询条件
07      mweathersearch.searchWeatherAsyn();                              //异步搜索
08  }
```

创建 onWeatherLiveSearched() 方法，实现获取实时天气状况返回信息的功能。代码如下：

```
01  public void onWeatherLiveSearched(LocalWeatherLiveResult weatherLiveResult , int rCode) {
02      if (rCode == AMapException.CODE_AMAP_SUCCESS) {
03          if (weatherLiveResult != null && weatherLiveResult.getLiveResult() != null) {
04              weatherlive = weatherLiveResult.getLiveResult();             //获取实时天气结果
05              reporttime1.setText(weatherlive.getReportTime()+"发布");     //显示发布时间
06              weather.setText(weatherlive.getWeather());                   //显示天气状况
07              Temperature.setText(weatherlive.getTemperature()+"°");       //显示实时温度
08              //显示风力
```

```
09                wind.setText(weatherlive.getWindDirection()+"风:"+
10                        weatherlive.getWindPower()+"级");
11                //显示湿度
12                humidity.setText("湿度:"+weatherlive.getHumidity()+"%");
13            }else {
14                //提示没有得到相关数据
15                ToastUtil.show(MainActivity.this, R.string.no_result);
16            }
17        }else {
18            //提示错误代码
19            ToastUtil.showerror(MainActivity.this, rCode);
20        }
21    }
```

（7）在 onCreate() 方法中调用 searchliveweather() 方法，实现开启应用后自动获取默认城市的天气状况的功能。

扩展学习

本实例已经实现了通过高德搜索功能的 SDK 获取固定城市天气状况的功能，这里在 activity_main.xml 布局文件中添加用于搜索城市的编辑框控件与按钮，然后在MainActivity 类中创建 onSearch() 方法，实现搜索任意城市的天气状况的功能。

明日科技

第16章 物联网开发

实例149 智能摄像头

应用场景：智能家居智能监控

源码位置：光盘\Code\16\149

实例说明

在现今互联网纵横的时代，手机可以掌控很多硬件设施。为实时监控家中情况，可以在家中安装摄像头并用手机来操控摄像头，包括控制摄像头上下左右移动，进行拍照和录像等。本实例将实现通过手机控制摄像头拍照和录像等功能。运行实例，将显示如图16.1所示连接摄像头的默认界面；单击上下翻转按钮，设置画面翻转，如图16.2所示；单击“开始录像”按钮，监控家中情况，如图16.3所示。

图16.1 默认连接摄像头界面

图16.2 设置摄像头画面翻转

图16.3 开始录制影像

关键技术

本实例主要使用“易视眼”摄像头及开发 SDK，View 类中的 setBackgroundResource() 方法和 Intent 类中的 putExtra() 方法。其中，在实现手机连接监控设备将序列号传递给服务器时，用到了 setBackgroundResource() 方法和 putExtra() 方法。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“YiShiYanCamera”，公司域名为“mingrisoft.com”。

（2）在“app\res\drawable”文件夹中导入相关的图片资源。

（3）在“app\libs”文件夹中导入相关的 Jar 包。

（4）在“app\src\main\java”文件夹中创建“jniLibs”文件夹并导入相关的 .So 文件。

（5）创建 MainActivity 类，用于输入用户名、密码和 UID，并判断是否为空，关键代码如下：

```
if (nameEt.getText().toString().trim().equals(null)){
    Toast.makeText(MainActivity.this, "用户不能为空", Toast.LENGTH_SHORT).show();
}else if (UIDEt.getText().toString().trim().equals(null)){
    Toast.makeText(MainActivity.this, "UID错误", Toast.LENGTH_SHORT).show();
}else if (codeEt.getText().toString().tr
im().equals(null)){
    Toast.makeText(MainActivity.this, "密码错误", Toast.LENGTH_SHORT).show();
}
```

（6）创建 CameraActivity 类，用于控制监控摄像头，可以远程控制摄像头的移动，并实时获取摄像头的语音，以摄像头上调为例，关键代码如下：

```
case R.id.top_btn:     //上调
    //参数含义：方向，速度，距离，触摸点的个数，外接设备，通道号
    mCamera.sendIOCtrl(Camera.DEFAULT_AV_CHANNEL, AVIOCTRLDEFs.IOTYPE_USER_IPCAM_PTZ_COMMAND,
            AVIOCTRLDEFs.SMsgAVIoctrlPtzCmd.parseContent((byte)
    AVIOCTRLDEFs.AVIOCTRL_PTZ_UP, (byte) 0, (byte) 0, (byte) 0, (byte) 0,
            (byte) Camera.DEFAULT_AV_CHANNEL));
    break;
```

（7）开启与关闭实时监听、监控摄像头的语音，关键代码如下：

```
case R.id.voice_btn:  //开启语音实时监听
    if (!mIsListening) {
        //打开开关
        mCamera.startListening(Camera.DEFAULT_AV_CHANNEL, true);  //开始监听
        mIsListening = true;
        voiceBtn.setBackgroundResource(R.drawable.voice);          //设置按钮的背景

    } else {
```

```
09          //关闭开关
10          mCamera.stopListening(Camera.DEFAULT_AV_CHANNEL);      //停止监听
11          mIsListening = false;
12          voiceBtn.setBackgroundResource(R.drawable.no_voice);  //设置按钮的背景
13      }
14      break;
```

扩展学习

本实例每次进入都需要进行登录，请尝试实现登录一次，以后免登录的功能。

实例 150　智能家居

应用场景：智能家居智能操作

源码位置：光盘\Code\16\150

实例说明

智能家居的时代即将来临，随着智能手机的普及，广大用户希望在手机上控制家中电器的使用。本实例介绍的智能家居，可通过手机控制家中的电器。运行实例，连接吸顶灯的具体工作流程如图 16.4 所示，连接氛围灯及电视机的工作流程与这一流程相同。

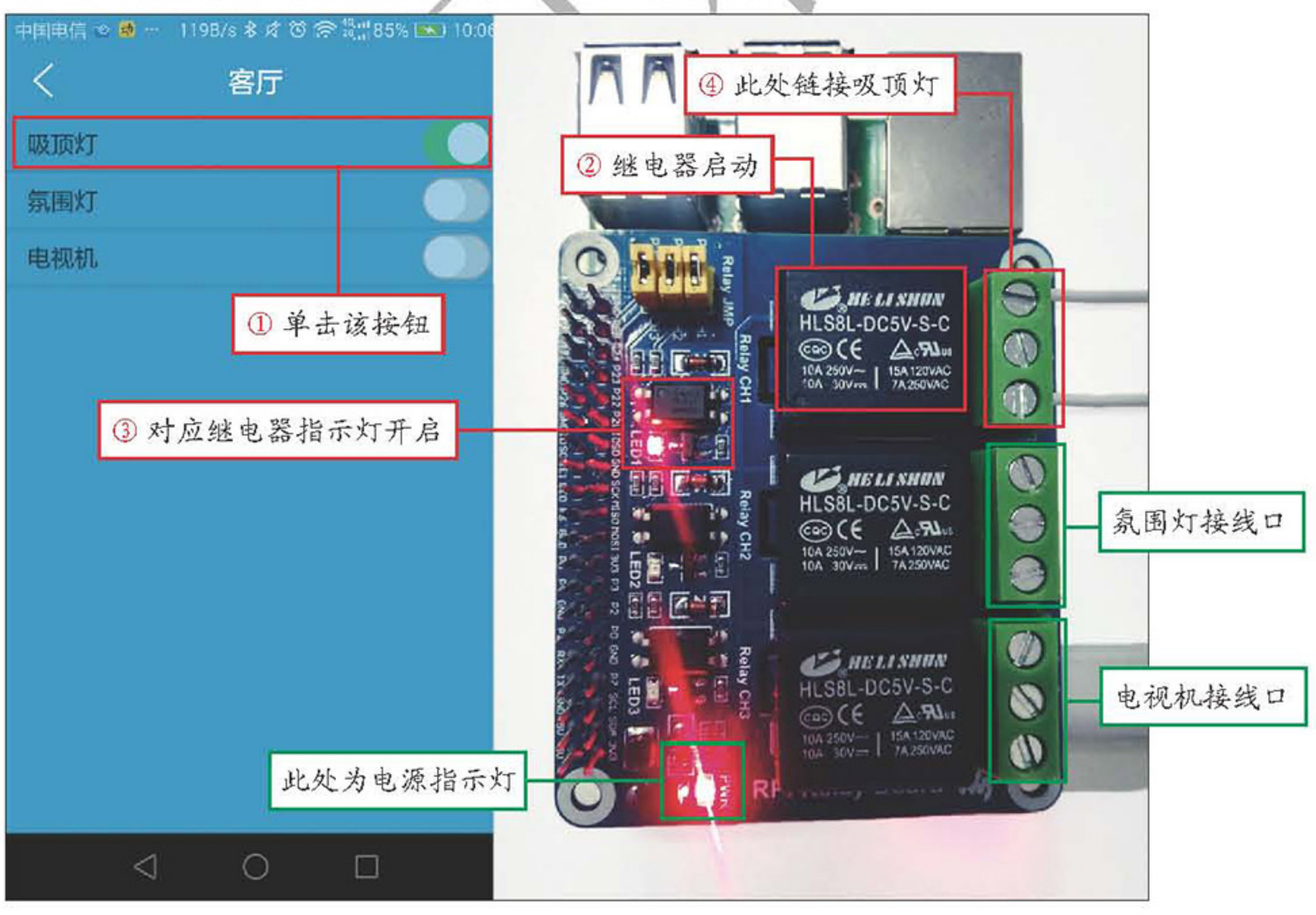


图 16.4　连接吸顶灯的工作流程

关键技术

本实例主要使用“树莓派”微电脑主板，220V 继电器，Resources 类中的 obtain TypedArry() 方法，Intent 类中的 putExtra() 方法。其中，在实现手机连接监控设备将序列号传递给服务器时，用到了 obtainTypedArry() 方法、putExtra() 方法。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“SmartHome”，公司域名为“mingrisoft.com”。
（2）在“app\res\mipmap”文件夹中导入相关的图片资源。
（3）在“app\res\values”文件夹中创建“array.xml”文件，用于存放房间的背景图片。
（4）在本实例中有如表 16.1 所示的关键类及其作用。

表 16.1　关键类

类　名	描　述	类　名	描　述
MainActivity	进入 App 主界面	DetailAdapter	详情界面的适配器
DetailActivity	进入房间的详情界面	SwitchView	自定义的开关按钮
MainListAdapter	主界面的适配器	QueueSingleton	单例模式

（5）在 MainListAdapter 类，获取背景图片，通过 for 循环提取，并加载到数组中，关键代码如下：

```
/**
 * 将图片提取出来
 **/
    TypedArray ar = context.getResources().obtainTypedArray(R.array.imgArray);
    int len = ar.length();
    resids = new int[len];
    //通过for循环获取数据
    for (int i = 0; i < len; i++) {
        resids[i] = ar.getResourceId(i, 0);
}
```

（6）在 DetailActivity 类，获取通过 MainActivity 传递的数据，并显示在 ListView 上，关键代码如下：

```
private String masterRoom[] = {"吸顶灯", "床头灯"};                    //主卧的开关
private String secondRoom[] = {"吸顶灯", "床头灯"};                    //次卧的开关
private String livingRoom[] = {"吸顶灯", "氛围灯","电视机"};           //客厅的开关
private String studyRoom[] = {"吸顶灯", "护眼灯"};                     //书房的开关
private String toilet[] = {"吸顶灯", "排气扇"};                        //卫生间的开关
private String kitchen[] = {"吸顶灯", "油烟机"};                       //厨房的开关
private int resids[];                                                  //用于加载图片
```

（7）在单击吸顶灯开关时，向服务器发送指定数据，打开继电器一的开关，并实时刷新按钮状态，开启与关闭的代码如下：

```
01  case 0:                                          //继电器一开关
02      if (oneBL) {
03          oneBL = false;
04          holder.switchView.setChecked(true);      //设置开关开启
05          holder.switchView.initAnimator();        //刷新动画
06          dataInternet("on1");                     //向服务器传递数据
07      } else {
08          oneBL = true;
09          holder.switchView.setChecked(false);     //设置开关关闭
10          holder.switchView.initAnimator();        //刷新动画
11          dataInternet("off1");                    //向服务器传递数据
12      }
13      break;
```

扩展学习

本实例所控制的单片机是使用 Python 语言编写的，请尝试编写 Python 语言的代码。

实例 151　手表消息通知

应用场景：Android Wear

源码位置：光盘\Code\16\151
视频位置：光盘\Video\16\151

实例说明

如今，智能手表越来越普及，各种样式的智能手表逐渐走入人们的生活。本实例将实现手机向智能手表发送消息功能。如图 16.5 所示，在该界面中有三个选项，选择其中一个选项后，会发送出一个通知，同时该通知会显示在 Android Wear 设备上。运行实例，Android Wear 上的通知效果如图 16.6 所示。

图 16.5　界面效果

图 16.6　通知效果

关键技术

本实例主要使用 NotificationCompat.Builder 创建通知，使用 NotificationManagerCompat 发送通知，从而实现手机发送消息到智能手表。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“Notification”，公司域名为“mingrisoft. com”。

（2）导入资源文件：复制“光盘\Code\16\151\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）在 MainActivity 类的 onCreate() 方法中执行初始化操作，包括初始化数据和初始控件的操作。代码如下：

```
01  @Override
02  protected void onCreate(Bundle savedInstanceState) {
03      super.onCreate(savedInstanceState);
04      setContentView(R.layout.activity_main);
05      intiView();                                                    //初始化控件
06      text = new ArrayList<>();                                      //数据集合
07      text.add("Android从入门到精通");                                //标题
08      text.add("Java编程词典");                                       //标题
09      text.add("Java从入门到精通");                                   //标题
10      text.add(getString(R.string.one));                             //简介
11      text.add(getString(R.string.two));                             //简介
12      text.add(getString(R.string.three));                           //简介
13  }
```

创建 sendNotification() 方法，在该方法中实现触发后发送消息通知。代码如下：

```
01  private void sendNotification(int index,int message) {
02      NotificationManagerCompat manager = NotificationManagerCompat.from(this);    //通知管理器
03      NotificationCompat.Builder builder = new NotificationCompat.Builder(this);   //通知构造器
04      builder.setContentTitle(text.get(index));                                     //设置标题
05      builder.setDefaults(Notification.DEFAULT_ALL);                                //设置通知属性
06      builder.setLargeIcon(                                                         //设置大图标
07              BitmapFactory.decodeResource(getResources(),resID[index]));
08      builder.setAutoCancel(true);                                                  //设置自关闭
09      builder.setSmallIcon(R.mipmap.ic_launcher);                                   //设置小图标
10      manager.notify(1,builder.build());                                            //发送通知
11  }
```

扩展学习

请尝试为消息通知添加大文本信息，并使其显示在 Android Wear 设备上。

明日科技

实例 152　手机与手表间的消息传递

应用场景：Android Wear

源码位置：光盘\Code\16\152
视频位置：光盘\Video\16\152

实例说明

视频讲解

本实例将实现手机与手表进行消息传递的功能。在手机中开启实时定位服务，手机界面效果如图 16.7 所示；手表上有 4 个按钮，用来选择查询哪些关键字，如图 16.8 所示；当单击其中一个按钮后；手表就会向手机发送一条数据，手机接收到消息后使用百度地图进行 POI 检索，当检索完成后检索的结果发送给手表，如图 16.9 所示；手表接收到检索结果，并将检索结果展示在手表中的界面上，效果如图 16.10 所示。

图 16.7　手机界面效果

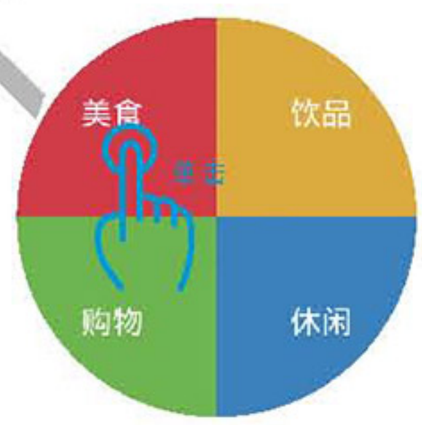


图 16.8　手表界面效果

图 16.9　加载数据等待

图 16.10　展示检索结果

关键技术

关键技术

本实例主要使用 WearableListenerService 类接收消息，使用 PutDataMapRequest 类发送消息，使用 GoogleApiClient 开启和关闭手机与手表的连接。

实现过程

（1）打开 Android Studio，新建一个 Android Wear 项目，命名为“Nearby”，公司域名为“mingrisoft.com”。

（2）导入资源文件，在这里需要注意 Moudle 的名称，要将文件放入对应的 Moudle 中。

（3）在名为 mobile 的 Moudle 中创建 ListenerService 类，让其继承 WearableListenerService 类，并重写 onDataChanged() 方法，在该方法中处理接收到的消息。代码如下：

代码位置：Nearby\mobile\src\main\java\com\mingrisoft\nearby\service\ListenerService.java

```
01  public class ListenerService extends WearableListenerService {
02      @Override
03      public void onDataChanged(DataEventBuffer dataEvents) {
04          super.onDataChanged(dataEvents);
05          final List<DataEvent> events = FreezableUtils.freezeIterable(dataEvents);
06          for(DataEvent event : events) {
07              final Uri uri = event.getDataItem().getUri();                    //获取消息的uri
08              final String path = uri!=null ? uri.getPath() : null;            //获取标识
09              if("/KEY".equals(path)) {                                        //判断标识
10                  final DataMap map = DataMapItem
11                          .fromDataItem(event.getDataItem()).getDataMap(); //获取DataMap对象
12                  String key = map.get("keyword");                             //获取消息内容
13                  Intent localIntent = new Intent("phone.localIntent");        //发送广播意图
14                  localIntent.putExtra("result", key);                         //添加附加内容
15                  //发送广播
16                  LocalBroadcastManager.getInstance(this).sendBroadcast(localIntent);
17              }
18          }
19      }
20  }
```

（4）在名为 mobile 的 Moudle 中的 MyActivityPhone 类的 onCreate() 方法中初始化界面。代码如下：

代码位置：Neraby\mobile\src\main\java\com\mingrisoft\nearby\activity\MyActivityPhone.java

```
01  @Override
02  protected void onCreate(Bundle savedInstanceState) {
03      super.onCreate(savedInstanceState);
04      SDKInitializer.initialize(getApplicationContext());
05      setContentView(R.layout.activity_my_phone);
06      //获取地图控件引用
07      mMapView = (MapView) findViewById(R.id.bmapView);
08      locationClient = new LocationClient(getApplicationContext());
09      initLocation();                                                          //初始化
10      locationClient.registerLocationListener(locationListener);
11      mBaiduMap = mMapView.getMap();
12      mMapView.showScaleControl(true);                                         //显示比例尺
```

明日科技

```
13      mMapView.showZoomControls(true);                                    //显示缩放按钮
14      //设置缩放，确保屏幕内容全部显示
15      mBaiduMap.setMapStatus(MapStatusUpdateFactory.zoomTo(18));
16      mGoogleApiClient = new GoogleApiClient.Builder(this)
17              .addConnectionCallbacks(this)                               //设置链接回调监听
18              .addApi(Wearable.API)
19              .build();
20      mGoogleApiClient.connect();                                         //创建连接
21      mResultReceiver = createBroadcastReceiver();
22      LocalBroadcastManager.getInstance(this).registerReceiver(
23              mResultReceiver, new IntentFilter("phone.localIntent"));
24      mPoiSearch = PoiSearch.newInstance();                               //获取检索对像
25      mPoiSearch.setOnGetPoiSearchResultListener(poiListener);            //设置检索监听
26  }
```

添加 POI() 方法，用来根据关键字执行检索功能，代码如下：

```
01  private void POI (String key){
02      mPoiSearch.searchNearby(new PoiNearbySearchOption()
03              .keyword(key)                                               //美食、饮品、购物、休闲
04              .location(mPoint)                                           //检索位置
05              .radius(1000)                                               //检索范围
06      );
07  }
```

添加 POI 检索的回调监听，并在 OnGetPoiSearchResultListener 接口的重写方法中，处理检索结果。代码如下：

```
01  OnGetPoiSearchResultListener poiListener = new OnGetPoiSearchResultListener(){
02      public void onGetPoiResult(PoiResult result){
03          //获取POI检索结果
04          Log.i("onGetPoiResult", "onGetPoiResult: ");
05          int resultCount = result.getTotalPoiNum();                      //检索总数
06          Log.i("检索总数", resultCount + " ");
07          List<PoiInfo> resultAllPoi = result.getAllPoi();                //检索数据总数
08          Log.w("检索总数", resultAllPoi.size() + " ");
09          Iterator<PoiInfo> iterator = resultAllPoi.iterator();           //迭代器
10          ArrayList<DataMap> dataMapList = new ArrayList<>();
11          DataMap dataMap;
12          while (iterator.hasNext()){                                     //遍历集合
13              PoiInfo poiInfo = iterator.next();
14              dataMap = new DataMap();
15              dataMap.putString("address" ,  poiInfo.address);            //地址
16              dataMap.putString("city" ,  poiInfo.city);                  //城市
17              dataMap.putString("name" ,  poiInfo.name);                  //名称
18              dataMap.putString("phoneNum" ,  poiInfo.phoneNum);          //电话
```

```
            dataMapList.add(dataMap);                                    //将数据添加进集合
        }
        sendDataToWear(dataMapList);                                     //发送数据
    }
    public void onGetPoiDetailResult(PoiDetailResult result){
        //获取Place详情页检索结果
        Log.i("PoiDetailResult", "onGetPoiResult: ");
    }

    @Override
    public void onGetPoiIndoorResult(PoiIndoorResult poiIndoorResult) {
        //室内检索结果
        Log.i("onGetPoiIndoorResult", "onGetPoiResult: ");
    }
};
```

添加 sendDataWear() 方法，用来将检索结果发送给手表。代码如下：

```
01  private void sendDataToWear(ArrayList<DataMap> dataMapList){
02      PutDataMapRequest dataMapRequest = PutDataMapRequest.create("/POI");
03      DataMap dataMap = dataMapRequest.getDataMap();                   //获取消息的载体
04      dataMap.putLong("time" , System.currentTimeMillis());            //保证Wear设备能接收到消息
05      dataMap.putDataMapArrayList("data",dataMapList);                 //检索数据集合
06      Wearable.DataApi                                                 //发送数据
07              .putDataItem(mGoogleApiClient,dataMapRequest.asPutDataRequest());
08  }
```

（5）在 wear 的 MyActivityWear 类中添加一些功能，用于接收从手机传递过来的信息。让 MyActivity Wear 类实现 DataApi.DataListener 接口，并重写 DataApi.DataListener 接口的 onDataChanged() 方法，在该方法中处理从手机发送过来的消息。代码如下：

```
01  @Override
02  public void onDataChanged(DataEventBuffer dataEventBuffer) {
03      Log.i("有数据", "onDataChanged: ");
04      final List<DataEvent> events = FreezableUtils.freezeIterable(dataEventBuffer);
05      for (DataEvent event : events) {
06          final Uri uri = event.getDataItem().getUri();                //获取uri
07          final String path = uri != null ? uri.getPath() : null;
08          if ("/POI".equals(path)) {                                   //判断标识
09              final DataMap map = DataMapItem                          //获取DataMap对象
10                      .fromDataItem(event.getDataItem()).getDataMap();
11              ArrayList<DataMap> list = map.getDataMapArrayList("data");//获取数据集合
12              ArrayList<Result> data = new ArrayList<>();              //创建集合该集合实现了序列化
13              DataMap dataMap;
```

```
            Result result;
            for (int i = 0,length = list.size();i < length;i++){
                dataMap = list.get(i);                                    //获取对应index的元素
                result = new Result();                                    //创建返回结果
                result.setAddress(dataMap.getString("address"));          //存入地址
                result.setCity(dataMap.getString("city"));                //存入城市
                result.setName(dataMap.getString("name"));                //存入名称
                result.setPhoneNum(dataMap.getString("phoneNum"));        //存入电话
                data.add(result);                                         //添加数据到集合中
            }
            Intent localIntent = new Intent("wearable.localIntent");      //发送广播意图
            localIntent.putParcelableArrayListExtra("data", data);        //发送集合
            //发送广播
            LocalBroadcastManager.getInstance(this).sendBroadcast(localIntent);
        }
    }
}
```

创建适配器，进行数据的绑定。将内容显示到列表中。这样一个手机与手表通讯的实例就大功告成了。

扩展学习

在该实例中，并未设置权限动态申请，请尝试为该实例增加动态申请定位的权限功能。

实例 153　手表动作通知

应用场景：Android Wear

源码位置：光盘\Code\16\153

视频位置：光盘\Video\16\153

实例说明

本实例主要实现智能手表中单击通知消息后，手机进入相关页面的功能。运行实例，在如图 16.11 所示的 Android Wear 通知中单击“自定义动作”按钮后，手机跳转到如图 16.12 所示的界面。

关键技术

本实例主要使用 NotificationCompat.Builder 类的 setContentIntent() 方法 NotificationCompat.WearableExtender 类的 addAction() 方法。

图 16.11　通知中的动作

图 16.12　界面效果

实现过程

（1）打开 Android Studio，新建一个项目，命名为“Action”，公司域名为“mingrisoft.com”。

（2）导入资源文件：复制“光盘\Code\16\153\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）在 MainActivity 类中的 onCreate() 方法中进行初始化操作。代码如下：

```
01  @Override
02  protected void onCreate(Bundle savedInstanceState) {
03      super.onCreate(savedInstanceState);
04      setContentView(R.layout.activity_main);
05      intiView();                                          //初始化控件
06      text = new ArrayList<>();                            //数据集合
07      text.add("Android从入门到精通");                      //标题
08      text.add("Java编程词典");                             //标题
09      text.add("Java从入门到精通");                         //标题
10      text.add(getString(R.string.one))                    //简介
11      text.add(getString(R.string.two));                   //简介
12      text.add(getString(R.string.three));                 //简介
13  }
```

创建 getActionIntent() 方法，用来打开 ResultActivity 的意图，并在意图中传入想要传递的数据。代码如下：

```
01  private Intent getActionIntent(int type,@DrawableRes int imageID, String title,
02          @StringRes int descID){
03      Intent intent = new Intent(this,ResultActivity.class);
```

```
    intent.putExtra(ActionType.TYPE,type)
            .putExtra("imageID",imageID)                    //图片资源ID
            .putExtra("name",title)                         //名称
            .putExtra("descID",descID);                     //简介资源ID
    return intent;
}
```

在 MainActivity 类中创建 sendNotification() 方法，用于创建和发送带有动作的通知。代码如下：

```
@RequiresApi(api = Build.VERSION_CODES.KITKAT_WATCH)
private void sendNotification(int index, int message) {
    //创建意图
    Intent defaultAction = getActionIntent(ActionType.DEFAULT,resID[index],
            text.get(index),stringID[index]);                //系统自带
    Intent customAction = getActionIntent(ActionType.CUSTOM,resID[index],
            text.get(index),stringID[index]);                //自定义
    Intent voiceAction = getActionIntent(ActionType.VOICE,resID[index],
            text.get(index),stringID[index]);                //语音
    PendingIntent defaultPending = PendingIntent.getActivity(this, 1, defaultAction,
            PendingIntent.FLAG_UPDATE_CURRENT);              //系统自带
    PendingIntent customPending = PendingIntent.getActivity(this, 2, customAction,
            PendingIntent.FLAG_UPDATE_CURRENT);              //自定义
    PendingIntent voicePending = PendingIntent.getActivity(this, 3, voiceAction,
            PendingIntent.FLAG_UPDATE_CURRENT);              //语音
    NotificationManagerCompat manager = NotificationManagerCompat.from(this);  //通知管理器
    NotificationCompat.Builder builder = new NotificationCompat.Builder(this); //通知构造器
    builder.setContentTitle(text.get(index));                //设置标题
    builder.setDefaults(Notification.DEFAULT_ALL);           //设置通知属性
    NotificationCompat.BigTextStyle style = new NotificationCompat.BigTextStyle();
    style.bigText(text.get(message));                        //设置内容
    builder.setStyle(style);                                 //设置大文本样式
    builder.setLargeIcon(                                    //设置大图标
            BitmapFactory.decodeResource(getResources(), resID[index]));
    builder.setAutoCancel(true);                             //设置自关闭
    builder.setContentIntent(defaultPending);                //在手机上打开
    builder.setSmallIcon(R.mipmap.ic_launcher);              //设置小图标
    builder.extend(new NotificationCompat.WearableExtender()
            .addAction(new NotificationCompat.Action.Builder
                    (R.mipmap.ic_launcher,"语音动作",voicePending)
                    .addRemoteInput(new RemoteInput.Builder("voice")
                            .setLabel("查看详情")
                            .build())
                    .build())
            .addAction(new NotificationCompat.Action.Builder
                    (R.mipmap.ic_launcher,"自定义动作",customPending)
```

```
37                      .build()));
38      manager.notify(1, builder.build());                    //发送通知
39  }
```

（4）在 ResultActivity 类中，将传递过来的数据展示在界面中。关键代码如下：

```
01  getSupportActionBar().setTitle(title);                     //设置标题
02  pic.setImageResource(intent.getIntExtra("imageID",0));     //图片
03  name.setText(intent.getStringExtra("name"));               //名称
04  desc.setText(getString(intent.getIntExtra("descID",0)));   //简介
```

扩展学习

请尝试将创建通知的设置大图片的配置代码删除，看看程序发出通知后的效果是什么样的。

实例 154　单对单蓝牙聊天

应用场景：蓝牙传输数据

源码位置：光盘\Code\16\154

实例说明

本实例将演示蓝牙相连的两台手机如何实现聊天功能。运行本实例，将 2 台 Android 手机通过蓝牙连接，效果如图 16.13 所示；单击连接的手机蓝牙名称选项，进入聊天页面，如图 16.14 所示；显示发送及接收到的消息，如图 16.15 所示。

图 16.13　连接测试机

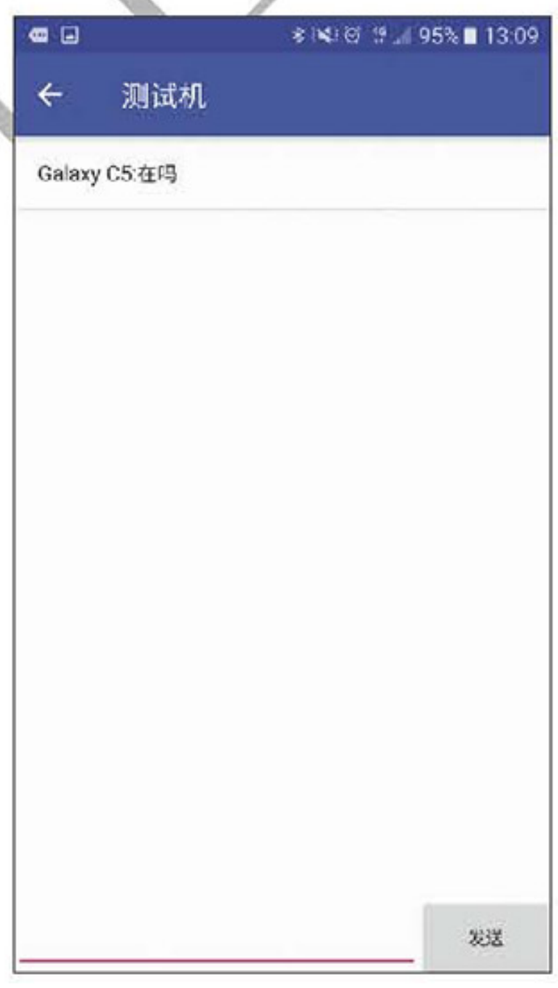


图 16.14　聊天页面

图 16.15　收到消息效果

关键技术

本实例主要使用 ListView 类中的 setAdapter() 方法、Adapter 类中的 onItemClick() 方法、Handler 类中的 sendMessage() 方法、ArrayAdapter 类中的 add() 方法、BluetoothDevice 类中的 createRfcommSockeToServiceRecord() 方法，实现蓝牙聊天功能。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“BluetoothDemo”，公司域名为“mingrisoft.com”。

（2）导入资源文件：复制“光盘\Code\16\154\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）在“app\java\com\mingrisoft\bluetoothdemo”文件夹下默认生成的 MainActivity 类中，创建 startBluetooth() 方法，用于创建广播服务，以及开启线程创建连接接收蓝牙信息。代码如下：

```
01  //开启蓝牙
02  private void startBluetooth() {
03      //创建蓝牙适配器
04      BluetoothAdapter adapter = BluetoothAdapter.getDefaultAdapter();
05      adapter.startDiscovery();
06      //创建广播接收蓝牙消息
07      receiver = new BluetoothReceiver(this.adapter);
08      IntentFilter filter = new IntentFilter(BluetoothDevice.ACTION_FOUND);
09      //注册蓝牙广播
10      registerReceiver(receiver, filter);
11      //开启建立连接线程类
12      new SocketThread(handler).start();
13  }
```

（4）在 MainActivity 类中实现 onItemClick() 方法，用来为蓝牙列表添加单击事件。代码如下：

```
01  //绑定单击事件
02  @Override
03  public void onItemClick(AdapterView<?> parent, View view, int position, long id) {
04      //获取蓝牙名称
05      BluetoothDevice device = (BluetoothDevice) parent.getAdapter().getItem(position);
06      //进入聊天页面
07      Intent intent = new Intent(this, ChatActivity.class);
08      intent.putExtra("device", device);
09      startActivity(intent);
10  }
```

（5）在“app\java\com\mingrisoft\bluetoothdemo”文件夹下创建 SocketThread 线程类，在其中创建蓝牙连接。代码如下：

```
01  //获取到一个Socket对象
02  BluetoothSocket socket = device.createRfcommSocketToServiceRecord(uuid);
```

```
03  //建立连接
04  socket.connect();
05  //用于接收发送消息
06  client = new ChatClient(socket);
07  //注册
08  client.register(handler);
```

扩展学习

尝试实现收到消息后，设置震动提醒用户。

实例 155　检测 Wi-Fi 信号强度

应用场景：Wi-Fi 检测

源码位置：光盘\Code\16\155

实例说明

本实例为检测 Wi-Fi 信号强度，通过本实例可以检测当前连接 Wi-Fi 的信号强度。运行实例，效果如图 16.16 所示；每隔 5 秒检测一次信号强度并弹出提示，如图 16.17 所示。

图 16.16　运行效果

图 16.17　信号强度提示

关键技术

关键技术

本实例主要使用 Activity 类中的 getSystemService() 方法、Timer 类中的 scheduleAtFixeRate() 方法、Handler 类中的 sendMessage() 方法、View 类中的 setBackgroundResource() 方法、WifiInfo 类中的 getRssi() 方法，实现检测 Wi-Fi 信号强度。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“WiFiDemo”，公司域名为“mingrisoft.com”。

（2）导入资源文件：复制“光盘\Code\16\155\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）在“app\java\com\mingrisoft\wifidemo”目录下默认生成的 MainActivity 类中，在 onCreate() 方法中初始化控件并创建 Timer 类。代码如下：

```
01  //图片控件初始化
02  wifi_image = (ImageView) findViewById(R.id.wifi_image);
03  wifiManager = (WifiManager) getSystemService(WIFI_SERVICE);
04  //使用定时器，每隔5秒获得一次信号强度值
05  Timer timer = new Timer();
```

（4）实现 Timer 类中的 scheduleAtFixedRate() 方法，完成每隔 5 秒检测一次 Wi-Fi 信号强度。代码如下：

```
01  timer.scheduleAtFixedRate(new TimerTask() {
02      @Override
03      public void run() {
04          wifiInfo = wifiManager.getConnectionInfo();
05          //获取信号强度值
06          //获取的值是-100~0的区间值，为int型数据，其中-50~0表示信号最好，
07          //-70~-50表示信号偏差，小于-70表示最差，导致无法连接
08          level = wifiInfo.getRssi();
09          //根据获得的信号强度发送信息
10          if (level <= 0 && level >= -50) {
11              Message msg = new Message();
12              msg.what = 1;
13              handler.sendMessage(msg);
14          } else if (level < -50 && level >= -70) {
15              Message msg = new Message();
16              msg.what = 2;
17              handler.sendMessage(msg);
18          } else if (level < -70 && level >= -80) {
19              Message msg = new Message();
20              msg.what = 3;
21              handler.sendMessage(msg);
22          } else if (level < -80 && level >= -100) {
23              Message msg = new Message();
24              msg.what = 4;
25              handler.sendMessage(msg);
26          } else {
27              Message msg = new Message();
28              msg.what = 5;
29              handler.sendMessage(msg);
30          }
```

```
    }
}, 1000, 5000);
```

（5）实现 Handler 类，接收信号强度消息，更新 UI 修改信号强度图片并弹出提示，代码如下：

```
handler = new Handler() {
    @Override
    public void handleMessage(Message msg) {
        switch (msg.what) {
            //如果收到正确的消息就获取Wi-Fi信息，改变图片并显示信号强度
            case 1:
                wifi_image.setBackgroundResource(R.drawable.single4);
                Toast.makeText(MainActivity.this,
                        "信号强度：" + level + "信号最好", Toast.LENGTH_SHORT).show();
                break;
            case 2:
                wifi_image.setBackgroundResource(R.drawable.single3);
                Toast.makeText(MainActivity.this,
                        "信号强度：" + level + "信号较好", Toast.LENGTH_SHORT.show();
                break;
            case 3:
                wifi_image.setBackgroundResource(R.drawable.single2);
                Toast.makeText(MainActivity.this,
                        "信号强度：" + level + "信号一般", Toast.LENGTH_SHORT).show();
               break;
           case 4:
               wifi_image.setBackgroundResource(R.drawable.single1);
               Toast.makeText(MainActivity.this,
                       "信号强度：" + level + "信号较差", Toast.LENGTH_SHORT).show();
               break;
           case 5:
               wifi_image.setBackgroundResource(R.drawable.single0);
               Toast.makeText(MainActivity.this,
                       "信号强度：" + level + "无信号", Toast.LENGTH_SHORT).show();
                   break;
             default:
                   wifi_image.setBackgroundResource(R.drawable.single0);
                   Toast.makeText(MainActivity.this, "无信号",
                   Toast.LENGTH_SHORT).show();
                   break;
         }
    }
};
```

扩展学习

请尝试添加显示当前连接 Wi-Fi 的名称。

实例 156	Wi-Fi 共享 应用场景：Wi-Fi 检测	源码位置：光盘\Code\16\156

实例说明

本实例实现了手机端与 PC 端的交互，以手机端充当服务器，PC 端作为客户端访问手机端开放的文件夹下的相关文件。运行实例，效果如图 16.18 所示；单击“开始”按钮显示 PC 端要输入的链接，如图 16.19 所示；在 PC 端浏览器输入提示的地址，如图 16.20 所示；在 PC 端单击“删除”按钮，查看手机相关文件夹里面的文件消失，如图 16.21 所示。

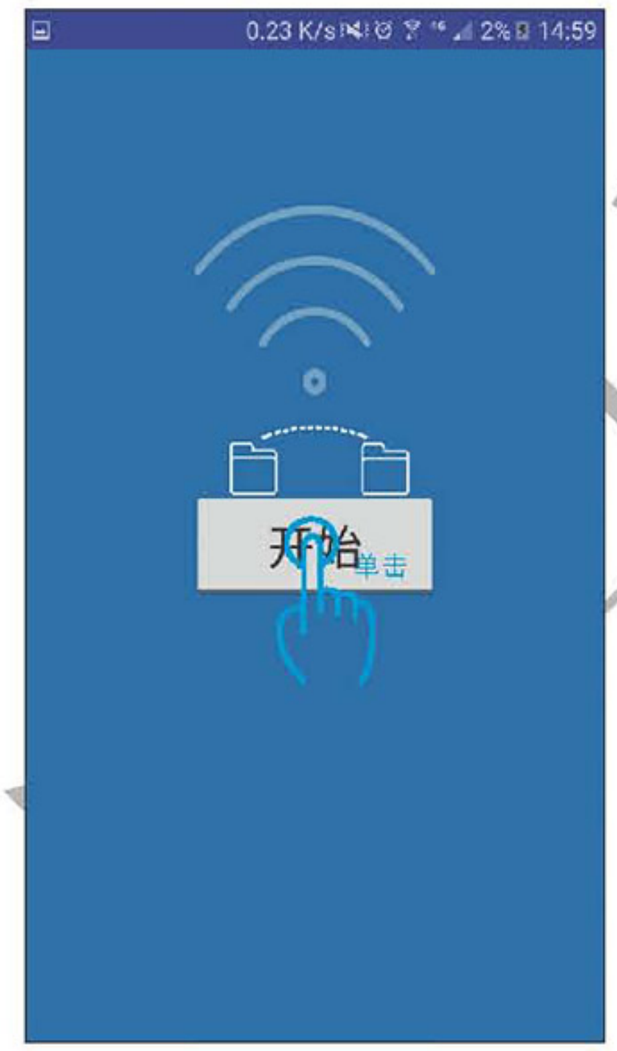


图 16.18　运行效果

图 16.19　显示 PC 端要输入的连接

图 16.20　PC 端效果

图 16.21　删除文件

关键技术

关键技术

本实例主要使用 ContextWrapper 类中的 startService() 方法、ServerSocket 类中的 ServerSocket () 方法、HttpService 类中的 HttpService() 方法、Thread 类中的 setDaemon() 方法实现 Wi-Fi 共享功能。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“WIFIServer”，公司域名为“mingrisoft.com”。

（2）导入资源文件：复制“光盘\Code\16\156\src”中的所有文件夹，粘贴到“app\libs”目录下。

（3）在“app\java\com\mingrisoft\wifiserver”文件夹下默认生成的 MainActivity 类中，实现单击“开始”按钮开启服务。代码如下：

```
@Override
public void onCheckedChanged(CompoundButton buttonView, boolean isChecked) {
    Log.e(TAG, "==============isChecked=================="+isChecked);
    if (isChecked) {
    //获取手机IP地址
    String ip = getLocalIpAddressIP4();
    if (ip == null) {
        Toast.makeText(this, R.string.msg_net_off, Toast.LENGTH_SHORT).show();
        urlText.setText("");
    } else {
        //开启服务
        startService(intent);
        urlText.setText("http://" + ip + ":" + WebService.PORT + "/");
    } else {
        stopService(intent);
        urlText.setText(" ");
    }
}
```

（4）在“app\java\com\mingrisoft\wifiserver”文件夹下创建文件“server”文件夹，并在该文件夹下创建 WebService 服务类，让其继承 Service 作为启动的服务。

（5）在“app\java\com\mingrisoft\wifiserver\server”文件夹下创建 WebServer 类，作为服务启动的线程类，让其继承 Thread 类，在实现的 run() 方法中完成服务器的创建。代码如下：

```
@Override
public void run() {
    ServerSocket serverSocket = null;
    try {
        //创建服务器套接字
        serverSocket = new ServerSocket(port);
        //创建HTTP协议处理器
        BasicHttpProcessor httpproc = new BasicHttpProcessor();
        //增加HTTP协议拦截器
        httpproc.addInterceptor(new ResponseDate());
        httpproc.addInterceptor(new ResponseServer());
        httpproc.addInterceptor(new ResponseContent());
        httpproc.addInterceptor(new ResponseConnControl());
        //创建HTTP服务
        HttpService httpService = new HttpService(httpproc,new DefaultConnectionReuseStrategy(),
                new DefaultHttpResponseFactory());
        //创建HTTP参数
        HttpParams params = new BasicHttpParams();0
        params.setIntParameter(CoreConnectionPNames.SO_TIMEOUT, 5000)    //设置服务器读取时间
              .setIntParameter(CoreConnectionPNames.SOCKET_BUFFER_SIZE, 8 * 1024)
              .setBooleanParameter(CoreConnectionPNames.STALE_CONNECTION_CHECK, false)
              .setBooleanParameter(CoreConnectionPNames.TCP_NODELAY, true)
              .setParameter(CoreProtocolPNames.ORIGIN_SERVER, "WebServer/1.1");
        //设置HTTP参数
        httpService.setParams(params);
        //创建HTTP请求执行器
        HttpRequestHandlerRegistry reqistry = new HttpRequestHandlerRegistry();
        //增加HTTP请求执行器
        reqistry.register("*" + SUFFIX_ZIP, new HttpZipHandler(webRoot));
        reqistry.register("*" + SUFFIX_DEL, new HttpDelHandler(webRoot));
        reqistry.register("*", new HttpFileHandler(webRoot));
        //设置HTTP请求执行器
        httpService.setHandlerResolver(reqistry);
        /* 循环接收各客户端 */
        isLoop = true;
        while (isLoop && !Thread.interrupted()) {
            //接收客户端套接字
            Socket socket = serverSocket.accept();
            //绑定至服务器端HTTP连接
            DefaultHttpServerConnection conn = new DefaultHttpServerConnection();
            conn.bind(socket, params);
            //派送至WorkerThread处理请求
            Thread t = new WorkerThread(httpService, conn);
            //设为守护线程
            t.setDaemon(true);
```

明日科技

```
            t.start();
        }
    } catch (IOException e) {
        isLoop = false;
        e.printStackTrace();
    } finally {
        try {
            if (serverSocket != null) {
                serverSocket.close();
            }
        } catch (IOException e) {
        }
    }
}
```

扩展学习

请尝试修改 PC 端顶部显示文字。

实例 157　周围 Wi-Fi 检测连接

应用场景：Wi-Fi 检测

源码位置：光盘\Code\16\157

实例说明

通过本实例可以检测周围可以连接的 Wi-Fi 并进行连接。运行实例，效果如图 16.22 所示，检测到 Wi-Fi 列表；单击某一条 Wi-Fi 信息，将弹出询问是否连接对话框，如图 16.23 所示；单击“确定”按钮弹出密码输入对话框，如图 16.24 所示；Wi-Fi 连接成功，如图 16.25 所示。

图 16.22　运行效果

图 16.23　连接对话框

图 16.24　密码输入框

图 16.25　连接成功

关键技术

本实例主要使用 Activity 类中的 getSystemService() 方法、AlertDialog 类中的 setView() 方法、setTitle() 方法、View 类中的 setBackgroundResource() 方法、WifiInfo 类中的 getRssi() 方法，实现检测 Wi-Fi 信号强度。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“WIFIConnent”，公司域名为“mingrisoft.com”。

（2）导入资源文件：复制“光盘\Code\16\157\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）在“app\java\com\mingrisoft\wificonnent”文件夹下默认生成的 MainActivity 类中，创建 initView() 方法用于初始化资源。代码如下：

```
01  private void initView() {
02      //初始化资源
03      wifiList = (ListView) this.findViewById(R.id.wifi_conn_lv);
04      wifi_switch_btn = (Button) this.findViewById(R.id.wifi_conn_switch_btn);
05      wifi_scan_btn = (Button) this.findViewById(R.id.wifi_conn_scan_btn);
06      wifi_cancle_btn = (Button) this.findViewById(R.id.wifi_conn_cancle_btn);
07      showConn = (TextView) this.findViewById(R.id.wifi_show_conn);
08      if (mWifiAdmin.getWifiState() == WifiManager.WIFI_STATE_DISABLED) {
09          wifi_scan_btn.setText("打开Wi-Fi");
10      } else {
11          wifi_scan_btn.setText("关闭Wi-Fi");
12          isOpen = true;
13      }
14      showConn.setText("已连接:     " + initShowConn());
15      //绑定单击事件
16      wifi_cancle_btn.setOnClickListener(this);
17      wifi_switch_btn.setOnClickListener(this);
18      wifiList.setOnItemClickListener(this);
19      wifi_scan_btn.setOnClickListener(this);
20      //Wi-Fi信息适配器
21      mConnList = new WifiConnListAdapter(getApplicationContext(), getAllNetWorkList());
22      //为列表绑定适配器
23      wifiList.setAdapter(mConnList);
24  }
```

在 MainActivity 类中，创建 checknet() 方法，用来检测网络连接。代码如下：

```
01  /**
02   * 监测网络连接
03   *
04   * @return : 如果为true，则连接正常；如果为false，则连接断开
```

```
 */
private boolean checknet() {
    ConnectivityManager connManager = (ConnectivityManager) this.getSystemService(
            CONNECTIVITY_SERVICE);
    //获取代表联网状态的NetWorkInfo对象
    networkInfo = connManager.getActiveNetworkInfo();
    if (null != networkInfo) {
        return networkInfo.isAvailable();
    }
    return false;
}
```

在 MainActivity 类中创建 initShowConn() 方法，用来显示 Wi-Fi 连接信息。代码如下：

```
//显示Wi-Fi连接信息
private String initShowConn() {
    //获取Wi-Fi服务
    WifiManager wifiManager = (WifiManager) getSystemService(WIFI_SERVICE);
    //获取Wi-Fi管理
    WifiInfo wifiInfo = wifiManager.getConnectionInfo();
    String s = wifiInfo.getSSID() + "\n"+"IP地址:" + mWifiAdmin.ipIntToString(wifiInfo.getIpAddress())
            + "\n" + "Mac地址: " + wifiInfo.getMacAddress();
    return s;
}
```

扩展学习

请尝试添加显示已连接的 Wi-Fi 名称。

实例 158　触碰 NFC 标签打开设置的应用

应用场景：NFC 使用

源码位置：光盘\Code\16\158

实例说明

本实例为 NFC 功能应用，通过本实例可以把手机中要打开的应用写入空白的 NFC 标签中，并通过手机再次触碰空白 NFC 便签打开写入的应用。运行实例，效果如图 16.26 所示；单击按钮进入选择应用列表页面，如图 16.27 所示；选择应用后触碰空白标签，写入成功后弹出提示，如图 16.28 所示。

关键技术

关键技术

本实例主要使用 PendingIntent 类中的 getActivity() 方法、NdefMessage 类中的 NdefMessage () 方法、NdefRecord 类中的 createApplicationRecord() 方法、Ndef 类中的 get() 方法、writeNdefMessage() 方法，实现触碰 NFC 标签打开设置的应用。

图 16.26　运行效果

图 16.27　选择应用

图 16.28　成功提示

实现过程

（1）打开 Android Studio，新建一个项目，命名为“NFCRunAPP”，公司域名为“mingrisoft.com”。

（2）导入资源文件：复制“光盘\Code\16\158\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）本实例要完成的是 NFC 功能，所以需要添加 NFC 权限，在“app\manifests”文件夹下默认生成的 AndroidManifest.xml 配置文件中加入 NFC 权限。代码如下：

```
<uses-permission android:name="android.permission.NFC" />
```

（4）在“app\java\com\mingrisoft\nfcread”文件夹下默认生成的MainActivity类中，创建writeNFCTag()方法，用于往空白 NFC 标签中写入要打开的应用包名。代码如下：

```
01  /**
02   * 在标签中写数据的方法
03   *
04   * @param tag
05   */
06  public void writeNFCTag(Tag tag) {
07      if (tag == null) {
08          return;
09      }
10      //封装标签消息
11      NdefMessage ndefMessage = new NdefMessage(
12          new NdefRecord[] { NdefRecord.createApplicationRecord(appPackageName)
13      });
```

```
        //转换成字节获得大小
        int size = ndefMessage.toByteArray().length;
        try {
            //判断NFC标签的数据类型（通过Ndef.get方法实现）
            Ndef ndef = Ndef.get(tag);
            //判断是否为NDEF标签
            if (ndef != null) {
                ndef.connect();
                //判断是否支持可写
                if (!ndef.isWritable()) {
                    return;
                }
                //判断标签的容量是否够用
                if (ndef.getMaxSize() < size) {
                    return;
                }
                //写入数据
                ndef.writeNdefMessage(ndefMessage);
                Toast.makeText(this, "写入成功，请退出应用后再扫描标签", Toast.LENGTH_LONG).show();
            }else {
            //Ndef格式类
            NdefFormatable format = NdefFormatable.get(tag);
            //判断是否获得了NdefFormatable对象，有一些标签是只读的或者不允许格式化的
                if (format != null) {
                    //连接
                    format.connect();
                    //格式化并将信息写入标签
                    format.format(ndefMessage);
                    Toast.makeText(this, "写入成功", Toast.LENGTH_SHORT).show();
                } else {
                   Toast.makeText(this, "写入失败", Toast.LENGTH_SHORT).show();
                }
            }
        } catch (Exception e) {
            //TODO: handle exception
        }
    }
```

（5）在“app\java\com\mingrisoft\nfcread”文件夹下创建 AppListView 类，继承 ListAcitvity 类，用于显示手机上所有应用的包名。单击某一个应用返回应用包名。代码如下：

```
@Override
public void onItemClick(AdapterView<?> parent, View view, int position, long id) {
    //回传信息
```

```
04        Intent intent = new Intent();
05        //要返回的信息
06        intent.putExtra("package_name", mPackages.get(position));
07        setResult(1, intent);
08        //关闭当前页面
09        finish();
10    }
```

扩展学习

请尝试为应用列表添加列表项，区分包名和应用名称。

明日科技

第17章 游戏开发

实例159 俄罗斯方块

应用场景：游戏

源码位置：光盘\Code\17\159

实例说明

俄罗斯方块是人们所喜爱的一款经典游戏。本实例将实现 Android 版俄罗斯方块游戏的开发。运行实例，显示如图 17.1 所示的界面；单击“开始游戏”按钮，进入游戏页面，如图 17.2 所示；分数达到能进入排行榜的时候，显示让用户输入姓名的画面，如图 17.3 所示。

图 17.1 运行效果

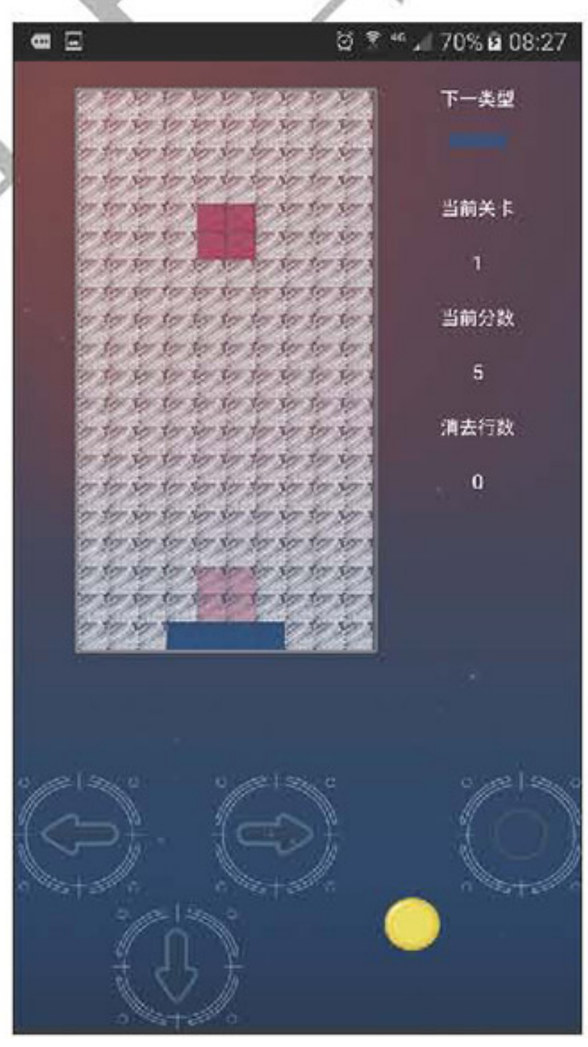


图 17.2 游戏中画面

图 17.3 游戏结束画面

关键技术

本实例主要使用 ContextWrapper 类中的 getSharedPreferences() 方法、Timer 类中的 schedule() 方法、Canvas 类中的 drawBitmap() 方法、SoundPool 类中的 SoundPool() 方法、MediaPlayer 类中的 create() 方法，实现俄罗斯方块游戏。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“MyTetris”，公司域名为“mingrisoft.com”。

（2）导入资源文件：复制“光盘\Code\17\159\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）在“app\java\com\mingrisoft\mytetris”文件夹下创建 MyGameView 类，继承 View，用于实现游戏界面和游戏声音管理。

（4）在 MyGameView 类中定义 4 个内部类，分别是 ImageManager 类（用于管理图像加载）、SoudManager 类（用于管理声音）、Ground 类（用于定义背景墙）和 Shape 类（用于管理下落方块）。

（5）在 MyGameView 类的内部类 ImageManager 中获取图片资源。代码如下：

```
InputStream is = getResources().getAssets().open("arrow_left.png");          //加载左按钮图片
arrowLeft = BitmapFactory.decodeStream(is);
is = getResources().getAssets().open("arrow_right.png");                     //加载右按钮图片
arrowRight = BitmapFactory.decodeStream(is);
is = getResources().getAssets().open("arrow_down.png");                      //加载向下按钮图片
arrowDown = BitmapFactory.decodeStream(is);
is = getResources().getAssets().open("arrow_change.png");                    //加载转动按钮图片
arrowChange = BitmapFactory.decodeStream(is);
is = getResources().getAssets().open("btn_md_up.png");                       //加载暂停按钮图片
arrowPause = BitmapFactory.decodeStream(is);
is = getResources().getAssets().open("gameover.png");                        //加载游戏结束图片
gameOverBitmap = BitmapFactory.decodeStream(is);
is = getResources().getAssets().open("pause.png");                           //加载游戏暂停图片
```

（6）在 MyGameView 类中定义内部类 SoundManager，在其中创建 playBackMusic() 方法，用于播放背景音乐。代码如下：

```
//播放背景音乐
public void playBackMusic() {
    mediaPlayer = MediaPlayer.create(context, R.raw.back);
    mediaPlayer.setLooping(true);
    mediaPlayer.start();
}
```

扩展学习

请尝试修改游戏初始方块落下速度，增加游戏难度。

实例 160 疯狂单击小游戏

应用场景：游戏

源码位置：光盘\Code\17\160

实例说明

本实例将实现疯狂单击并记录单击次数的功能，用以锻炼手指的灵活性。运行实例，将显示图 17.4 所示的界面；单击“Go”按钮开始游戏，进入游戏页面，如图 17.5 所示；单击“Go”按钮所在的红色区域，单击一次记数一次，游戏时间为 10 秒，游戏结束后显示单击数，如图 17.6 所示。

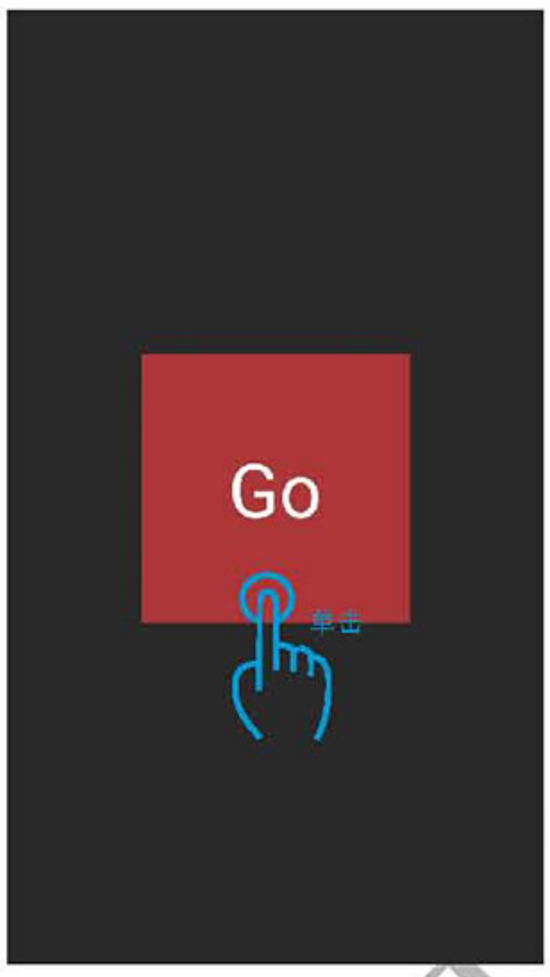


图 17.4　运行效果

图 17.5　游戏中画面

图 17.6　游戏结束画面

关键技术

本实例主要使用 ContextWrapper 类中的 getSharedPreferences() 方法、TextView 类中的 setText() 方法、View 类中的 setVisibility() 方法、setOnClickListener() 方法、AlertDialog 类中的 Builder() 方法，实现疯狂单击小游戏。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“MyFKDJ”，公司域名为“mingrisoft.com”。

（2）导入资源文件：复制“光盘\Code\17\160\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）在“app\res\layout”默认生成的 activity_main.xml 布局文件中，添加点击区域以及显示文字。

（4）在“app\java\com\mingrisoft\myfkdj”文件夹下默认生成的 MainActivity 类中，初始化控件绑定单击事件。代码如下：

```
//开启保存文件方法类型为应用私有，没有者创建用于保存文件
sp = ctx.getSharedPreferences("SP", MODE_PRIVATE);
//初始化控件
text_dwon = (TextView) findViewById(R.id.text_dwon);
text_time = (TextView) findViewById(R.id.text_time);
text_ci = (TextView) findViewById(R.id.text_ci);
text_play = (TextView) findViewById(R.id.text_play);
moemy_ci = (TextView) findViewById(R.id.moemy_ci);
text_topBack  = (ImageButton) findViewById(R.id.text_topBack);
if (!sp.getString("user","").equals("")) {
    //获取最高分
    k = sp.getString("user", "");
    //转换为int类型
    j = Integer.valueOf(k).intValue();
    moemy_ci.setText("历史最高得分：" + k + "次");
}
//设置控件隐藏
text_time.setVisibility(View.GONE);
text_dwon.setVisibility(View.GONE);
text_ci.setVisibility(View.GONE);
moemy_ci.setVisibility(View.GONE);
//绑定点击事件
text_topBack.setOnClickListener(new OnClickListener() {
    @Override
    public void onClick(View arg0) {
        //TODO Auto-generated method stub
        i += 1;
        if(i<5){
            text_dwon.setBackgroundResource(R.drawable.t1);
        }else if(i<10){
            text_dwon.setBackgroundResource(R.drawable.t2);
        }else if(i<15){
            text_dwon.setBackgroundResource(R.drawable.t3);
        }else if(i<20){
            text_dwon.setBackgroundResource(R.drawable.t4);
        }else if(i<25){
            text_dwon.setBackgroundResource(R.drawable.t5);
        }else if(i<30){
            text_dwon.setBackgroundResource(R.drawable.t6);
        }else if(i<35){
            text_dwon.setBackgroundResource(R.drawable.t7);
        }else {
            text_dwon.setBackgroundResource(R.drawable.t8);
        }
        text_ci.setText(i + "次");
    }
});
```

明日科技

```
48  text_play.setOnClickListener(new OnClickListener() {
49      @Override
50      public void onClick(View arg0) {
51          //TODO Auto-generated method stub
52          //线程每个1秒执行一次
53          timer.schedule(task, 1000, 1000);                    //timeTask
54          //设置控件隐藏
55          text_play.setVisibility(View.GONE);
56          //设置控件显示
57          text_time.setVisibility(View.VISIBLE);
58          text_dwon.setVisibility(View.VISIBLE);
59          text_ci.setVisibility(View.VISIBLE);
60          moemy_ci.setVisibility(View.VISIBLE);
61      }
62  });
```

（5）创建计时线程，记录游戏时间。代码如下：

```
01  TimerTask task = new TimerTask() {
02      @Override
03      public void run() {
04          runOnUiThread(new Runnable() {
05              @Override
06              public void run() {
07                  recLen--;
08                  text_time.setText(recLen + 1 + " 秒");
09                  if (recLen < 0) {                            //倒计时结束后的事件
10                      timer.cancel();
11                      text_dwon.setClickable(false);           //设置为不可点
12                      dialog();                                //游戏结束提示
13                      //记住最高分
14                      if (i > j) {
15                          //保存最高分
16                          SharedPreferences.Editor editor = sp.edit();
17                          editor.putString("user", String.valueOf(i));
18                          editor.commit();
19                      }
20                  }
21              }
22          });
23      }
24  };
```

扩展学习

请尝试将游戏时间从 10 秒增到 15 秒。

实例 161　五子棋游戏

应用场景：游戏

源码位置：光盘\Code\17\161

实例说明

本实例为对战类的五子棋游戏。运行实例，将显示 17.7 所示的界面，玩家左上角显示的是当前要落子的颜色，单击棋盘上的位置，显示棋子，如图 17.8 所示，有一方胜利后，会提示胜利方，如图 17.9 所示。

图 17.7　开始游戏

图 17.8　游戏中

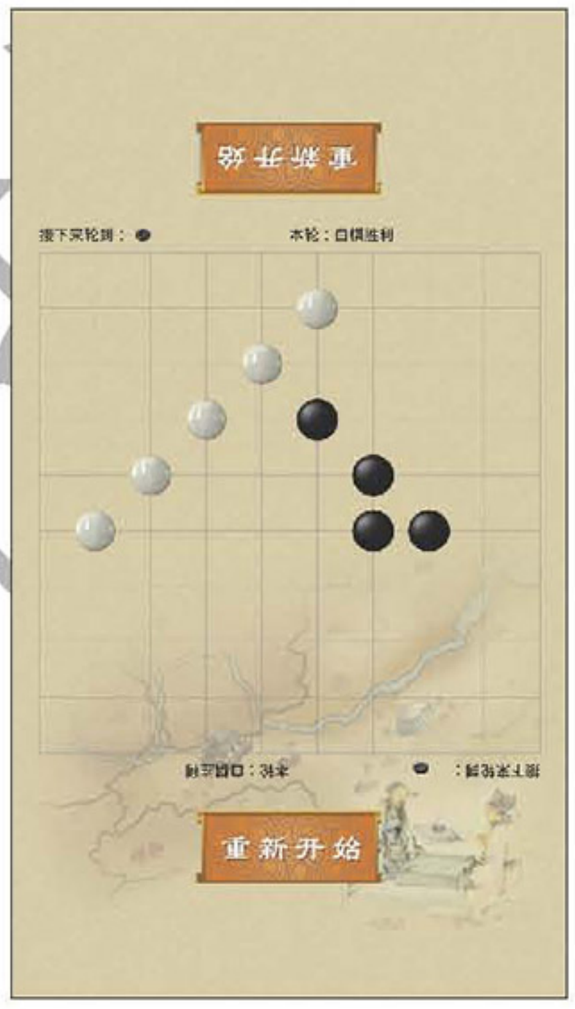


图 17.9　游戏结束画面

关键技术

关键技术

本实例主要使用 List 类中的 add() 方法，Point 类中的 Point() 方法，Canvas 类中的 drawText() 方法、rotate() 方法、drawLine() 方法，实现五子棋游戏。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“MyGobang”，公司域名为“mingrisoft.com”。

（2）导入资源文件：复制“光盘\Code\17\161\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）在“app\java\com\mingrisoft\mygobang”文件夹下创建 MyGobangView 类，继承 View，用于实现五子棋核心游戏的页面以及逻辑。

（4）在 MyGobangView 类的 init() 方法中，同时加载初始化相关属性以及加载控件。代码如下：

```
01  private void init() {
02      //设置画笔颜色
03      mPaint.setColor(Color.GRAY);
04      //去掉锯齿
05      mPaint.setAntiAlias(true);
06      //防抖动
07      mPaint.setDither(true);
08      //设置画笔样式
09      mPaint.setStyle(Paint.Style.STROKE);
10      mTpaint.setTextSize(30f);
11      mTpaint.setColor(Color.BLACK);
12      //去掉锯齿
13      mTpaint.setAntiAlias(true);
14      //防抖动
15      mTpaint.setDither(true);
16      //设置画笔样式
17      mTpaint.setStyle(Paint.Style.STROKE);
18      //初始化图片资源
19      mWhitePiece = BitmapFactory.decodeResource(getResources(), R.drawable.stone_w2);
20      mBlackPiece = BitmapFactory.decodeResource(getResources(), R.drawable.stone_b1);
21      mBoxBlack= BitmapFactory.decodeResource(getResources(), R.drawable.box_b1);
22      mBoxWhite= BitmapFactory.decodeResource(getResources(), R.drawable.box_w2);
23  }
```

（5）让 MyGobangView 类继承 View，并重写 onDraw() 方法，用于绘制页面。代码如下：

```
01  @Override
02  protected void onDraw(Canvas canvas) {
03      super.onDraw(canvas);
04      //绘制背景
05      drawBoard(canvas);
06      //绘制棋子
07      draePicec(canvas);
08      //判断是否结束
09      checkGameOver(canvas);
10      //绘制棋盘
11      outPicec(canvas);
12  }
```

（6）在“app\res\layout”默认生成的 activity_main.xml 布局类中写入自定义游戏布局。代码如下：

```
01  <!--游戏页面-->
02  <com.mingrisoft.mygobang.MyGobangView
03      android:id="@+id/action_settings"
04      android:layout_width="match_parent"
05      android:layout_height="match_parent"
```

明日科技

```
06        android:layout_centerInParent="true"
07        android:layout_weight="6"
08    />
```

扩展学习

请在有棋子胜利后尝试实现弹出提示弹窗，显示胜利的一方。

实例 162　仿像素鸟游戏

应用场景：游戏

源码位置：光盘\Code\17\162
视频位置：光盘\Video\17\162

实例说明

像素鸟游戏是一款人们所喜欢的游戏，单击屏幕控制小鸟的上下移动，小鸟飞过一个管子就会得一分，碰到管子后立刻结束游戏。本实例通过 SurfaceView+SurfaceHolder+Thread 实现仿像素鸟小游戏。运行实例，将显示如图 17.10 所示的界面；点击屏幕，飞过管子得分，如图 17.11 所示；碰到管子结束游戏，进入结束页，如图 17.12 所示。

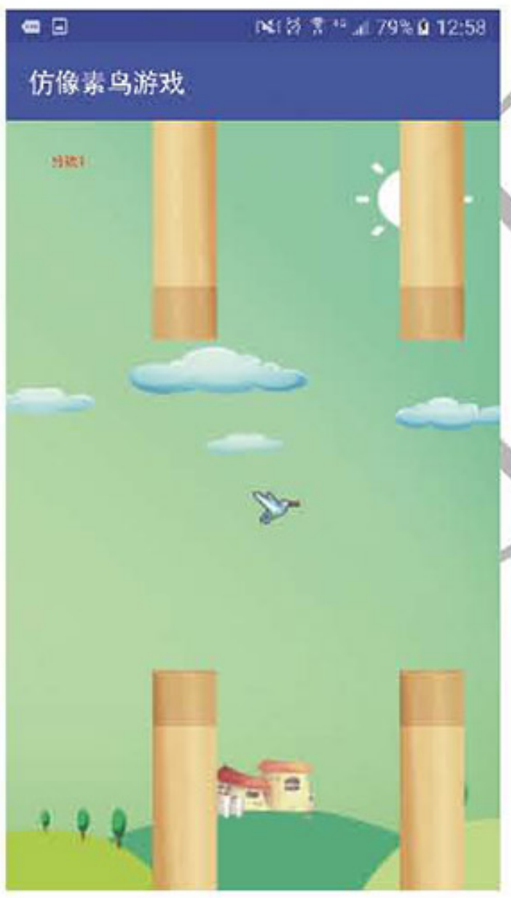


图 17.10　运行效果

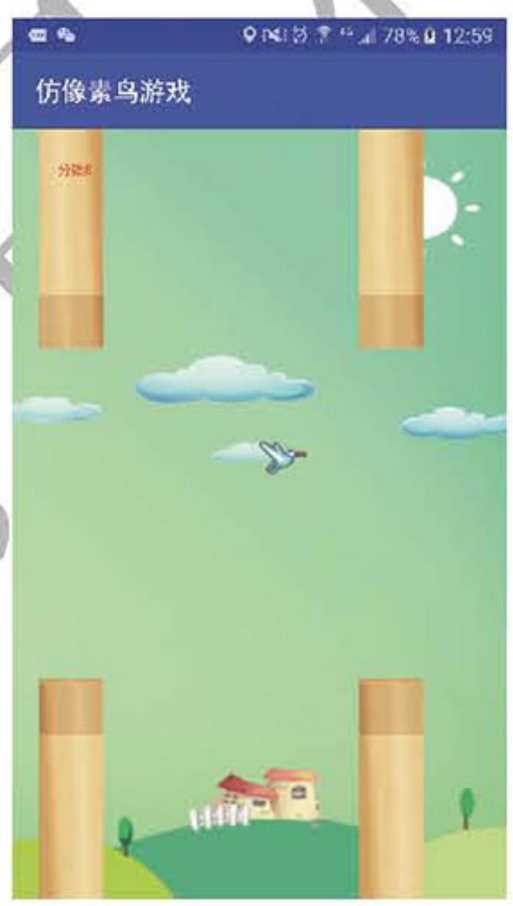


图 17.11　通过管子得分画面

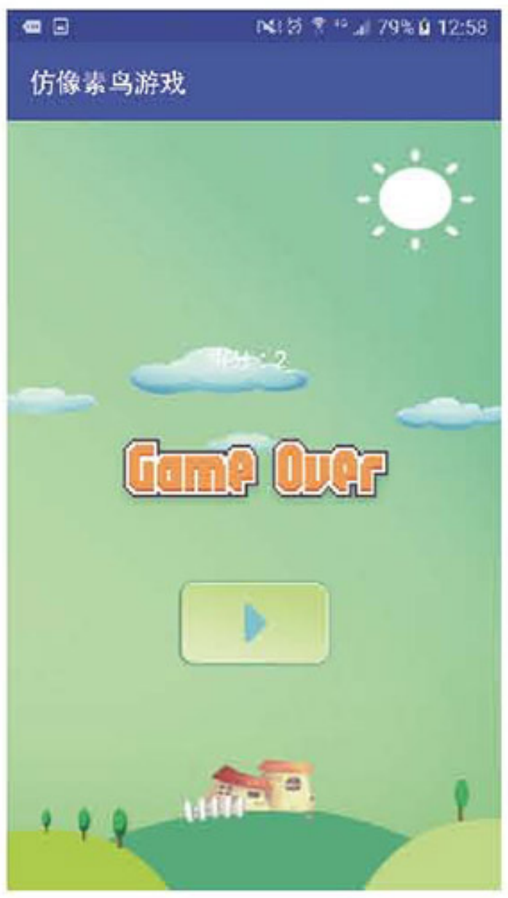


图 17.12　游戏结束画面

关键技术

本实例主要使用 SurfaceHolder 接口类中的 surfaceCreated() 方法、Thread 类中的 Thread() 方法、Canvas 类中的 drawBitmap() 方法、Paint 类中的 setColor() 方法，实现像素鸟游戏。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“MyBird”，公司域名为“mingrisoft.com”。

（2）导入资源文件：复制“光盘\Code\17\162\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）在“app\java\com\mingrisoft\mybird”文件夹下创建 BirdGameView 类，继承 SurfaceView，并实现 SurfaceHolder.Callback 接口，用于实现游戏。代码如下：

```
01  public class BirdGameView extends SurfaceView implements SurfaceHolder.Callback
```

实现 SurfaceHolder.Callback 接口类的 surfaceCreated() 方法，创建 SurfaceView 控件时会调用该方法，并在其中加载资源开启游戏逻辑子线程。代码如下：

```
01  //创建的时候调用方法
02  @Override
03  public void surfaceCreated(SurfaceHolder holder) {
04      //TODO Auto-generated method stub
05      //加载图片资源
06      game_over = BitmapFactory.decodeResource(getResources(), R.drawable.bird_gameover);
07      start_btn = BitmapFactory.decodeResource(getResources(), R.drawable.bird_start_btn);
08      bj = BitmapFactory.decodeResource(getResources(), R.drawable.bird_bg);
09      bird_im = BitmapFactory.decodeResource(getResources(), R.drawable.bird_hero);
10      //获取小鸟图片宽高用于绘制以及判断
11      bird_h = bird_im.getHeight();
12      bird_w = bird_im.getWidth();
13      top_za = BitmapFactory.decodeResource(getResources(), R.drawable.obstacle_up);
14      bom_za = BitmapFactory.decodeResource(getResources(), R.drawable.obstacle_down);
15      //获取障碍宽高用于判断游戏结束用
16      up_h = top_za.getHeight();
17      up_w = top_za.getWidth();
18      //开启线程标识
19      isLogic = true;
20      isDrawUI = true;
21      //绘制线程
22      threadDrawUI = new ThreadDrawUI();
23      //逻辑线程
24      threadLogic = new ThreadLogic();
25      //开启线程
26      threadDrawUI.start();
27      threadLogic.start();
28  }
```

创建 GameOver() 方法，用于在游戏结束时绘制结束游戏画面。代码如下：

```
01  //游戏结束方法
02  public void GameOver() {
03      //锁定整个画布，在内存要求比较高的情况下，建议参数不要为null
```

```
        isDrawUI = false;
        isLogic = false;
        Canvas canvas = holder.lockCanvas(null);
        //游戏结束画面
        onGameStop(canvas);
        holder.unlockCanvasAndPost(canvas);
        //游戏结束标识
        k = true;
        //清空得分
        score = 0;
    }
    //绘制游戏结束画面
    private void onGameStop(Canvas canvas) {
        RectF rectF = new RectF(0, 0, mWidth, mHeight);
        //绘制背景
        canvas.drawBitmap(bj, null, rectF, null);
        //绘制游戏结束画面
        canvas.drawBitmap(game_over, mWidth / 2 - game_over.getWidth() / 2,
                mHeight / 2 - game_over.getHeight(), null);
        //绘制重新开始按钮
        canvas.drawBitmap(start_btn, mWidth / 2 - start_btn.getWidth() / 2,
                mHeight / 2 + game_over.getHeight(), null);
        mPaint.setColor(Color.WHITE);
        mPaint.setTextSize(50f);
        //绘制得分
        canvas.drawText("得分：" + score, mWidth / 2 - 100,
                mHeight / 2 - game_over.getHeight() * 2, mPaint);
    }
```

（4）在“app\res\layout”默认生成的 activity.xml 布局文件中，导入 BirdGameView 自定义游戏类。代码如下：

```
<?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:gravity="center_vertical|center_horizontal"
    android:background="@drawable/bird_bg">
    <com.mingrisoft.mybird.BirdGameView
        android:layout_width="match_parent"
        android:layout_height="match_parent" />
</RelativeLayout>
```

扩展学习

游戏中障碍管道的高度是固定的，尝试修改成随机障碍高度，增加游戏难度。

实例 163　仿别踩白块儿游戏

应用场景：游戏

源码位置：光盘\Code\17\163

实例说明

别踩白块儿是一款一夜爆红的小游戏，本实例将仿做一款简单的别踩白块儿游戏。运行实例，显示如图 17.13 所示的界面；单击“Play”按钮，进入游戏界面，如图 17.14 所示；单击黑色块会得 1 分，同时色块变灰，如图 17.15 所示；如果单击到白色块或者漏点击黑色色块，则游戏结束，如果 17.16 所示。

图 17.13　运行效果

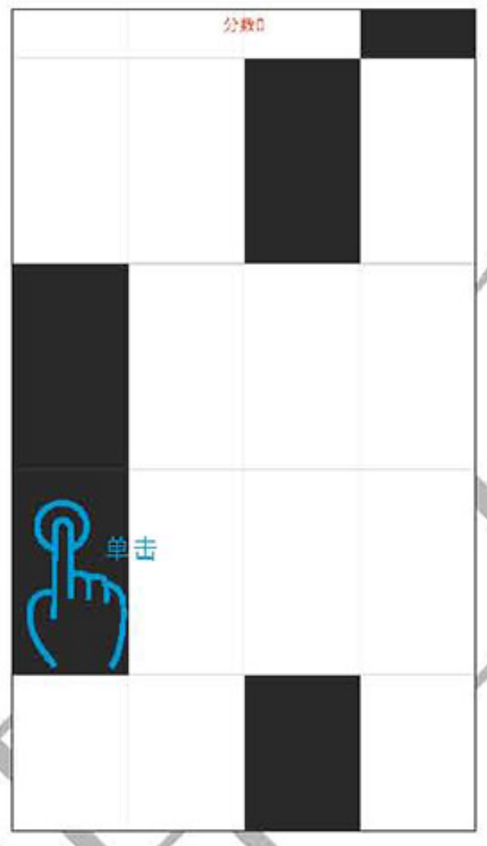


图 17.14　游戏中画面

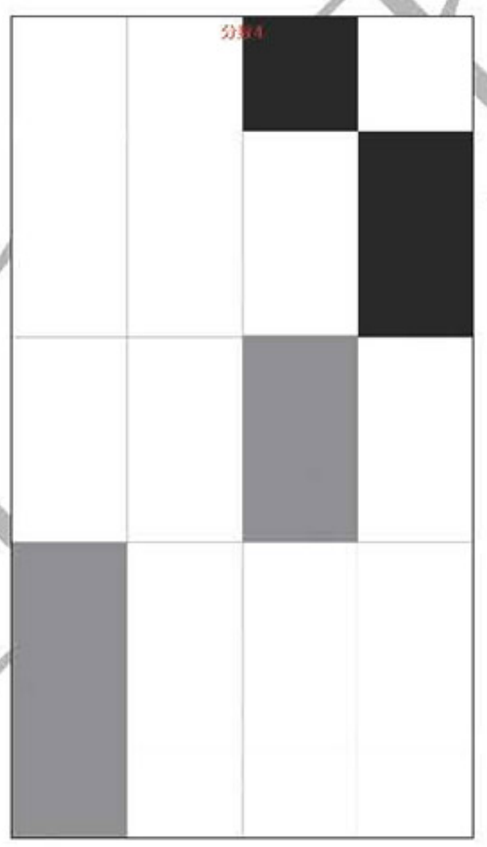


图 17.15　游戏得分页

图 17.16　游戏结束画面

关键技术

本实例主要使用 SurfaceHolder 接口类中的 surfaceCreated() 方法、Thread 类中的 Thread() 方法、Canvas 类中的 drawRect() 方法、View 类中的 onTouchEvent() 方法，实现别踩白块游戏。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“MyWhiteBlock”，公司域名为“mingrisoft.com”。

（2）导入资源文件：复制“光盘\Code\17\163\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）在“app\java\com\mingrisoft\mywhiteblock”文件夹下创建 GameView 类，继承 SurfaceView，用于实现游戏功能。

（4）在 GameView 类的初始化方法中，调用 init() 方法。代码如下：

```
01  public void init() {
02      //游戏菜单页
03      gameMenu = new GameMenu();
04      gameMenu.setChange(this,context);
05      //游戏进行中
06      gamePlay=new GamePlay();
07      gamePlay.setchange(this);
08      //游戏结束页
09      gameOver=new GameOver();
10      gameOver.setchange(this,context);
11      //游戏得分
12      gameScore=new GameScore();
13      hodler = getHolder();
14      hodler.addCallback(this);
15  }
```

（5）在“app\java\com\mingrisoft\mywhiteblock”文件夹下创建 MyRect 类，并创建 ondraws() 方法加载不同的色块。代码如下：

```
01  public void ondraws(Canvas canvas,Rect rect,int flag) {
02      //TODO Auto-generated method stub
03      switch (flag) {
04      //黑色
05      case 1:
06          canvas.drawRect(rect, paint);
07          break;
08      //白色
09      case 2:
10          canvas.drawRect(rect, paintwite);
11          break;
12      //灰色
13      case 3:
14          canvas.drawRect(rect, paintgray);
15          break;
16      //红色
17      case 4:
18          canvas.drawRect(rect, paintred);
19          break;
20      }
21  }
```

（6）在“app\java\com\mingrisoft\mywhiteblock”文件夹下，默认生成的 MainActivity 类中加载游戏布局。代码如下：

```
01  public class MainActivity extends Activity {
```

明日科技

```
02      //定义宽
03      public static int screen_width;
04      //定义高
05      public static int screen_height;
06      @Override
07      protected void onCreate(Bundle savedInstanceState) {
08          super.onCreate(savedInstanceState);
09          //设置没有顶部栏
10          requestWindowFeature(Window.FEATURE_NO_TITLE);
11          //设置全屏显示
12          this.getWindow().setFlags(WindowManager.LayoutParams.FLAG_FULLSCREEN,
13                  WindowManager.LayoutParams.FLAG_FULLSCREEN);
14          //获取屏幕信息
15          WindowManager wm = (WindowManager) this.getSystemService(Context.WINDOW_SERVICE);
16          //屏幕宽
17          screen_width = wm.getDefaultDisplay().getWidth();
18          //屏幕高
19          screen_height = wm.getDefaultDisplay().getHeight();
20          //装载布局
21          setContentView(new GameView(this));
22      }
23  }
```

扩展学习

当前游戏起始速度较慢，并且很简单，请尝试修改起始速度让游戏难度增加。

实例 164 猜牌小游戏

应用场景：游戏

源码位置：光盘\Code\17\164

实例说明

本实例为一个猜牌的小游戏。运行实例，将显示如图 17.17 所示的主界面；单击其中一张牌，如果选中的牌与要求相符，则显示如图 17.18 所示的猜对效果界面，否则显示如图 17.19 所示的选错效果界面。

关键技术

关键技术

本实例主要使用 ImageView 类中的 setImageDrawable() 方法，TextView 类中的 setText() 方法，View 类中的 setVisibility() 方法、setOnClickListener() 方法，Random 类中的 nextInt() 方法，实现小游戏点击猜牌。

图 17.17　运行效果

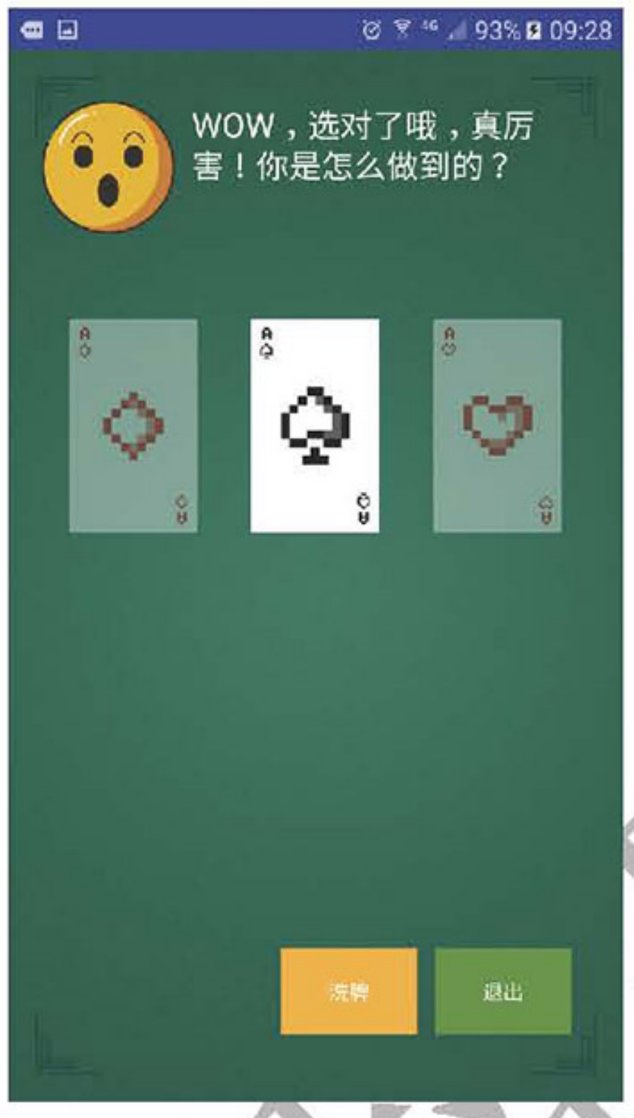


图 17.18　猜对效果

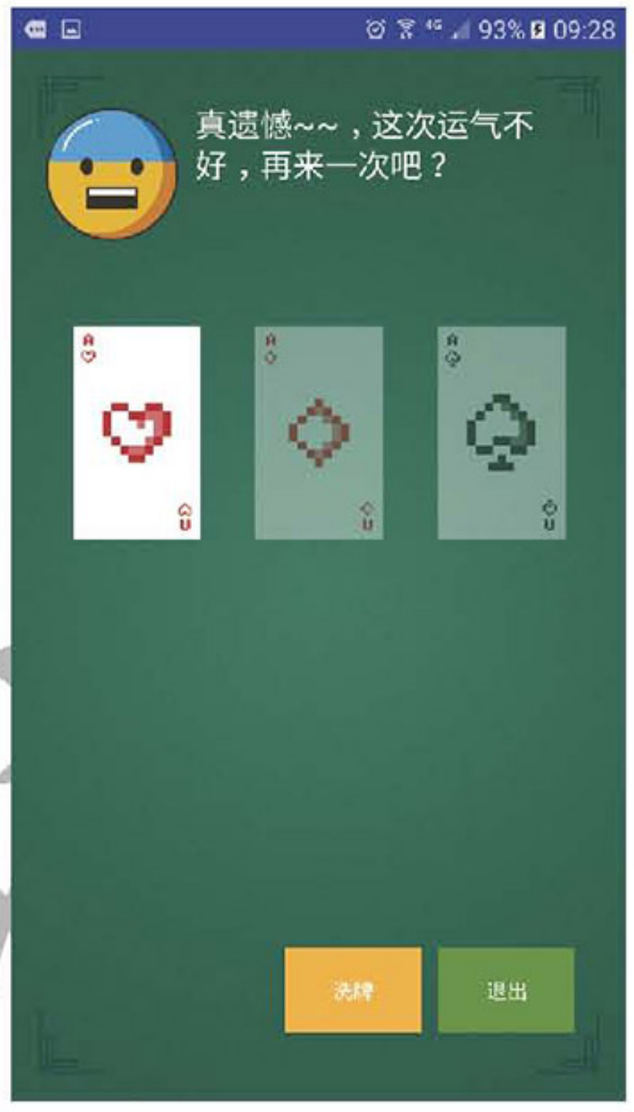


图 17.19　猜错效果

实现过程

（1）打开 Android Studio，新建一个项目，命名为“MyGuessCard”，公司域名为“mingrisoft.com”。

（2）导入资源文件：复制“光盘\Code\17\164\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）在“app\res\layout”默认生成的 activity_main.xml 布局文件中，添加牌布局。代码如下：

```
<!--牌容器-->
<LinearLayout
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:layout_gravity="center_horizontal"
    android:layout_weight="1"
    android:orientation="horizontal"
    android:paddingTop="40dip"
    >
    <!--牌1布局-->
    <ImageView
        android:id="@+id/image1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_weight="1"
```

```
        android:padding="10dip"
        android:src="@drawable/poker_back"
        />
    <!--牌2布局-->
    <ImageView
        android:id="@+id/image2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_weight="1"
        android:padding="10dp"
        android:src="@drawable/poker_back"
        />
    <!--牌3布局-->
    <ImageView
        android:id="@+id/image3"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_weight="1"
        android:padding="10dip"
        android:src="@drawable/poker_back"
        />
</LinearLayout>
```

（4）在“app\java\com\mingrisoft\myguesscard”文件夹下默认生成的 MainActivity 类中，创建 random() 方法实现随机洗牌。代码如下：

```
//随机洗牌函数
private void random(){
    Random rand = new Random();
    int temp1 = 0;
    int temp2 = 0;
    for(int i=0; i<3; i++){
        temp1 = rand.nextInt(3);
        temp2 = poker[i];
        poker[i] = poker[temp1];
        poker[temp1] = temp2;
    }
}
```

为牌添加单击事件，以牌 1 为例。代码如下：

```
//牌1监听器
image_View1.setOnClickListener(new OnClickListener(){
    @Override
```

明日科技

```
04      public void onClick(View v) {
05          //TODO Auto-generated method stub
06          //设置图片
07          image_View1.setImageDrawable(getResources().getDrawable(poker[0]));
08          image_View2.setImageDrawable(getResources().getDrawable(poker[1]));
09          image_View3.setImageDrawable(getResources().getDrawable(poker[2]));
10          //设置透明度
11          image_View2.setAlpha(100);  //设置未被选中牌的渐隐效果
12          image_View3.setAlpha(100);
13          //判断点击是否是黑桃
14          if(poker[0] == R.drawable.poker_a){
15              outPut.setText("WOW，选对了哦，真厉害！你是怎么做到的？");
16              imageFace.setImageDrawable(getResources().getDrawable(R.drawable.qq_suprise));
17          }else{
18              outPut.setText("真遗憾~~，这次运气不好，再来一次吧？");
19              imageFace.setImageDrawable(getResources().getDrawable(R.drawable.qq_despise));
20          }
21      }
22  });
```

扩展学习

请尝试再添加 3 张牌，用以增加游戏难度。

实例 165　拼图小游戏

应用场景：游戏

源码位置：光盘\Code\17\165

实例说明

本实例将实现一个拼图的小游戏。运行实例，效果如图 17.20 所示；单击其中某一碎片，如果旁边有空白区域则替换，如图 17.21 所示；拼图成功后，效果如图 17.22 所示。

关键技术

关键技术

本实例主要使用 ImageView 类中的 setImageDrawable() 方法，TextView 类中的 setText() 方法、View 类中的 setVisibility() 方法、setOnClickListener() 方法，Random 类中的 nextInt() 方法，实现拼图小游戏。

日日科技

图 17.20　运行效果

图 17.21　移动后效果

图 17.22　拼图完成效果

实现过程

（1）打开 Android Studio，新建一个项目，命名为“PinTu”，公司域名为“mingrisoft.com”。

（2）导入资源文件：复制“光盘\Code\17\165\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）在“app\res\layout”默认生成的在 activity_main.xml 布局文件中添加拼图布局。

（4）在“app\java\com\mingrisoft\pintu”文件夹下默认生成的 MainActivity 类中创建 Random() 方法，打乱拼图模块顺序。代码如下：

```
01  //打乱顺序
02  private void Random() {
03      int ran1, ran2;
04      for (int i = 0; i < 20; i++) {
05          ran1 = new Random().nextInt(length - 1);
06          do {
07              ran2 = new Random().nextInt(length - 1);
08              if (ran1 != ran2) {
09                  break;
10              }
11          } while (true);
12          Swap(ran1, ran2);
13      }
14      //随机排序
15      ib_00.setImageDrawable(getResources().getDrawable(imageId[imageIndex[0]]));
16      ib_01.setImageDrawable(getResources().getDrawable(imageId[imageIndex[1]]));
```

```
17      ib_02.setImageDrawable(getResources().getDrawable(imageId[imageIndex[2]]));
18      ib_10.setImageDrawable(getResources().getDrawable(imageId[imageIndex[3]]));
19      ib_11.setImageDrawable(getResources().getDrawable(imageId[imageIndex[4]]));
20      ib_12.setImageDrawable(getResources().getDrawable(imageId[imageIndex[5]]));
21      ib_20.setImageDrawable(getResources().getDrawable(imageId[imageIndex[6]]));
22      ib_21.setImageDrawable(getResources().getDrawable(imageId[imageIndex[7]]));
23      ib_22.setImageDrawable(getResources().getDrawable(imageId[imageIndex[8]]));
24  }
```

（5）在 MainActivity 类中创建 move() 方法，用于实现拼图碎片与空白区域，交换位置。代码如下：

```
01  //单击的图片与空白区域的图片交换位置
02  public void move(int imageBtnId, int site) {
03      //相邻图片坐标
04      int sizeX = site / imageX;
05      int sizeY = site % imageY;
06      //空白图片坐标
07      int blackX = blankSwap / imageX;
08      int blackY = blankSwap % imageY;
09      int x = Math.abs(sizeX - blackX);
10      int y = Math.abs(sizeY - blackY);
11      //满足条件
12      if ((x == 0 && y == 1) || (x == 1 && y == 0)) {
13          //单击图片的id
14          ImageButton clickButton = (ImageButton) findViewById(imageBtnId);
15          //当前单击的图片设置为不可见状态
16          clickButton.setVisibility(View.INVISIBLE);
17          //空白图片的ID
18          ImageButton blankButton = (ImageButton) findViewById(blankImgId);
19          //用单击图片覆盖空白图片
20          blankButton.setImageDrawable(getResources().getDrawable(imageId[imageIndex[site]]));
21          //将空白图片的属性设置为可见
22          blankButton.setVisibility(View.VISIBLE);
23          //交换标记
24          Swap(site, blankSwap);
25          //重新计算空白的位置
26          blankSwap = site;
27          //交换两者的ID
28          blankImgId = imageBtnId;
29          GameOver();
30      }
31  }
```

扩展学习

请尝试添加计时功能，计算完成游戏所用时间。

明日科技

实例 166　仿关灯游戏

应用场景：游戏

源码位置：光盘\Code\17\166
视频位置：光盘\Video\17\166

实例说明

关灯游戏是一款简单的益智游戏，十分有趣。本实例通过 SurfaceView+SurfaceHolder+Thread 技术实现关灯游戏。运行实例，效果如图 17.23 所示；单击屏幕上的灯泡，包含该位置及上、下、左、右，点亮的灯泡会变为关闭状态，关闭的灯泡会变成点亮状态，如图 17.24 所示；灯泡全部关闭后，进入下一卡，如图 17.25 所示。

图 17.23　运行效果

图 17.24　游戏中画面

图 17.25　进入下一关画面

关键技术

本实例主要使用 SurfaceHolder 接口类中的 surfaceCreated() 方法、Thread 类中的 Thread() 方法、Canvas 类中的 drawBitmap() 方法、View 类中的 onTouchEvent() 方法，实现关灯效果。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“MyLightsOut”，公司域名为“mingrisoft.com”。

（2）导入资源文件：复制“光盘\Code\17\166\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）在“app\java\com\mingrisoft\mylightsout”文件夹下创建 LightGameView 类，继承 SurfaceView，并实现 SurfaceHolder.Callback 接口，用于实现游戏。代码如下：

```
public class LightGameView extends SurfaceView implements SurfaceHolder.Callback
```

实现 SurfaceHolder.Callback 接口的 surfaceCreated() 方法，用于在创建 SurfaceView 控件时，加载资源开启游戏逻辑子线程。代码如下：

```
01  //创建的时候调用方法
02  @Override
03  public void surfaceCreated(SurfaceHolder holder) {
04      //TODO Auto-generated method stub
05      //加载资源
06      game_over = BitmapFactory.decodeResource(getResources(), R.drawable.bird_gameover);
07      start_btn = BitmapFactory.decodeResource(getResources(), R.drawable.bird_start_btn);
08      light_on = BitmapFactory.decodeResource(getResources(), R.drawable.light_on);
09      light_off = BitmapFactory.decodeResource(getResources(), R.drawable.light_off);
10      //判断初始灯的开关
11      inzy();
12  }
13  private void inzy() {
14      //绘制画笔
15      mPaint.setColor(Color.WHITE);
16      mPaint.setTextSize(50);
17      mPaint.setAntiAlias(true);
18      mapLists = new HashMap<Point, Boolean>();
19      //设置数组记录亮起的等 默认全部关闭
20      for (int j = 0; j < 5; j++) {
21          for (int i = 0; i < 5; i++) {
22              Point p = new Point(i, j);
23              //默认所有灯关闭
24              mapLists.put(p, false);
25          }
26      }
27      //根据关卡设置设置默认亮灯
28      for (int i = 0; i < level+3; i++) {
29          int x = new Random().nextInt(4);
30          int y = new Random().nextInt(4);
31          //判断开启灯
32          mapLists.put(new Point(x, y), true);
33      }
34      isDrawUI = true;
35      //绘制线程
36      threadDrawUI = new ThreadDrawUI();
37      //开启线程
38      threadDrawUI.start();
39      kill = false;
40  }
```

明日科技

创建 checkgameover() 方法，用于检查灯泡是否全部关闭。如果全部关闭，返回 true，否则返回 false。代码如下：

```
01  //循环遍历数组，判断等是否全部关闭
02  public boolean checkgameover() {
03      for (Map.Entry<Point, Boolean> entry : mapLists.entrySet()) {
04          Boolean value = entry.getValue();
05          if (value == true) {
06              return false;
07          }
08      }
09      return true;
10  }
```

（4）在“app\res\layout”目录下默认生成的 activity_main.xml 布局文件中，导入 LightGameView 自定义游戏类。代码如下：

```
01  <?xml version="1.0" encoding="utf-8"?>
02      <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
03          android:layout_width="match_parent"
04          android:layout_height="match_parent"
05          android:id="@+id/close_light_ll"
06          android:background="@drawable/gamebj"
07          android:orientation="vertical">
08      <com.mingrisoft.mylightsout.LightGameView
09          android:layout_marginTop="40dp"
10          android:layout_gravity="center"
11          android:layout_width="match_parent"
12          android:layout_height="match_parent"/>
13  </LinearLayout>
```

扩展学习

请尝试在游戏中加入计时功能，以记录通关所用的时间。

实例 167　连连看游戏

应用场景：游戏

源码位置：光盘\Code\17\167

实例说明

连连看游戏是颇受欢迎的经典游戏之一。本实例带领大家完成 Android 版连连看游戏的开发。运行实例，效果 17.26 所示；单击“开始游戏”按钮，进入游戏页面，如图 17.27 所示；将页面上的内容都消除后，获得胜利，如图 17.28 所示。

图 17.26　运行效果

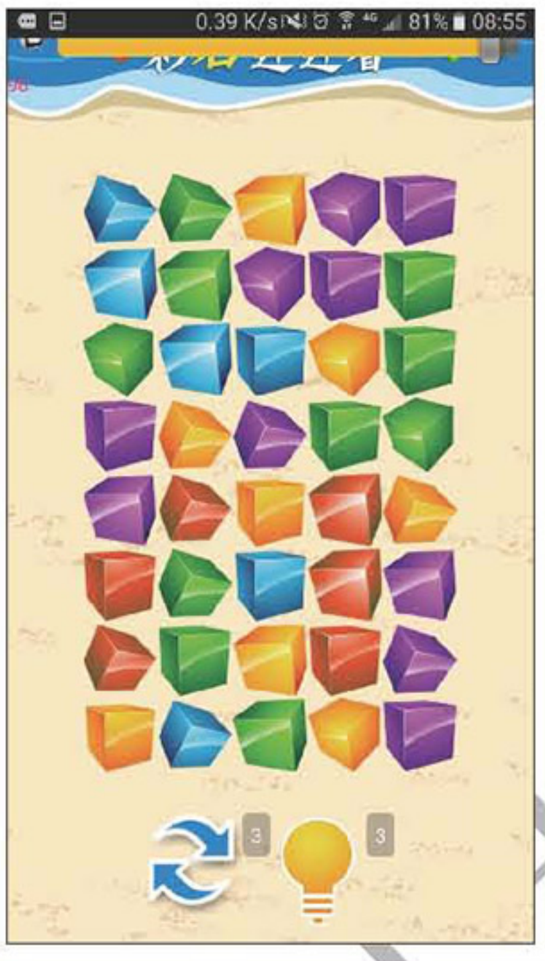


图 17.27　游戏中画面

图 17.28　游戏结束画面

关键技术

关键技术

本实例主要使用 Canvas 类中的 drawBitmap() 方法、drawLine() 方法、MediaPlayer 类中的 create() 方法、SoundPool 类中的 SoundPool() 方法，实现连连看游戏。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“BubbleGame”，公司域名为“mingrisoft.com”。

（2）导入资源文件：复制“光盘\Code\17\167\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）在“app\java\com\mingrisoft\bubblegame”文件夹下创建 BoardView 类继承 View 类，用于实现游戏界面绘制。

（4）在“app\java\com\mingrisoft\bubblegame”文件夹下创建 GameView 类，继承 BoardView 类，用于实现游戏逻辑类。

（5）在“app\java\com\mingrisoft\bubblegame”文件夹下创建 SoundPlay 类，用于管理游戏音效。

（6）在“app\java\com\mingrisoft\bubblegame”文件夹下的 GameView 类中，创建 initSound() 方法，用于初始化音乐资源。代码如下：

```
01  //初始化声音资源
02  public static void initSound(Context context){
03      soundPlay = new SoundPlay();
04      soundPlay.initSounds(context);
05      soundPlay.loadSfx(context, R.raw.choose, ID_SOUND_CHOOSE);
06      soundPlay.loadSfx(context, R.raw.disappear1, ID_SOUND_DISAPEAR);
```

```
07      soundPlay.loadSfx(context, R.raw.win, ID_SOUND_WIN);
08      soundPlay.loadSfx(context, R.raw.lose, ID_SOUND_LOSE);
09      soundPlay.loadSfx(context, R.raw.item1, ID_SOUND_REFRESH);
10      soundPlay.loadSfx(context, R.raw.item2, ID_SOUND_TIP);
11      soundPlay.loadSfx(context, R.raw.alarm, ID_SOUND_ERROR);
12  }
```

（7）在 GameView 类中创建 RefreshTime 线程，用于计时。代码如下：

```
01  //计时线程
02  class RefreshTime extends Thread {
03      public void run() {
04          while (leftTime >= 0 && !isStop) {
05              timerListener.onTimer(leftTime);
06              leftTime--;
07              try {
08                  Thread.sleep(1000);
09              } catch (InterruptedException e) {
10                  e.printStackTrace();
11              }
12          }
13          if(!isStop){
14              setMode(LOSE);
15              soundPlay.play(ID_SOUND_LOSE, 0);
16          }
17      }
18  }
```

扩展学习

尝试显示时间，用于提示用户的当前剩余时间。

实例 168　2048 游戏

应用场景：游戏

源码位置：光盘\Code\17\168

实例说明

2048 游戏是一款风靡一时的游戏，本实例将完成 2048 游戏的开发。运行实例，效果如图 17.29 所示，在数字位置可以上下左右滑动，进行得分，如图 17.30 所示；如果没有位置可移动，则游戏结束，如图 17.31 所示。

明日科技

图 17.29　运行效果

图 17.30　游戏中画面

图 17.31　游戏结束画面

关键技术

本实例主要使用 SharedPreferences 类中的 getBoolean() 方法，Canvas 类的 drawBitmap () 方法，Paint 类中的 setTextSize() 方法、setTypeface() 方法、setAntiAlias() 方法，实现 2048 游戏。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“2048Game”，公司域名为“mingrisoft.com”。

（2）导入资源文件：复制“光盘\Code\17\168\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）在“app\java\com\mingrisoft\2048game”文件夹下创建 MainView 类，用于显示游戏界面类。

（4）在“app\java\com\mingrisoft\2048game”文件夹下默认生成的 MainAcitvity 类中，调用 MainView 类充当布局。代码如下：

```
01  @Override
02  protected void onCreate(Bundle savedInstanceState)
03  {
04      super.onCreate(savedInstanceState);
05      requestWindowFeature(Window.FEATURE_NO_TITLE);
06      view = new MainView(getBaseContext());
07      //保存游戏状态
08      SharedPreferences settings = PreferenceManager.getDefaultSharedPreferences(this);
09      view.hasSaveState = settings.getBoolean("save_state", false);
10      if (savedInstanceState != null)
11      {
12          if (savedInstanceState.getBoolean("hasState"))
13          {
```

```
            //加载游戏状态
            load();
        }
    }
    //装载布局
    setContentView(view);
}
```

（5）在“app\java\com\mingrisoft\2048game”文件夹下的 MainView 类中创建 getCellRectangleIds() 方法用于根据数字设置背景颜色。代码如下：

```
//设置不同数字的背景颜色
private int[] getCellRectangleIds() {
    int[] cellRectangleIds = new int[numCellTypes];
    cellRectangleIds[0] = R.drawable.cell_rectangle;
    cellRectangleIds[1] = R.drawable.cell_rectangle_2;
    cellRectangleIds[2] = R.drawable.cell_rectangle_4;
    cellRectangleIds[3] = R.drawable.cell_rectangle_8;
    cellRectangleIds[4] = R.drawable.cell_rectangle_16;
    cellRectangleIds[5] = R.drawable.cell_rectangle_32;
    cellRectangleIds[6] = R.drawable.cell_rectangle_64;
    cellRectangleIds[7] = R.drawable.cell_rectangle_128;
    cellRectangleIds[8] = R.drawable.cell_rectangle_256;
    cellRectangleIds[9] = R.drawable.cell_rectangle_512;
    cellRectangleIds[10] = R.drawable.cell_rectangle_1024;
    cellRectangleIds[11] = R.drawable.cell_rectangle_2048;
    for (int xx = 12; xx < cellRectangleIds.length; xx++) {
        cellRectangleIds[xx] = R.drawable.cell_rectangle_4096;
    }
    return cellRectangleIds;
}
```

扩展学习

当前游戏随机生成 2~4 个方块，尝试每次都随机位置生成 4 个方块。

实例 169　抓住大公鸡游戏

应用场景：游戏

源码位置：光盘\Code\17\169

实例说明

围住神经猫游戏是一款十分火爆的游戏，本实例将模仿此游戏完成抓住大公鸡游戏的开发。运行实例，效果如图 17.32 所示；单击屏幕上的灰色小点，小点变亮，如图 17.33 所示；如果公鸡没有灰色位置可移动则被围住，游戏胜利，如图 17.34 所示。

图 17.32　运行效果

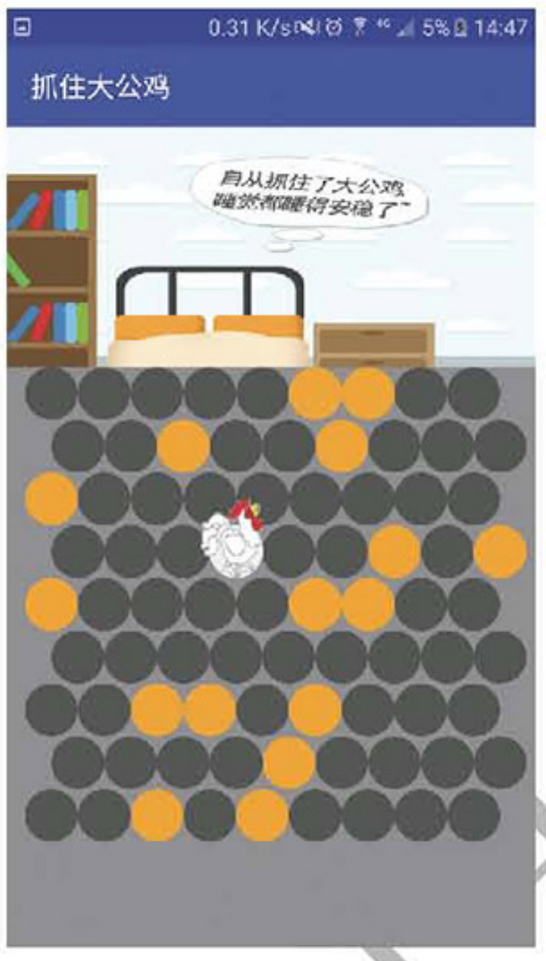


图 17.33　游戏中画面

图 17.34　游戏结束画面

关键技术

本实例主要使用 Activity 类中的 requestWindowFeature() 方法，Resources 类中的 getDrawable () 方法，AlertDialog 类中的 setTitle() 方法、setMessage() 方法、setPositiveButton () 方法，实现该游戏。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“CrazyCat”，公司域名为“mingrisoft.com”。

（2）导入资源文件：复制“光盘\Code\17\169\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）在“app\java\com\mingrisoft\CrazyCat”文件夹下默认生成的 MainAcitvity 类中，为“开始游戏”按钮添加单击事件。代码如下：

```
01  //开始按钮单击事件
02  public void onClick(View view) {
03      //跳转到游戏页
04      Intent intent = new Intent(MainActivity.this, PlayActivity.class);
05      startActivity(intent);
06  }
```

（4）在“app\java\com\mingrisoft\2048game”文件夹下创建 PlayGround 类，用于游戏页面的主逻辑，然后在其中创建 win() 方法，用于通过成功后显示弹窗。代码如下：

```
01  //通关成功
02  private void win() {
```

```
03      Builder dialog = new Builder(context);
04      dialog.setTitle("通关成功");
05      dialog.setMessage("你用" + (steps + 1) + "步捕抓到了大公鸡。");
06      dialog.setCancelable(false);
07      dialog.setNegativeButton("再玩一次", new DialogInterface.OnClickListener() {
08          public void onClick(DialogInterface dialog, int which) {
09              initGame();
10          }
11      });
12      dialog.setPositiveButton("取消", null);
13      dialog.show();
14  }
```

扩展学习

尝试修改地图大小，从而可以能更容易抓住大公鸡。

实例 170　仿飞机大战

应用场景：游戏

源码位置：光盘\Code\17\170

实例说明

本实例将完成仿飞机大战游戏的开发。运行实例，效果如图 17.35 所示；滑动屏幕，飞机跟随自己的手指移动，如图 17.36 所示；如果自己的飞机碰撞到飞机，游戏结束，如图 17.37 所示。

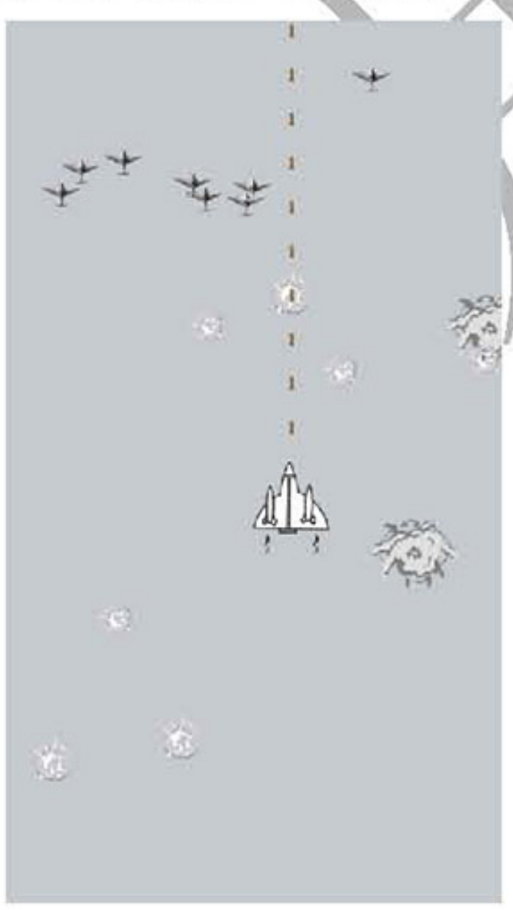


图 17.35　运行效果

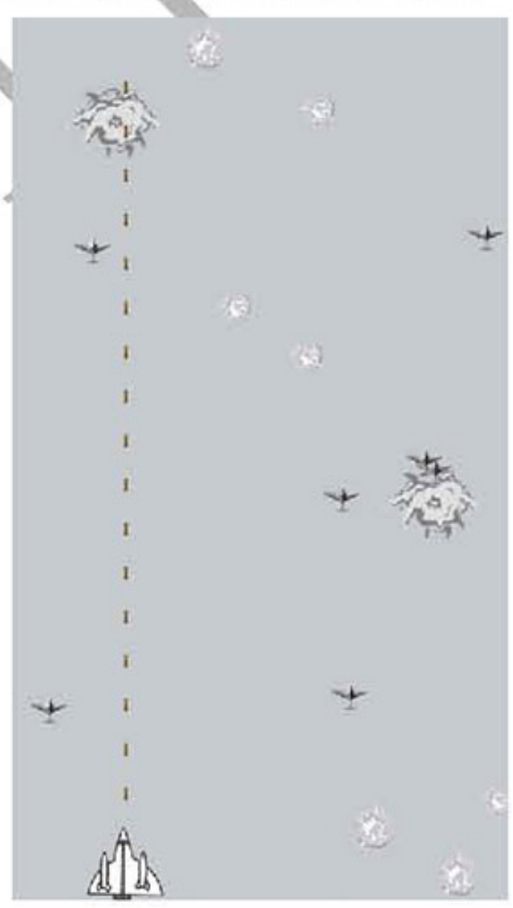


图 17.36　随手指移动后画面

图 17.37　游戏结束画面

关键技术

本实例主要使用 BitmapFactory 类中的 decodeResource() 方法、Paint 类中的 setColor () 方法、Random 类中的 nextInt() 方法、Thread 类中的 sleep() 方法、Canvas 类中的 drawBitmap () 方法，实现了仿飞机大战游戏。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“PlaneFighter”，公司域名为“mingrisoft.com”。

（2）导入资源文件：复制“光盘\Code\17\170\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）在“app\java\com\mingrisoft\planefigther\UI”文件夹下创建 AirSurfaceView 类，用于实现游戏页面的主逻辑以及页面绘制。

（4）在“app\java\com\mingrisoft\planefigther”文件夹下创建 model 文件夹，存放精灵实体类，如飞机、敌方飞机和子弹等。

（5）在“app\java\com\mingrisoft\planefigther”文件夹下默认生成的 MainAcitvity 类中，创建 initView() 方法，初始化游戏布局。代码如下：

```
01  private void initView() {
02      //设置满屏显示
03      FrameLayout.LayoutParams full = new FrameLayout.LayoutParams(
04              ViewGroup.LayoutParams.MATCH_PARENT,
05              ViewGroup.LayoutParams.MATCH_PARENT);
06      //设置在屏幕中间位置显示
07      FrameLayout.LayoutParams center =
08      new FrameLayout.LayoutParams(FrameLayout.LayoutParams.WRAP_CONTENT,
09      FrameLayout.LayoutParams.WRAP_CONTENT);
10      //新的游戏布局
11      rootView = new FrameLayout(this);
12      //布局背景
13      rootView.setBackgroundColor(Color.WHITE);
14      //初始化游戏页面
15      mainView = new AirSurfaceView(this);
16      //添加游戏页面到布局全屏显示
17      rootView.addView(mainView, full);
18      //中间位置显示
19      center.gravity = Gravity.CENTER;
20      //创建按钮
21      restart = new Button(this);
22      //设置按钮文字
23      restart.setText(R.string.restart_game);
24      //设置重新开始游戏按钮字号
25      restart.setTextSize(getResources().getDimension(R.dimen.restart_btn_text_size));
```

```
26        //隐藏游戏按钮
27        restart.setVisibility(View.GONE);
28        //绑定按钮单击事件
29        restart.setOnClickListener(this);
30        //添加按钮到布局中间位置
31        rootView.addView(restart, center);
32        //设置新建布局为当前页面的布局
33        setContentView(rootView);
34    }
```

扩展学习

请尝试调节子弹速度，加快子弹的射击速度。

明日科技

第18章

5.0+效果（Material Design）

实例 171　仿小米联系人折叠动画效果

应用场景：界面效果

源码位置：光盘\Code\18\171

实例说明

本实例将实现仿小米联系人折叠动画效果。运行实例，将显示如图 18.1 所示的默认效果；当向上滑动后会出现如图 18.2 所示的效果。在向上滑动的过程中标题栏中的字体会逐渐缩小，背景图逐渐被遮挡。而向下滑动时标题栏中的字体会逐渐放大，背景图会逐渐显示。

图 18.1　初始效果

图 18.2　界面收缩的效果

关键技术

本实例主要使用 AppBarLayout 方法和 CollapsingToolbarLayout 方法，实现仿小米联系人折叠的动画效果。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“CollapsingToolbarLayout”，公司域名为“mingrisoft. com”。

（2）在 activity_main.xml 文件中将 CoordinatorLayout 布局设置为布局文件中的根布局，该布局是用来协调子控件行为的，也就是通过设置子控件的行为来实现一些动画效果。布局的代码如下：

代码位置：CollapsingToolbarLayout\app\src\main\res\layout\activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<!-- CoordinatorLayout:协调布局，协调子控件的行为 -->
<!-- android:fitsSystemWindows：设为true，适应系统窗口 -->
<android.support.design.widget.CoordinatorLayout
    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:layout_width="match_parent"
    android:layout_height="match_parent"
    android:fitsSystemWindows="true"
    tools:context="com.mrkj.sample195.ScrollingActivity">
    <!-- AppBarLayout：相当于竖直排列的线性布局 -->
    <!-- 具有滑动手势的行为，但需要在以CoordinatorLayout -->
    <!-- 为父布局的布局中使用 -->
    <android.support.design.widget.AppBarLayout
        android:id="@+id/app_bar"
        android:fitsSystemWindows="true"
        android:layout_height="wrap_content"
        android:layout_width="match_parent"
        android:theme="@style/AppTheme.AppBarOverlay">
        <!-- CollapsingToolbarLayout：用来包装ToolBar来使用它 -->
        <!-- 实现折叠效果，通常用作AppBarLayout的子布局 -->
        <android.support.design.widget.CollapsingToolbarLayout
            android:id="@+id/toolbar_layout"
            android:fitsSystemWindows="true"
            android:layout_width="match_parent"
            android:layout_height="match_parent"
            app:layout_scrollFlags="scroll|exitUntilCollapsed"
            app:contentScrim="?attr/colorPrimary">
            <!-- 背景图片 -->
            <ImageView
                android:layout_width="match_parent"
```

明日科技

```
33                android:layout_height="300dp"
34                app:layout_collapseParallaxMultiplier="0.5"
35                android:fitsSystemWindows="true"
36                app:layout_collapseMode="parallax"
37                android:scaleType="fitXY"
38                android:src="@mipmap/mrkj_head"/>
39            <!-- ToolBar用来替代ActionBar -->
40            <android.support.v7.widget.Toolbar
41                android:id="@+id/toolbar"
42                android:layout_height="?attr/actionBarSize"
43                android:layout_width="match_parent"
44                app:layout_collapseMode="pin"
45                app:popupTheme="@style/AppTheme.PopupOverlay" />
46        </android.support.design.widget.CollapsingToolbarLayout>
47    </android.support.design.widget.AppBarLayout>
48    <include layout="@layout/content_scrolling" />
49 </android.support.design.widget.CoordinatorLayout>
```

（3）在 MainActivity 类中实现替换自带的 ActionBar。代码如下：

代码位置：CollapsingToolbarLayout\app\src\main\java\com\mingrisoft\collapsingtoolbarLayout\MainActivity.java

```
01 @Override
02 protected void onCreate(Bundle savedInstanceState) {
03     super.onCreate(savedInstanceState);
04     setContentView(R.layout.activity_scrolling);
05     Toolbar toolbar = (Toolbar) findViewById(R.id.toolbar);
06     toolbar.setTitle("明日科技");
07     setSupportActionBar(toolbar);
08 }
```

扩展学习

请尝试更改布局中的 collapseMode 属性，来改变界面显示的效果。

实例 172　主题切换

应用场景：界面

源码位置：光盘\Code\18\172

实例说明

本实例将实现主题切换的动画效果。运行实例，将显示如图 18.3 所示的界面；单击右侧弹出的菜单项，选择切换主题，切换后界面中的主题将会改变，如图 18.4 所示。

明日科技

图 18.3　绿色主题效果

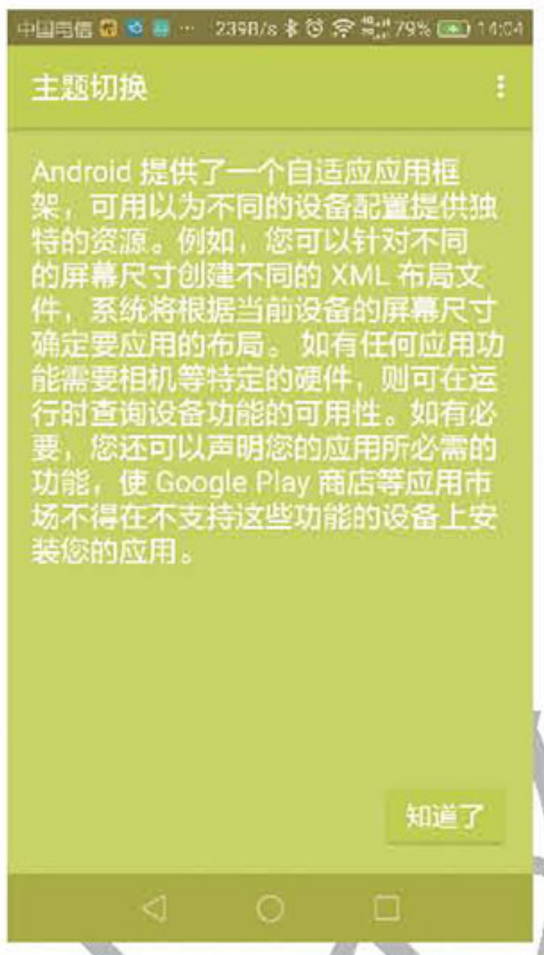


图 18.4　黄色主题效果

关键技术

关键技术

本实例主要使用 Activity 类中的 setTheme () 方法设置主题，使用 recreate() 方法重启 Activity 界面。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“Theme”，公司域名为“mingrisoft. com”。

（2）导入资源文件：复制“光盘\Code\18\172\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）在 color 资源文件中设置了一套主题颜色，本实例以绿色主题为例。代码如下：

代码位置：Theme\app\src\main\res\values\color.xml

```
<color name="status_blue">#2a36b1</color>          <!-- 900 -->
<color name="action_blue">#5677fc</color>          <!-- 500 -->
<color name="window_bg_blue">#91a7ff</color>       <!-- 300 -->
<color name="navigation_blue">#455ede</color>      <!-- 700 -->
<color name="bg_blue">#738ffe</color>      <!-- 400 -->
<color name="fg_blue">#e7e9fd</color>      <!-- 50 -->
<color name="control_normal_blue">#e7e9fd</color>      <!-- 50 -->
<color name="control_highlight_blue">#afbfff</color>      <!-- 200 -->
<color name="control_activated_blue">#4d69ff</color>      <!-- A700 -->
<color name="button_normal_blue">#738ffe</color> <!-- 400 -->
```

（4）在 styles 资源文件中设置主题样式。代码如下：

代码位置：Theme\app\src\main\res\values-v21\styles.xml

```
<style name="Blue" parent="android:Theme.Material">
    <item name="android:colorPrimaryDark">@color/status_blue</item>
    <item name="android:statusBarColor">@color/status_blue</item>
    <item name="android:colorPrimary">@color/action_blue</item>
    <item name="android:windowBackground">@color/window_bg_blue</item>
    <item name="android:navigationBarColor">@color/navigation_blue</item>
    <item name="android:colorForeground">@color/fg_blue</item>
    <item name="android:colorBackground">@color/bg_blue</item>
    <item name="android:colorAccent">@color/control_activated_blue</item>
    <item name="android:colorControlNormal">@color/control_normal_blue</item>
    <item name="android:colorControlHighlight">@color/control_highlight_blue</item>
    <item name="android:colorControlActivated">@color/control_activated_blue</item>
    <item name="android:colorButtonNormal">@color/button_normal_blue</item>
    <item name="android:textColor">@color/white_text</item>
    <item name="android:textColorPrimaryDisableOnly">@color/white_text</item>
    <item name="android:textColorPrimary">@color/white_text</item>
</style>
```

（5）在 MainActivity 类中实现切换主题功能。代码如下：

代码位置：Theme\app\src\main\java\com\mingrisoft\theme\MainActivity.java

```
public class MainActivity extends Activity {
    private static int theme = R.style.Blue;                    //设置默认主题
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setTheme(theme);                                        //设置主题
        setContentView(R.layout.activity_main);
    }
    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        getMenuInflater().inflate(R.menu.change,menu);
        return true;
    }
    /**
     * 切换主题
     * @param item
     * @return
     */
    @Override
    public boolean onOptionsItemSelected(MenuItem item) {
        switch (item.getItemId()){
            case R.id.blue:                                     //设置为蓝色主题
                theme = R.style.Blue;
                break;
            case R.id.gray:                                     //设置为黑色主题
                theme = R.style.Black;
```

```
                break;
            case R.id.green:                                    //设置为绿色主题
                theme = R.style.Green;
                break;
            case R.id.yellow:                                   //设置为黄色主题
                theme = R.style.Yellow;
                break;
        }
        recreate();                                             //重启界面
        return super.onOptionsItemSelected(item);
    }
}
```

扩展学习

请尝试参照 color.xml 文件和 styles.xml 文件来设计一套主题效果。

实例 173　切换多个选项卡

应用场景：ViewPager 的切换指示

源码位置：光盘\Code\18\173

实例说明

ViewPager 指示器在新闻客户端中很常见，Google 提供的 design 支持包中有一个叫作 TabLayout 的控件，在项目中只需要将 ViewPager 与 TabLayout 绑定，就可以实现在多个选项卡之间进行切换的功能。运行实例，效果如图 18.5 所示；单击任意一个选项卡可以切换到相应的页面，如图 18.6 所示。

图 18.5　界面默认效果

图 18.6　切换后的效果

关键技术

本实例主要使用 TabLayout 类中的 setupWithViewPager () 方法将 ViewPager 与 TabLayout 绑定起来，从而实现多个选项卡切换的效果。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“TabLayout”，公司域名为“mingrisoft. com”。

（2）导入资源文件：复制“光盘\Code\18\173\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）在 MainActivity 类中创建内部类 MyAdapter，用来为 ViewPager 添加展示的数据。代码如下：

代码位置：TabLayout\app\src\main\java\com\mingrisoft\tabLayout\MainActivity.java

```
/**
 * ViewPage适配器
 */
class MyAdapter extends PagerAdapter{
    @Override
    public int getCount() {
        return viewList.size();
    }
    @Override
    public boolean isViewFromObject(View view, Object object) {
        return view == object;
    }
    /**
     * 加载item
     * @param container
     * @param position
     * @return
     */
    @Override
    public Object instantiateItem(ViewGroup container, int position) {
        View view  = viewList.get(position);
        container.addView(view);                                    //添加
        return view;
    }
    /**
     * 销毁Item
     * @param container
     * @param position
     * @param object
     */
    @Override
    public void destroyItem(ViewGroup container, int position, Object object) {
        container.removeView(viewList.get(position));               //移除
```

明日科技

```
    }
    /**
     * 关联指示器
     * @param position
     * @return
     */
    @Override
    public CharSequence getPageTitle(int position) {
        return titles[position];
    }
}
```

在 onCreate() 方法中初始化控件，并为 ViewPager 绑定适配器，再将 ViewPager 与 TabLayout 进行绑定。代码如下：

```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
    tabLayout = (TabLayout) findViewById(R.id.tab);
    viewPager = (ViewPager) findViewById(R.id.page);
    viewList = new ArrayList<>();                                    //实例化集合
    for (int i = 0; i < titles.length; i++) {
        //加载布局并填充成控件
        View view = getLayoutInflater().inflate(R.layout.item,null);
        view.setBackgroundResource(redIDs[i]);                       //设置背景
        viewList.add(view);                                          //添加布局
    }
    viewPager.setAdapter(new MyAdapter());                           //绑定适配器
    tabLayout.setupWithViewPager(viewPager,true);                    //绑定ViewPager
}
```

扩展学习

请尝试在布局文件更改 TabLayout 的选项的字体颜色。

实例 174　浮动按钮和消息显示

应用场景：浮动按钮、消息提示

源码位置：光盘\Code\18\174

实例说明

在 Android 的 design 支持包中有一个浮动按钮和一个 SnackBar 提示条，本实例将展示它们的使用方式。运行实例，效果如图 18.7 所示；当单击按钮时，会在屏幕的底部弹出 SnackBar 的效果，如图 18.8 所示。

图 18.7　界面的默认效果

图 18.8　消息提示

关键技术

本实例主要使用 SnackBar 类中的 make() 方法创建一个 SnackBar 对象，并通过 show() 方法将 SnackBar 显示出来，另外，还使用 setAction() 方法设置 SnackBar 按钮的单击事件。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“SnackBar”，公司域名为“mingrisoft. com”。

（2）设置 activity.xml 布局文件，根布局设置为 CoordinatorLayout，然后添加一个 ImageView 和一个 FloatingActionButton 悬浮按钮。代码如下：

代码位置：SnakBar\app\src\main\res\layout\activity.xml

```
01 <?xml version="1.0" encoding="utf-8"?>
02 <android.support.design.widget.CoordinatorLayout
03 xmlns:android="http://schemas.android.com/apk/res/android"
04     xmlns:app="http://schemas.android.com/apk/res-auto"
05     xmlns:tools="http://schemas.android.com/tools"
06     android:layout_width="match_parent"
07     android:layout_height="match_parent"
08     tools:context="com.mrkj.sample_196.MainActivity">
09     <!-- 背景图片 -->
10     <ImageView
11         android:layout_width="match_parent"
```

```
        android:layout_height="match_parent"
        android:scaleType="centerCrop"
        android:src="@mipmap/mrkj_background"
        />
    <!-- 浮动按钮 -->
    <android.support.design.widget.FloatingActionButton
        android:id="@+id/fab"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_margin="@dimen/fab_margin"
        android:layout_gravity="right|bottom"
        android:src="@drawable/mrkj_btn"
        app:backgroundTint="#DCE7DD"
        app:rippleColor="#DCE7DD"
        app:borderWidth="0dp"
        />
</android.support.design.widget.CoordinatorLayout>
```

（3）在 MainActivity 类的 onCreate() 方法中，初始化 FloatingActionButton 悬浮按钮，并为 FloatingActionButton 按钮添加单击事件，在单击事件中添加 SnackBar 的显示设置。代码如下：

```
//实例化控件
FloatingActionButton fab = (FloatingActionButton) findViewById(R.id.fab);
fab.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {
        //显示Snackbar
        Snackbar snackbar = Snackbar.make(view, "提示", Snackbar.LENGTH_LONG)
                .setAction("知道了", new View.OnClickListener() {
                    @Override
                    public void onClick(View v) {
                    }
                });
        snackbar.show();                    //显示SnackBar
    }
});
```

扩展学习

尝试修改 SnackBar 的背景颜色和字体颜色。

实例 175	过场动画 应用场景：界面切换效果	源码位置：光盘\Code\18\175

实例说明

本实例将实现在浏览照片时添加过场动画效果的功能。运行实例，将显示如图 18.9 所示的界面；当单击某张图片后，会跳转到如图 18.10 所示的界面。在跳转界面的过程中将会显示切换界面的动画，同时在退出显示图片时同样会展示一段动画。本实例使用了共享元素界面过渡动画。

图 18.9　图片的列表

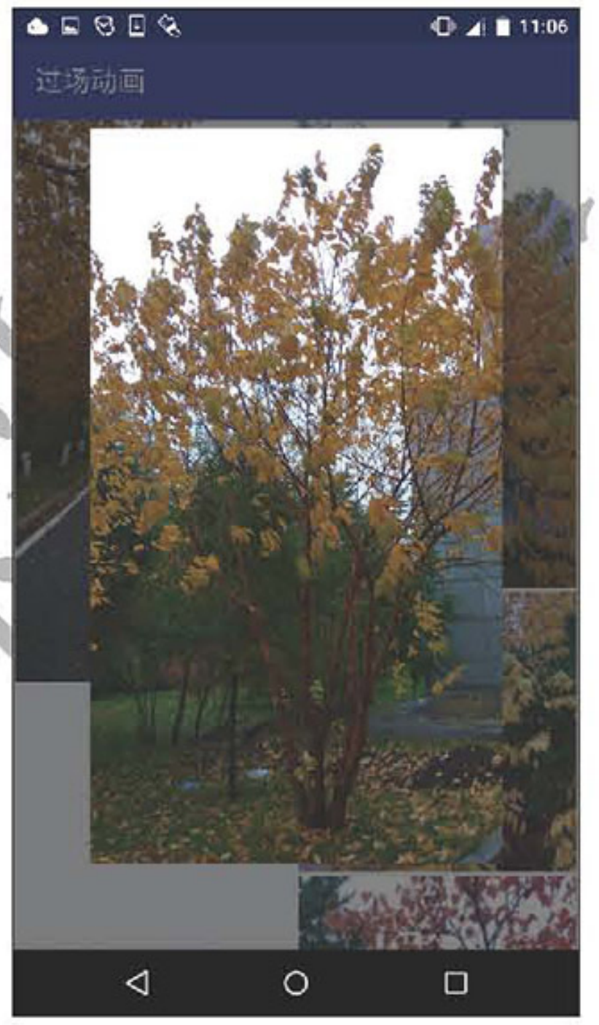


图 18.10　显示图片界面

关键技术

本实例实现时主要使用 TransitionSet 类中的 addTransition() 方法（用来设置界面的过渡效果）和 setDuration() 方法（用来设置时间），TransitionSet 类本身也是用来设置过渡效果的。另外，这里使用了 ActivityOptions.makeSceneTransitionAnimation() 方法来设置共享元素动画。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“Scene”，公司域名为“mingrisoft. com”。

（2）导入资源文件：复制“光盘\Code\18\175\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）在 MainActivity 类的 onCreate() 方法中，初始化主界面中的数据和控件，将资源文件中的图片获取出来，并给它们随机设置要显示的高度。代码如下：

代码位置：Scene\app\src\main\java\com\mingrisoft\scene\MainActivity.java

```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
    showList = (RecyclerView) findViewById(R.id.list);
    //获取屏幕的宽度
    DisplayMetrics dm = getResources().getDisplayMetrics();
    width = dm.widthPixels;
    //获取图片
    for (int i = 1; i <= 15; i++) {
        bitmaps.add(getRes("jpg" + i));
    }
    //随机给图片设置高度
    for (int i = 0; i < 15; i++) {
        height.add(getHeight());
    }
    //创建适配器
    MyAdapter adapter = new MyAdapter(this,bitmaps,height);
    //设置item的单击事件
    adapter.setOnItemClickListener(this);
    //设置列表的展示效果
    showList.setLayoutManager(new StaggeredGridLayoutManager(2,
                              GridLayoutManager.VERTICAL));
    //绑定适配器
    showList.setAdapter(adapter);
}
```

设置列表的 Item 单击事件，在 Item 单击事件中来实现跳转界面的功能。当设备版本大于或等于版本号 21，进行带有过渡动画效果的界面，否则进行普通的界面跳转。代码如下：

```
/**
 * Item单击事件
 * @param view
 * @param position
 */
@Override
public void onItemClick(ImageView view, int position) {
    //做一个版本的判断
    if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.LOLLIPOP) {
        //创建用于实现过渡动画效果的对象
        TransitionSet transitionSet = new TransitionSet();
        //添加过渡效果
        transitionSet.addTransition(new ChangeBounds());
        //设置动画时长为1000毫秒
```

明日科技

```
        transitionSet.setDuration(1000);
        //设置将要跳转到界面的进入动画
        getWindow().setSharedElementEnterTransition(transitionSet);
        //设置当前界面的退出动画
        getWindow().setSharedElementExitTransition(transitionSet);
        //设置将要跳转到的界面的退出动画
        getWindow().setSharedElementReturnTransition(transitionSet);
        //设置重返当前界面的进入动画
        getWindow().setSharedElementReenterTransition(transitionSet);
        //设置跳转界面的意图
        Intent intent = new Intent(this, ShareActivity.class);
        //传值
        intent.putExtra("position",position + 1);
        //设置共享元素动画
        ActivityOptions options = ActivityOptions.makeSceneTransitionAnimation(this,
                                          view, "pic");
        //启动跳转界面
        startActivity(intent, options.toBundle());
    }else{                                          //API小于21时执行此处跳转界面
        startActivity(new Intent().putExtra("position",position + 1));
    }
}
```

（4）在 ShareActivity 界面中添加过渡效果的设置，内容跟在主界面里的设置基本一样。然后添加获取图片并展示的功能。代码如下：

```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.LOLLIPOP){
        TransitionSet transitionSet = new TransitionSet();
        transitionSet.addTransition(new ChangeBounds());
        transitionSet.addTarget("pic");
        transitionSet.setDuration(1000);
        //设置跳转界面进入动画
        getWindow().setSharedElementEnterTransition(transitionSet);
        //设置当前界面的退出动画
        getWindow().setSharedElementExitTransition(transitionSet);
        //设置跳转界面的退出动画
        getWindow().setSharedElementReturnTransition(transitionSet);
        //设置重返当前界面的进入动画
        getWindow().setSharedElementReenterTransition(transitionSet);
    }
    setContentView(R.layout.activity_share);
    Bitmap bitmap =getRes("jpg"+getIntent().getIntExtra("position",0));
```

```
20        //展示图片
21        ImageView iv = (ImageView) findViewById(R.id.pic);
22        iv.setImageBitmap(bitmap);
23    }
```

扩展学习

在展示图片的界面中增加单击屏幕退出界面的功能。

实例 176　切换列表效果

应用场景：数据的列表展示

源码位置：光盘\Code\18\176

实例说明

本实例将通过两种方法展示列表效果，并实现两种列表效果进行切换。运行实例，如图 18.11 所示的界面为 ListView 实现的 UI 布局效果；单击“横向列表效果”按钮，将显示如图 18.12 所示的横向列表效果，本实例使用 GridView 类来实现。在本实例中，只需要使用 RecyclerView 控件就可以实现这两种效果。

图 18.11　列表效果图

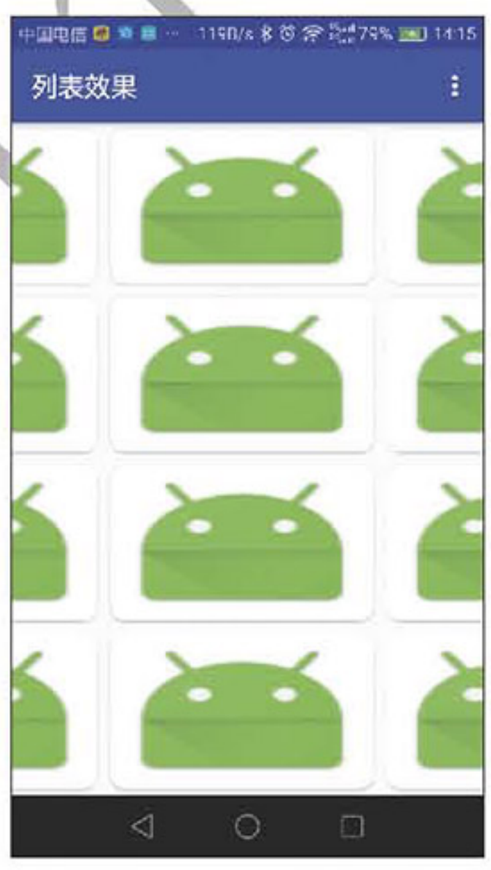


图 18.12　横向列表效果

关键技术

关键技术

本实例主要使用 RecyclerView 类中的 setAdapter() 方法设置适配器，使用 setLayoutManger() 方法设置 RecyclerView 的展示效果。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“List”，公司域名为“mingrisoft. com”。

（2）导入资源文件：复制“光盘\Code\18\176\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）创建 RecyclerView 的适配器，该适配器继承 RecyclerView.Adapter 类。具体实现代码如下：

代码位置：List\app\src\main\java\com\mingrisoft\List\adapter\MyAdapter.java

```
01  public class MyAdapter extends RecyclerView.Adapter<MyAdapter.MyHolder>{
02
03      private LayoutInflater inflater;                    //布局填充器
04      private List<String> list;                          //数据集合
05      private Context context;                            //上下文对象
06
07      /**
08       * 构造方法
09       * @param list
10       * @param context
11       */
12      public MyAdapter(List<String> list, Context context) {
13          this.list = list;
14          this.context = context;
15          this.inflater = LayoutInflater.from(context);
16      }
17
18      /**
19       * 创建item
20       * @param parent
21       * @param viewType
22       * @return
23       */
24      @Override
25      public MyHolder onCreateViewHolder(ViewGroup parent, int viewType) {
26          //填充布局
27          View view = inflater.inflate(R.layout.list_item,parent,false);
28          //创建复用类
29          MyHolder holder = new MyHolder(view);
30          return holder;
31      }
32
33      /**
34       * 绑定控件
35       * @param holder
36       * @param position
37       */
```

明日科技

```
38      @Override
39      public void onBindViewHolder(MyHolder holder, int position) {
40          //下载图片
41          Glide.with(context)
42                  .load(list.get(position))                          //加载图片
43                  .placeholder(R.mipmap.ic_launcher)                 //加载时的图片
44                  .error(R.mipmap.ic_launcher)                       //加载错误的图片
45                  .fitCenter()
46                  .centerCrop()                                      //图片显示方式
47                  .transform(new GlideRoundTransform(context))       //设置边框
48                  .into(holder.pic);                                 //加载到的控件
49      }
50
51      /**
52       * 获取数据条数
53       * @return
54       */
55      @Override
56      public int getItemCount() {
57          return list.size();
58      }
59
60      /**
61       * 复用类
62       */
63      class MyHolder extends RecyclerView.ViewHolder{
64          ImageView pic;
65          public MyHolder(View itemView) {
66              super(itemView);
67              pic = (ImageView) itemView.findViewById(R.id.food);
68          }
69      }
70  }
```

（4）在 MainActivity 类的 onCreate() 方法中，初始化控件并绑定数据。代码如下：

代码位置：List\app\src\main\java\com\mingrisoft\list\MainActivity.java

```
01  @Override
02  protected void onCreate(Bundle savedInstanceState) {
03      super.onCreate(savedInstanceState);
04      setContentView(R.layout.activity_main);
05      recyclerView = (RecyclerView) findViewById(R.id.list_show);
06      //创建适配器
07      MyAdapter adapter = new MyAdapter(initData(), this);
08      recyclerView.setAdapter(adapter);                              //绑定适配器
```

明日科技

```
09      //设置recyclerView的展示效果
10      recyclerView.setLayoutManager(new LinearLayoutManager(this));
11  }
```

在 MainActivity 类中重写 onOptionsItemSelected() 方法，用来切换列表的效果，这里通过设置 setLayoutManager() 方法来改变 RecyclerView 的展示效果。代码如下：

```
01  @Override
02  public boolean onOptionsItemSelected(MenuItem item) {
03      switch (item.getItemId()){
04          case R.id.vertical:                                    //纵向列表
05              recyclerView.setLayoutManager(new LinearLayoutManager(this));
06              break;
07          case R.id.horizontal:                                  //横向列表
08              recyclerView.setLayoutManager(newStaggeredGridLayoutManager(4,
09                      LinearLayoutManager.HORIZONTAL));
10              break;
11          case R.id.grid:                                        //表格列表
12              recyclerView.setLayoutManager(new GridLayoutManager(this,2));
13              break;
14      }
15      return super.onOptionsItemSelected(item);
16  }
```

扩展学习

本实例并没有给 RecyclerView 类设置 Item 的单击事件，请尝试给 RecyclerView 类添加 Item 单击事件。

实例 177　弹出底部弹窗

应用场景：弹窗

源码位置：光盘\Code\18\177

实例说明

在一些 App 应用中，会从底部弹出一个窗口，有“拍照”“从相册中选择”等菜单项，单击任一菜单项，执行相关的操作。本实例将实现弹出底部弹窗的效果。运行实例，如图 18.13 所示；单击右上角按钮，将显示弹窗弹出效果的界面，如图 18.14 所示。

关键技术

本实例主要使用 BottomSheetDialog 类中的 setContentView() 方法加载弹窗的布局，使用 show() 方法显示弹窗，使用 dismiss() 方法关闭弹窗。

图 18.13 界面效果

图 18.14 弹窗弹出效果

实现过程

（1）打开 Android Studio，新建一个项目，命名为“BottomDialog”，公司域名为“mingrisoft.com”。

（2）导入资源文件：复制“光盘\Code\18\177\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）在 MainActivity 类中添加 show() 方法，在 show() 方法中将要加载到 BottomSheetDialog 中的布局填充成控件，然后加载到 BottomSheetDialog 中，并设置 BottomSheetDialog 的状态栏为半透明效果，然后显示弹窗。代码如下：

```
01  //加载弹窗视图布局
02  View bottom = getLayoutInflater().inflate(R.layout.bottom_layout, null);
03  //创建弹窗对象
04  BottomSheetDialog dialog = new BottomSheetDialog(this);
05  dialog.setContentView(bottom);                          //设置弹窗布局
06  //设置弹窗的状态栏半透明
07  dialog.getWindow().addFlags(WindowManager.LayoutParams.FLAG_TRANSLUCENT_STATUS);
08  //显示弹窗
09  dialog.show();
```

在 MainActivity 类中创建内部类 MineOnClick，并实现 OnClickListener 接口，用来为弹窗中的控件添加单击事件，当单击弹窗中的选项时关闭弹窗。代码如下：

```
01  /**
02   * 单击事件
03   */
04  class MineOnClick implements View.OnClickListener{
```

```
05      private BottomSheetDialog dialog;
06      public MineOnClick(BottomSheetDialog dialog) {
07          this.dialog = dialog;
08      }
09      @Override
10      public void onClick(View v) {
11           dialog.dismiss();                          //关闭弹窗
12      }
13  }
```

扩展学习

请尝试为弹窗添加自定义的样式，改变展示弹窗后的窗口颜色。

实例 178	根据图片调节风格 应用场景：界面效果	源码位置：光盘\Code\18\178

实例说明

本实例将实现滑动列表功能，状态栏、标题栏和底部功能栏的颜色会随着列表中显示的第一张图片的颜色而改变。运行实例，将显示如图 18.15 的效果，当前界面的风格是绿色的；当向上滑动后会呈现图 18.16 所示的效果，状态栏、标题栏和底部功能栏都变成了深红色。

图 18.15　界面默认效果

图 18.16　向上滑动后的效果

关键技术

本实例主要使用 Palette 类中的 Palette.from().generate() 方法实现异步检测图片的色彩，使用 getVibrantSwatch() 方法获取图片中最具有活力的颜色。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“Palette”，公司域名为“mingrisoft. com”。

（2）导入资源文件：复制“光盘\Code\18\178\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）在 MainActivity 类的 onCreate() 方法中，初始化图片资源和列表控件，并设置界面的默认效果。代码如下：

代码位置：Palette\app\src\main\java\com\mingrisoft\palette\MainActivity.java

```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
    toolbar= (Toolbar) findViewById(R.id.toolbar);
    setSupportActionBar(toolbar);                          //添加ToolBar替代ActionBar
    bitmaps = new ArrayList<>();                           //图片集合
    showList = (ListView) findViewById(R.id.list);
    for (int i = 1; i <= 12; i++) {
        bitmaps.add(getRes("image" + i));                  //获取资源中的图片
    }
    if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.LOLLIPOP) {
        window = getWindow();                              //获取窗体对象
    }
    showList.setAdapter(new MyAdapter());                  //设置适配器
    //监听列表的滑动事件
    showList.setOnScrollListener(new MyScrollListener());
    //初始化主题效果
    changeToolBarWithStatusBarColor(0);
}
```

在 MainActivity 类中创建 changeToolBarWithStatusBarColor() 方法，用来检测出最具活力颜色，并设置给标题栏。代码如下：

```
//异步检测图片的颜色
Palette.from(bitmaps.get(position)).generate(new Palette.PaletteAsyncListener() {
    @TargetApi(Build.VERSION_CODES.LOLLIPOP)
    @Override
    public void onGenerated(Palette palette) {
        //获取到充满活力的这种色调
        Palette.Swatch vibrantSwatch = palette.getVibrantSwatch();
        if (vibrantSwatch != null){
```

```
09              //设置ToolBar的背景颜色
10              toolbar.setBackgroundColor(vibrantSwatch.getRgb());
11              if (window != null){
12                  //设置状态栏的颜色
13                  window.setStatusBarColor(colorBurn(vibrantSwatch.getRgb()));
14                  //设置底部功能栏的颜色
15                  window.setNavigationBarColor(colorBurn(vibrantSwatch.getRgb()));
16              }
17          }
18      }
19  }
```

扩展学习

本实例中只是获取了最具有活力的颜色，请尝试查阅 API 文档并获取其他色调。

实例 179　特殊提示信息效果

应用场景：用户反馈信息

源码位置：光盘\Code\18\179

实例说明

本实例中将使用 design 支持包下的 TextInputLayout 控件实现输入提示的界面效果（添加依赖：compile 'com.android.support:design:24.1.1'）。运行实例，界面的默认效果如图 18.17 所示，当输入内容时会在输入内容的上方显示输入提示，效果如图 18.18 所示。

图 18.17　界面默认的效果

图 18.18　输入内容有提示

关键技术

本实例主要使用 TextInputLayout 类中的 setError () 方法设置错误提示，使用 getEdit Text () 方法获取输入框对象。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“Reply”，公司域名为“mingrisoft. com”。

（2）导入资源文件：复制“光盘\Code\18\179\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）在 MainActivity 类的 onCreate() 方法中，初始化控件对象。代码如下：

```
/**
 * 初始化操作
 * @param savedInstanceState
 */
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
    email = (TextInputLayout) findViewById(R.id.email);
    code = (TextInputLayout) findViewById(R.id.code);
}
```

在 MainActivity 类中添加单击事件，模拟提交操作，在单击操作中增加判断，用来做提交验证。如果验证成功，则会弹出提示框通知用户提交成功，否则会在界面中以红色的文字来提示用户操作错误。代码如下：

```
/**
 * 单击事件模拟提交操作
 * @param view
 */
public void login(View view){
    if ("".equals(email.getEditText().getText().toString())){
        email.setError("邮箱不能为空");
    }else if ("".equals(code.getEditText().getText().toString())){
        code.setError("验证码不能为空");
    }else {
        code.setErrorEnabled(false);
        email.setErrorEnabled(false);
        Toast.makeText(this, "提交成功", Toast.LENGTH_SHORT).show();
    }
}
```

明日科技

扩展学习

请尝试更改提示文字的字体颜色。

实例 180　展开与收起菜单栏

应用场景：抽屉效果

源码位置：光盘\Code\18\180

实例说明

实现抽屉效果需要用到 DrawerLayout 布局控件。本实例中使用 DrawerLayout 来实现抽屉效果，而实现抽屉打开的布局是使用 design 支持包下的 NavigationView 来实现的。运行实例，抽屉打开的效果如图 18.19 所示，抽屉收起的效果如图 18.20 所示。

图 18.19　菜单栏展开效果

图 18.20　菜单栏收起效果

关键技术

关键技术

本实例主要使用 NavigationView 方法实现抽屉效果，使用 setNavigationItemSelectedListener() 方法来设置 NavigationView 中的子菜单的单击事件监听，使实现抽屉效果更加简洁。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“Sidebar”，公司域名为“mingrisoft. com”。

（2）导入资源文件：复制“光盘\Code\18\180\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）在 MainActivity 类的 onCreate() 方法中初始化控件，并使用 ToolBar 替代自带的 ActionBar（因为 AppCompatActivity 是 FragmentActivity 的子类，所以会带有 ActionBar），然后关联 ToolBar 与 DrawLayout，并设置子菜单的单击事件监听。代码如下：

代码位置：Sidebar\app\src\main\java\com\mingrisoft\sidebar\MainActivity.java

```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
    Toolbar toolbar = (Toolbar) findViewById(R.id.toolbar);
    //添加ToolBar来替代ActionBar
    setSupportActionBar(toolbar);
    DrawerLayout drawer = (DrawerLayout) findViewById(R.id.drawer_layout);
    //关联ActionBar与DrawerLayout
    ActionBarDrawerToggle toggle = new ActionBarDrawerToggle(
            this, drawer, toolbar, R.string.navigation_drawer_open,
            R.string.navigation_drawer_close);
    //添加抽屉状态的监听
    drawer.addDrawerListener(toggle);
    //与关联的DrawerLayout同步状态
    toggle.syncState();
    NavigationView navigationView = (NavigationView) findViewById(R.id.nav_view);
    //设置子菜单的点击监听事件
    navigationView.setNavigationItemSelectedListener(this);
}
```

在 onNavigationItemSelected() 方法中设置子菜单单击事件，触发时关闭抽屉。代码如下：

```
public boolean onNavigationItemSelected(MenuItem item) {
    DrawerLayout drawer = (DrawerLayout) findViewById(R.id.drawer_layout);
    drawer.closeDrawer(GravityCompat.START);                    //关闭抽屉
    return true;
}
```

扩展学习

请尝试为 NavigationView 控件中的菜单选项添加单击事件。

明日科技

第19章 商城常用

实例 181 转盘大抽奖

应用场景：控件效果

源码位置：光盘\Code\19\181

实例说明

一些商城类的 App 应用中会有一些抽奖的功能，本实例将实现一个转盘抽奖的效果。运行实例，将显示如图 19.1 所示的界面。当单击 GO 按钮后，转盘开始旋转，再次单击后转盘停止转动，并通过 Toast 将获取结果显示在界面上，如图 19.2 所示。

图 19.1 转盘效果

图 19.2 抽奖结果

关键技术

本实例主要通过自定义控件展示礼品和转盘的效果，需要使用 Canvas 的 drawBitmap() 方法来绘制图片。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“Lucky”，公司域名为“mingrisoft. com”。

（2）导入资源文件：复制“光盘\Code\19\181\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）创建 PrizeView 类，让其继承 View 类，并添加构造方法，然后在构造方法中进行初始化操作，即创建画笔 Paint、添加数据、创建属性动画。代码如下：

代码位置：Lucky\app\src\main\java\com\mingrisoft\lucky\PrizeView.java

```
paint = new Paint();                                          //创建画笔
paint.setAntiAlias(true);                                     //设置抗锯齿
paint.setStyle(Style.FILL);                                   //设置画笔的风格
paint.setColor(Color.YELLOW);                                 //设置画笔颜色
paint.setStrokeWidth(3);
addData();                                                    //添加数据
//设置动画
animator = ValueAnimator.ofInt(0,360);                        //创建动画
animator.addUpdateListener(new AnimatorUpdateListener() {
    @Override
    public void onAnimationUpdate(ValueAnimator animation) {
        //设置旋转角度
        startAngle = (Integer) animation.getAnimatedValue();
        invalidate();                                         //重绘
    }
});
animator.setInterpolator(new LinearInterpolator());          //设置为线性插值器
animator.setDuration(500);                                    //设置动画时间间隔为500毫秒
animator.setRepeatMode(ValueAnimator.RESTART);                //设置动画循环模式
animator.setRepeatCount(ValueAnimator.INFINITE);              //设置动画循环播放
```

重写 onMeasure() 方法，测量控件在布局中的宽度和高度，并根据测量数值和测量模式来给控件设置宽度和高度；重写 onDraw() 方法，绘制抽奖的转盘效果。代码如下：

```
screenW = getMeasuredWidth();
screenH = getMeasuredHeight();
int centerX = screenH / 2;                                    //获取中心点x坐标
int centerY = screenH / 2;                                    //获取中心点y坐标
float dis = screenH / 3f;
```

明日科技

```
paint.setStyle(Style.FILL);                                    //设置画笔的样式
RectF dst = new RectF(0, 0, screenW, screenH);                 //设置区域
RectF arcRect = new RectF(distanceR, distanceR, screenW - distanceR, screenH - distanceR);
canvas.drawBitmap(background, null, dst, paint);               //绘制背景
for (int i = 0; i < list.size();i++){
    //绘制礼物
    drawPrize(canvas, centerX, centerY, dis, arcRect,i);
}
//绘制指针图片
canvas.drawBitmap(centerBitmap, centerX - centerBitmap.getWidth() / 2,
        centerY - centerBitmap.getHeight() / 2, paint);
```

绘制抽奖转盘中的每个礼品。代码如下：

```
int color = (index + 1)% 2 == 0 ? 0xFFF17E01:0xFFFFC300;
paint.setColor(color);                                         //设置画笔的颜色
//绘制扇形
canvas.drawArc(arcRect, startAngle + index *
        sweepAngle, sweepAngle, true, paint);
//获取旋转角度
double radians = Math.toRadians(startAngle +
        sweepAngle / 2 + index * sweepAngle);
//获取图片绘制的x起始坐标
int x = (int) (centerX + dis * Math.cos(radians)) -
        prizeBitmap.get(index).getWidth() / 2;
//获取图片绘制的y起始坐标
int y = (int) (centerY + dis * Math.sin(radians)) -
        prizeBitmap.get(index).getHeight() / 2;
//绘制礼品图片
canvas.drawBitmap(list.get(index).bmp, x, y, paint);
//更新位置记录
list.get(index).startAngle = startAngle + index * sweepAngle;
list.get(index).endAngle =  list.get(index).startAngle + sweepAngle;
```

（4）在 activity_main 布局文件中添加自定义控件，注意标签名是包名加类名（代码位置：Lucky\app\src\main\res\layout\activity_main.xml）。

（5）在 MainActivity 类中获取抽奖转盘控件的对象，并给控件添加抽奖结果的回调添加监听。代码如下：

代码位置：Lucky\app\src\main\java\com\mingrisoft\lucky\MainActivity.java

```
//获取控件
prizeView = (PrizeView) this.findViewById(R.id.lotterywheel);
//获取抽奖的结果
```

```
04  prizeView.setOnSelectListener(new PrizeView.SelectListener() {
05      @Override
06      public void onSelect(String name) {
07          //弹出框提示抽奖结果
08          Toast.makeText(getApplicationContext(), name, Toast.LENGTH_SHORT).show();
09      }
10  });
```

扩展学习

请尝试为抽奖转盘的“Go”按钮增加单击切换效果。

实例 182　刮刮乐大抽奖

应用场景：抽奖活动页

源码位置：光盘\Code\19\182
视频位置：光盘\Video\19\182

实例说明

京东、淘宝等移动客户端经常推出各种刮奖活动，这种活动受到很多用户的喜爱。本实例将使用 Canvas 对象，实现“刮刮乐”的效果。运行实例，效果如图 19.3 所示；手指在灰色部分滑动，显示中奖信息，如图 19.4 所示；单击再次抽奖，将获取新的刮刮乐，如图 19.5 所示。

图 19.3　灰色部分滑动

图 19.4　单击再抽一次

图 19.5　新的刮刮乐

关键技术

本实例主要使用自定义 TextView 控件。在自定义 TextView 控件中，使用 Paint+Canvas 类完成对刮刮乐控件的绘制，使用 Random 类随机获取奖励信息。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“GuaGuaLe”，公司域名为“mingrisoft.com”。

（2）导入资源文件：复制“光盘\Code\19\182\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）在“app\java\com\mingrisoft\guaguale”文件夹下创建 manage 文件夹，并在其中创建 LotteryManage 类，用于设置中奖比例，以及中奖显示信息，代码如下：

```
//设置中奖比例 以及中奖显示信息
public class LotteryManage {
    public static LotteryInfo getRandomLottery() {
        LotteryInfo info = null;
        //声明随机类
        Random r = new Random();
        //获取1000以内随机数
        int num = r.nextInt(1000);
        if (num < 10) {    //判断随机数小于10
            info = new LotteryInfo(LotteryInfo.TYPE_FIRST, "一等奖");
        } else if (num < 30) {                                          //判断随机数小于30
            info = new LotteryInfo(LotteryInfo.TYPE_SECOND, "二等奖");
        } else if (num < 100) {                                         //判断随机数小于100
            info = new LotteryInfo(LotteryInfo.TYPE_THIRD, "三等奖");
        } else {                                                        //其他随机数
            info = new LotteryInfo(LotteryInfo.TYPE_NONE, "再接再厉");
        }
        return info;
    }
}
```

（4）在“app\java\com\mingrisoft\guaguale”文件夹下创建 util 文件夹，并在其中创建 MyView 类，让其继承 TextView 类，用于自定义刮刮乐控件。代码如下：

```
public class MyView extends TextView {
```

设置画笔样式。代码如下：

```
mPaint = new Paint();                                                   //初始化画笔
mPaint.setColor(mContext.getResources().getColor(R.color.view_color));  //设置画笔颜色
mPaint.setAlpha(0);                                                     //设置透明度
mPaint.setXfermode(new PorterDuffXfermode(Mode.DST_IN));                //设置相交模式
mPaint.setAntiAlias(true);                                              //设置防止锯齿
mPaint.setStyle(Style.STROKE);                                          //设置画笔空心画笔
mPaint.setStrokeWidth(50);                                              //画笔宽度
```

在 MyView 类中，使用 onTouchEvent() 方法处理手势事件。代码如下：

```
//手势判断
@Override
public boolean onTouchEvent(MotionEvent event) {
```

明日科技

```
04      int action = event.getAction();
05      switch (action) {
06          case MotionEvent.ACTION_DOWN:              //按下事件
07              touchDown(event);
08              break;
09          case MotionEvent.ACTION_MOVE:              //手指移动事件
10              touchMove(event);
11              break;
12          case MotionEvent.ACTION_CANCEL:
13          case MotionEvent.ACTION_UP:
14              break;
15      }
16      return true;
17  }
```

扩展学习

本实例运行的时候，中奖内容只是文字内容，尝试在中奖内容中显示文字和图片。

实例 183　随机抽奖

应用场景：抽奖活动

源码位置：光盘\Code\19\183

实例说明

随机抽奖是根据随机产生的数字来进行轮播图片的更换的。本实例将实现一个随机抽奖的效果。运行实例，效果如图 19.6 所示；单击抽奖转盘的中心位置，将显示随机产生的奖品信息，如图 19.7 所示。

图 19.6　单击抽奖转盘的中心位置

图 19.7　显示随机产生的奖品信息

关键技术

本实例主要使用 Random 类中的 nextInt() 方法、Message 类中的 getData() 方法、Thread 类中的 sleep() 方法，实现随机抽奖的运行效果。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“RandomDraw”，公司域名为“mingrisoft. com”。

（2）导入资源文件：复制“光盘\Code\19\183\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）打开 MainActivity 类，首先获取显示抽奖轮播图片的 ImageView 控件与图片资源。然后在 onTouchEvent() 方法中设置单击屏幕中心的位置来启动抽奖图片的轮播效果。代码如下：

```
01  //按下中间的位置时，60位中心点上下左右的距离
02  if(event.getX()>width/2-60 && event.getX()<width/2+60
03          && event.getY()>height/2-60 && event.getY()<height/2+60){
04      Random random = new Random();                                    //创建一层随机数
05      number = random.nextInt((8))+1;                                  //产生1~8随机数
06      new Thread(new Runnable() {                                      //创建线程
07          public void run() {
08              Random random = new Random();                            //创建二层随机数
09              int num = random.nextInt((8));                           //产生0~7随机数
10              while(number<60+num){                                    //循环次数
11                  Message message = new Message();                     //更换图片的消息
12                  switch (number%8) {                                  //计算圈数与余数为中奖位置
13                      case 0:
14                          //根据一层随机数与8的余数进行消息的发送，起到更换背景图片的作用
15                          message.what = MESSAGE_8;
16                          break;
17                      case 1:
18                          message.what = MESSAGE_1;
19                          break;
20                      case 2:
21                          message.what = MESSAGE_2;
22                          break;
23                  case 3:
24                      message.what = MESSAGE_3;
25                      break;
26                  case 4:
27                      message.what = MESSAGE_4;
28                      break;
29                  case 5:
30                      message.what = MESSAGE_5;
31                      break;
32                  case 6:
```

明日科技

```
                    message.what = MESSAGE_6;
                    break;
                case 7:
                    message.what = MESSAGE_7;
                    break;
                case 8:
                    message.what = MESSAGE_8;
                    break;
                default:
                    break;
            }
            number++;                                    //循环+1产生轮盘转动效果
            mHandler.sendMessage(message);               //发送消息
        }
    }
}
```

通过线程中 sleep() 方法实现抽奖时轮播图片的速度控制。代码如下：

```
if(number<50){                                           //随机数小于50
    try {
        Thread.sleep(100);                               //线程休眠时间为0.1秒
    } catch (InterruptedException e) {
        //TODO Auto-generated catch block
        e.printStackTrace();
    }
}else if(number>50 && number<60){                        //随机数大于50小于60
    try {
        Thread.sleep(300);                               //线程休眠时间为0.3秒
    } catch (InterruptedException e) {
        //TODO Auto-generated catch block
        e.printStackTrace();
    }
}else{
    try {
        Thread.sleep(400);                               //线程休眠时间为0.4秒
    } catch (InterruptedException e) {
        //TODO Auto-generated catch block
        e.printStackTrace();
    }
}
```

通过 Message 消息中的 setData() 方法将中奖轮播图的位置保存在消息中，再通过 sendMessage() 方法发送该消息。代码如下：

```
01  //停止消息的发送
02  Message message = new Message();                    //创建停止消息
03  message.what = MESSAGE_STOP;                        //设置要发送的停止消息
04  Bundle bundle = new Bundle();                       //创建Bundle对象
05  if ((number-1)%8==0){                               //跳出循环后number++，所以这里需要减1
06      bundle.putInt("msg",0);                         //保存停止的位置
07  }else {
08      bundle.putInt("msg",(number-1)%8);              //保存停止的位置
09  }
10  message.setData(bundle);                            //将位置信息保存在消息中
11  mHandler.sendMessage(message);                      //发送消息
```

扩展学习

本实例实现了单击抽奖轮盘的中心位置后启动抽奖的轮播图片。请尝试通过添加 isStart 标记，实现抽奖结束后获取再一次单击启动抽奖的界面，防止连续单击启动抽奖。

实例 184	订餐类购物车 应用场景：购物车类	源码位置：光盘\Code\19\184

实例说明

订餐类购物车通常根据购买物品的类型和数量进行统计并计算总价。运行实例，效果如图 19.8 所示；单击按钮选择需要购买的商品及购买数量，此时将显示购物车中商品的数量及总价，如图19.9所示。

图 19.8　添加需要购买的商品

图 19.9　显示购物车中商品的数量及总价

关键技术

本实例主要使用自定义类 OperateGoodsDataBase 中的自定义方法 getGoodsNumberAll()、getGoodsTypeList() 和 getGoodsPriceAll()，实现订餐购物车的运行效果。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“ShoppingCart”，公司域名为“mingrisoft. com”。

（2）导入资源文件：复制“光盘\Code\19\184\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）打开 Gradle Scripts 目录中的 build.gradle(Module:app) 文件，在该文件中添加依赖代码并单击窗口顶部的 Sync Now 进行同步。

（4）在“app\src\main\java”目录中创建 data 包，该包中包含 6 个类文件，用于保存数据相关类。通过 OperateGoodsDataBase 类中的自定义方法 getGoodsNumberAll() 获取所有已经购买的商品数量。代码如下：

```
01  public static int getGoodsNumberAll(Context context, int menupos) {
02      DbUtils utils = DbUtils.create(context);                          //创建数据库连接
03      int mGoodsNum = 0;                                                //所有购物数量
04      ArrayList<GoodsInfo> mGoodsInfoList = null;                       //商品类型总数
05      mGoodsInfoList = getGoodsTypeList(context);                       //获取所有商品类型
06      if(mGoodsInfoList == null){
07          return 0;
08      }
09      //根据商品类型的总数计算购物车中所有商品的总数
10      for(int i = 0; i < mGoodsInfoList.size() ; i++){
11          if(mGoodsInfoList.get(i).getMenupos() == menupos){            //计算总数
12              mGoodsNum += Integer.parseInt(mGoodsInfoList.get(i).getGoodsnum());
13          }
14      }
15      utils.close();                                                    //关闭数据库
16      return mGoodsNum;
17  }
```

通过自定义方法 getGoodsTypeList()，实现获取所有商品的类型，代码如下：

```
01  public static ArrayList<GoodsInfo> getGoodsTypeList(Context context){
02      ArrayList<GoodsInfo> list = null;                                 //商品信息
03      try {
04          //获取所有商品信息
05          list = (ArrayList<GoodsInfo>) DbUtils.create(context).findAll(GoodsInfo.class);
06          if(list == null){                                             //如果没有数据则返回空
```

```
            return null;
        }else{                                                    //否则返回商品信息
            return list;
        }
    } catch (DbException e) {
        e.printStackTrace();
    }
    return null;
}
```

通过自定义方法 getGoodsPriceAll()，实现获取购物车中所有已经购买商品的总价格，代码如下：

```
public static int getGoodsPriceAll(Context context, int menupos) {
    DbUtils utils = DbUtils.create(context);                      //创建数据库连接
    int mGoodsPrice = 0;
    ArrayList<GoodsInfo> mGoodsInfoList = null;                   //商品类型总数
    mGoodsInfoList = getGoodsTypeList(context);                   //获取所有商品类型
    if(mGoodsInfoList == null){
        return 0;
    }
    //根据商品类型的总数计算购物车中所有商品的总价
    for(int i = 0; i < mGoodsInfoList.size(); i++){
        if(mGoodsInfoList.get(i).getMenupos() == menupos){
            //计算总价
            mGoodsPrice += Integer.parseInt(mGoodsInfoList.get(i).getGoodsnum())
                    * Integer.parseInt(mGoodsInfoList.get(i).getGoodsprice());
        }
    }
    utils.close();
    return mGoodsPrice;
}
```

（5）在“app\src\main\jav”目录中创建 adapter 包，在该包中创建 RecyclerViewAdapter 类，用于订餐菜单的适配工作。然后创建 animation 包，在该包中创建 GoodsAnimation 类，用于实现商品添加购物车时的动画效果。

（6）打开 MainActivity 类，在该类中首先设置订餐菜单的显示，然后设置订餐数据的计算与动画效果。

扩展学习

本实例实现了订餐类购物车的总价统计功能。请尝试在 MainActivity 类中重写 onClick() 方法，实现为点餐按钮添加单击事件，并在该事件中显示共计消费的金额（加送餐费）。

实例 185 仿淘宝购物发表评价页

应用场景：发表评价页

源码位置：光盘\Code\19\185
视频位置：光盘\Video\19\185

实例说明

视频讲解

在 App 应用中经常会用到评价类的页面，像商城、O2O 类的订单都需要进行评价，那么星级评分是必不可少的。一般情况下，系统自带的 RatingBar 是远远无法满足开发需求的，本实例将使用图片自定一个 RatingBar。运行实例，将显示如图 19.10 所示的初始界面；从左向右滑动星星评价控件，改变星星控件选中状态，如图 19.11 所示；单击“发表评价”按钮显示服务评价信息，如图 19.12 所示。

图 19.10 初始界面

图 19.11 单击发表评价

图 19.12 显示服务评价

关键技术

关键技术

本实例主要使用 RatingBar 控件的自定义图片背景，使用 setOnRatingBatingChangeListener() 方法对 RatingBar 控件的改变事件进行监听。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“StarRating”，公司域名为“mingrisoft.com”。

（2）导入资源文件：复制“光盘\Code\19\185\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）根据图片自定义一个 RatingBar 的背景条 rating_bj.xml 文件，和图片放到同一个目录下，即放

到“app\res\drawable”文件夹下，代码如下：

```
<?xml version="1.0" encoding="utf-8"?>
<layer-list xmlns:android="http://schemas.android.com/apk/res/android">
    <!--默认显示的是RatingBar的样式-->
    <item
        android:id="@android:id/background"
        android:drawable="@drawable/start_off"></item>
    <!--与用户交互（选中评价等级）时的过渡样式-->
    <item
        android:id="@android:id/secondaryProgress"
        android:drawable="@drawable/start_off"></item>
    <!—RatingBar为当选中评价等级时的样式-->
    <item
        android:id="@android:id/progress"
        android:drawable="@drawable/start_on"></item>
</layer-list>
```

（4）RatingBar 的样式是通过自定义 styles 样式来切换的。在 Android 系统中，可以在“app\res\values”目录下的 styles.xml 文件中，通过 <style> 标记自定义样式，并让其继承开发者需要的控件类型。代码如下：

```
<style name="MyRatingBar" parent="@android:style/Widget.RatingBar">
    <item name="android:progressDrawable">@drawable/rating_bj</item>
    <item name="android:minHeight">18dp</item>
    <item name="android:maxHeight">18dp</item>
</style>
```

（5）在“app\res\layout”目录下默认生成的 activity_main.xml 布局文件中，添加 RatingBar 控件，并加入自定义的 style 属性。代码如下：

```
<!--显示星星评价控件-->
<RatingBar
    android:id="@+id/room_ratingbar1"
    style="@style/MyRatingBar"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginLeft="10dp"
    android:numStars="5"
    android:rating="4"></RatingBar>
```

（6）在“app\java\com\mingrisoft\starrating”文件夹下默认生成的 MainActivity 类中，使用 setOnRatingBatingChangeListener() 方法对 RatingBar 控件的改变事件进行监听并设置显示分数。代码如下：

```
//绑定星级评价控件的改变事件
ratingBar1.setOnRatingBarChangeListener(new RatingBar.OnRatingBarChangeListener() {
```

```
03      @Override
04      public void onRatingChanged(RatingBar ratingBar, float rating, boolean fromUser) {
05          //设置与描述相符的分数
06          fs1.setText(rating + "分");
07      }
08  });
```

扩展学习

本实例中设置的星星数量为 5 个，请尝试改变星星数量。实现方法是在布局文件 activity_main.xml 中的 RatingBar 控件里设置星星的数量。

实例 186　弹出小广告

应用场景：对话框

源码位置：光盘\Code\19\186

实例说明

当一些视频软件暂停播放视频的时候，会在屏幕中心的位置弹出一个广告。本实例将实现一个对话框小广告。运行实例，效果如图 19.13 所示；单击“启动广告”的按钮，在屏幕的中心将以对话框的方式弹出一个广告，如图 19.14 所示；单击对话框中的广告，将跳转到广告所链接的网页中，如图 19.15 所示。当显示图 19.15 界面时，单击对话框周围将关闭对话框。

图 19.13　单击“启动广告”按钮

图 19.14　单击广告对话框

图 19.15　显示广告对应的网页

关键技术

本实例主要使用 Uri 类中的 parse() 方法、Intent 类中的 Intent() 构造方法、Window 类，实现对话框小广告的运行效果。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“SmallAds”，公司域名为“mingrisoft. com”。

（2）导入资源文件：复制“光盘\Code\19\186\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）在“app\res\layout”目录中创建一个名称为“dialog.xml”的布局文件，在该文件中添加 ImageView 控件，用于显示对话框中的广告图片。（代码位置：app\res\layout\dialog.xml）

（4）打开 MainActivity 类，让其实现 View.OnClickListener 接口，并重写接口中的 onClick() 方法，在该方法中，实现对话框的显示。代码如下：

```
dlg = new AlertDialog.Builder(this,R.style.MyDialog).create();    //创建对话框实例
dlg.show();                                                       //显示对话框
Window window = dlg.getWindow();                                  //获取对话框窗口
window.setGravity(Gravity.CENTER);                                //此处可以设置对话框显示的中心位置
window.setContentView(R.layout.dialog);                           //设置对话框布局文件
```

获取对话框中显示广告的图像控件，然后为该控件设置单击事件，在该事件中进行网页的跳转。代码如下：

```
//获取显示广告的图像控件
ImageView adsImageView= (ImageView) window.findViewById(R.id.ads_img);
//小广告图像控件的单击事件
adsImageView.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        //设置跳转的网页地址
        Uri uri = Uri.parse("http://www.mingrisoft.com/");
        Intent intent = new Intent(Intent.ACTION_VIEW, uri);
        startActivity(intent);                                    //启动跳转
    }
});
```

扩展学习

本实例实现了对话框小广告的运行效果。请尝试在 MainActivity 类中的 onClick() 方法内，添加“dlg.setCanceledOnTouchOutside(false);”代码，实现在单击对话框周围时对话框不再关闭的效果。

实例 187　广告闪屏页

应用场景：启动 App 时的广告闪屏页

源码位置：光盘\Code\19\187
视频位置：光盘\Video\19\187

实例说明

视频讲解

在 App 应用中，大部分广告都在闪屏界面插入，插入的广告在计时完毕后，自动跳转到主界面，用户也可以单击“跳过”按钮直接进入到主界面。本实例将实现开启 App 应用后闪屏页的效果。运行实例，将显示如图 19.16 和图 19.17 所示的广告闪屏倒计时界面，单击跳过按钮进入主页。

图 19.16　闪屏页倒计时界面一

图 19.17　闪屏倒计时界面二

关键技术

关键技术

本实例主要使用 Timer 类中的 scedule() 方法，Handler 类中的 sendMessage() 方法，实现倒计时功能。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“SplashScreenTwo”，公司域名为“mingrisoft.com”。

（2）在“app\res\mipmap”文件夹中导入相关的图片资源。

（3）修改显示界面的主题样式，去除系统默认的 ActionBar 标题栏，打开“app\res\values\styles.xml”文件，将“DarkActionBar”修改为“NoActionBar”。

（4）在 MainActivity 类中，加载布局代码之前，将界面设置为全屏，关键代码如下：

```
//全屏
getWindow().setFlags(WindowManager.LayoutParams.FLAG_FULLSCREEN,
                                                    WindowManager.LayoutParams.FLAG_FULLSCREEN);
```

（5）设置全屏后添加布局，然后初始化控件。

（6）如果加载网络图片，需要给网络请求添加一个列队，初始化自定义图片缓存，然后开始加载网络图片，关键代码如下：

```
queue = Volley.newRequestQueue(this);                          //初始化单例列队
AllLruCacheLoad cache = new AllLruCacheLoad();                 //初始化缓存
final ImageLoader imageloader = new ImageLoader(queue, cache);   //初始化图片缓存
imageView.setImageUrl(url, imageloader);                       //开始加载图片
imageView.setDefaultImageResId(R.mipmap.ic_launcher);          //加载等待过程中显示的图片
```

（7）在闪屏界面执行倒计时操作时，每过 1 秒线程就会给 handler 发送一条消息，使显示的计时时间减 1，关键代码如下：

```
public void run() {
    Message message = new Message();                           //初始化message
    message.what = 1;
    message.obj = i;                                           //将剩余的计时时间传给handler
    handler.sendMessage(message);                              //发送消息给handler
    i--;                                                       //开始倒计时
}
0, 1000);                                                      //每隔1秒执行一次该方法
```

（8）在执行倒计时时，界面会显示剩余时间，当倒计时结束时，界面自动跳转到主界面，关键代码如下：

```
timeTv.setText(i + "");                                        //显示倒计时时间
if (i == 0) {                                                  //计时时间结束后，自动跳转主界面
    timer.cancel();                                            //取消计时器
    Intent intent = new Intent(MainActivity.this, HomeActivity.class);   //跳转到主界面
    startActivity(intent);                                     //开始跳转
    finish();                                                  //退出当前界面
}
```

扩展学习

本实例实现了在进入界面后便开始倒计时，请尝试设置延迟三秒之后开始倒计时的功能。

实例 188 收货地址管理

应用场景：管理收货地址

源码位置：光盘\Code\19\188

实例说明

商城、订餐类 App 应用需要对用户的收货地址进行管理，例如在发送订单的时候，需要输入收货地址。本实例将实现收货地址管理页面，主要体现城市选择的三级联动。运行实例，将显示如图 19.18 所示的界面；单击所在地区栏目，进入地区选择页面，如图 19.19 所示；选择完城市后单击“确认”按钮返回“编辑收货地址”页面，如图 19.20 所示。

图 19.18 单击“所在地区”

图 19.19 单击“确认”按钮

图 19.20 “编辑收货地址”页面

关键技术

关键技术

本实例主要使用 Activity 类中的 startActivityForResult () 方法、onActivityResult() 方法、setResult() 方法，AdapterView 类中的 onItemClick() 方法，SQLiteDatabase 类中的 openOrCreateDatabase() 方法，实现编辑及管理收货地址。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“DZGLThreeLD”，公司域名为“mingrisoft.com”。

（2）导入资源文件：复制“光盘\Code\19\188\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）在“res”目录下新建 raw 文件夹，用于存放城市数据库文件。

（4）在“app\java\com\mingrisoft\dzglthreeld”文件夹下默认生成的 MainActivity 类中，添加按钮单击事件，实现跳转到城市选择页面。代码如下：

```
//点击事件
@Override
public void onClick(View v) {
    if(v == tv_city1){
        //跳转到城市选择页面
        Intent in = new Intent(this, CitySelect1Activity.class);
        in.putExtra("city", city);
        //回调"1"
        startActivityForResult(in, 1);
    }
}
```

（5）在“app\java\com\mingrisoft\dzglthreeld”文件夹下创建 CityDBManager 类，用于管理数据库，主要打开数据库获取到数据库里的内容。代码如下：

```
public void openDatabase() {
    //打开数据库
    this.database = this.openDatabase(DB_PATH + "/" + DB_NAME);
}
```

创建关闭数据库的方法代码如下：

```
//关闭数据库
public void closeDatabase() {
    if (this.database != null)
        this.database.close();
}
```

（6）在“app\java\com\mingrisoft\dzglthreeld”文件夹下创建 CitySelect1Activity 类，用于城市选择，在其中创建内部类，用于地区数据适配。代码如下：

```
//自定义地区适配器
class CityAdapter extends ArrayAdapter<MyRegion> {
    LayoutInflater inflater;
    public CityAdapter(Context con) {
        super(con, 0);
        inflater = LayoutInflater.from(CitySelect1Activity.this);
    }
    @Override
    public View getView(int arg0, View v, ViewGroup arg2) {
        //加载单条item布局
        v = inflater.inflate(R.layout.city_item, null);
        //初始化显示地区内容控件
        TextView tv_city = (TextView) v.findViewById(R.id.tv_city);
        //设置显示内容
        tv_city.setText(getItem(arg0).getName());
        return v;
    }
```

明日科技

```
18      public void update() {
19          //更新列表内容
20          this.notifyDataSetChanged();
21      }
22  }
```

扩展学习

本实例在编辑收货地址时，单击“确认”按钮无效果。请尝试为“确认”按钮添加单击事件，实现单击后弹出输入的收货地址信息。

实例 189 使用瀑布流展示明日图书

应用场景：图片浏览

源码位置：光盘\Code\19\189

实例说明

App 应用在大量展示图片的时候，大多会使用瀑布流效果。本实例主要使用 RecycleView 控件实现瀑布流效果。运行实例，初始效果如图 19.21 所示；长按列表上条目可将该项删除，删除效果如图 19.22 所示；向上滑动可展示其他栏目，效果如图 19.23 所示。

图 19.21　长按栏目

图 19.22　向上滑动

图 19.23　展示更多栏目

关键技术

关键技术

本实例主要使用 MyRecyclerAdapter 类中的 MyRecyclerAdapter() 方法、ItemClickListener() 方法、

ItemLongClickListener() 方法、StaggeredGridLayoutManager 类中的 StaggeredGridLayoutManager() 方法，LayoutInflater 类中的 inflate() 方法。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“PBLDemo”，公司域名为“mingrisoft.com”。

（2）导入资源文件：复制“光盘\Code\19\189\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）使用 RecycleView 控件，需要在 build.gradle 配置文件的 dependencies 属性中添加引用。代码如下：

```
compile 'com.android.support:recyclerview-v7:24.1.1'
```

（4）在“app\res\layout”目录下默认生成的 activity_main.xml 布局文件中添加 RecycleView 控件。代码如下：

```
<android.support.v7.widget.RecyclerView
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:id="@+id/recyclerView">
</android.support.v7.widget.RecyclerView>
```

（5）在“app\java\com\mingrisoft\pbldemo”文件夹下创建 MyRecyclerAdapter 类，自定义适配器，用于装载数据到 RecycleView 控件中。

（6）在“app\java\com\mingrisoft\pbldemo”文件夹下默认生成的 MainActivity 类中，为 RecycleView 控件设置适配器，添加条目单击事件。代码如下：

```
//列表控件
mRecyclerView = (RecyclerView) this.findViewById(R.id.recyclerView);
mRecyclerView.setItemAnimator(new DefaultItemAnimator());
//设置RecyclerView布局管理器为2列垂直排布
mRecyclerView.setLayoutManager(new  StaggeredGridLayoutManager(2,
        StaggeredGridLayoutManager.VERTICAL));
adapter = new MyRecyclerAdapter(this,lists,resids);
mRecyclerView.setAdapter(adapter);
adapter.setOnClickListener(new MyRecyclerAdapter.OnItemClickListener() {
    //单击事件
    @Override
    public void ItemClickListener(View view, int postion) {
        Toast.makeText(MainActivity.this,"单击了：“+postion,Toast.LENGTH_SHORT).show();
    }
    //长按事件
    @Override
    public void ItemLongClickListener(View view, int postion) {
        //长按删除
        lists.remove(postion);
```

```
20            adapter.notifyItemRemoved(postion);
21        }
22    });
```

扩展学习

请尝试修改瀑布流的样式，实现在每个条目中添加一行说明文字。

实例 190　带时间轴效果的物流信息

应用场景：物流信息页

源码位置：光盘\Code\19\190

实例说明

在商城类 App 应用中经常会看到时间轴形式展示效果，如时间轴日记或者物流信息。本实例主要实现显示物流信息的效果。运行实例，显示如图 19.24 所示的物流信息界面；向上滑动，可查看物流信息，如图 19.25 所示。

图 19.24　显示物流信息界面

图 19.25　查看物流全部信息界面

关键技术

关键技术

本实例主要使用 List 类中的 add() 方法、TextView 类中的 setText() 方法、View 类中的 setTag() 方法、ListView 类中的 setAdapter() 方法，实现物流信息展示功能。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“SJZListView”，公司域名为“mingrisoft.com”。

（2）导入资源文件：复制“光盘\Code\19\190\src 中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）在“app\res\layou”目录下默认生成的 activity_main.xml 布局文件中，添加 ListView 控件，用于显示物流信息。

（4）在“app\res\layout”文件夹下新建 time_item.xml 布局文件，作为单条信息的布局文件。

（5）在“app\java\com\mingrisoft\sjzlistview”文件夹下默认生成的 MainActivity 类中，添加物流信息到物流信息数组中。代码如下：

```
01  //物流信息集合
02  list =new ArrayList<>();
03  list.add(new KuaiDi("2016-09-19 04:42:11","到达【广东省广州邮件处理中心】"));
04  list.add(new KuaiDi("2016-09-19 01:14:30","离开【深圳市处理中心】，下一站【广州市】"));
05  list.add(new KuaiDi("2016-09-18 23:18:21","到达【深圳市处理中心】"));
06  list.add(new KuaiDi("2016-09-18 21:26:51","离开【深圳市龙华函件中心】，下一站【深圳市】"));
07  list.add(new KuaiDi("2016-09-18 21:12:53","【深圳市龙华函件中心】已收寄"));
08  list.add(new KuaiDi("2016-09-19 04:42:11","到达【广东省广州邮件处理中心】"));
09  list.add(new KuaiDi("2016-09-19 01:14:30","离开【深圳市处理中心】，下一站【广州市】"));
10  list.add(new KuaiDi("2016-09-18 23:18:21","到达【深圳市处理中心】"));
11  list.add(new KuaiDi("2016-09-18 21:26:51","离开【深圳市龙华函件中心】，下一站【深圳市】"));
12  list.add(new KuaiDi("2016-09-18 21:12:53","【深圳市龙华函件中心】已收寄"));
13  list.add(new KuaiDi("2016-09-18 08:51:33","您的包裹已出库"));
14  list.add(new KuaiDi("2016-09-18 08:40:23","您的订单待配货"));
15  list.add(new KuaiDi("2016-09-18 08:33:50","您的订单开始处理"));
```

为 ListView 绑定自定义适配器，填充数据。代码如下：

```
01  adapter = new KuaiDiAdapter(this,list);        //初始化适配器
02  lv.setAdapter(adapter);                         //绑定适配器
```

（6）在“app\java\com\mingrisoft\sjzlistview”文件夹新建 KuaiDiAdapter 类，用于数据的适配，设置数据。代码如下：

```
01  @Override
02  public View getView(int position, View convertView, ViewGroup parent) {
03      ViewHolder holder;
04      if (convertView == null) {
05          convertView = mInflater.inflate(R.layout.time_item, null);
06          holder = new ViewHolder();
07          holder.tv_content = (TextView) convertView.findViewById(R.id.tv_content);
08          holder.tv_time = (TextView) convertView.findViewById(R.id.tv_time);
09          convertView.setTag(holder);          //绑定ViewHolder对象
10      } else {
```

```
11            holder = (ViewHolder) convertView.getTag();        //取出ViewHolder对象
12        }
13        KuaiDi kd = list.get(position);
14        holder.tv_content.setText(kd.getContent());
15        holder.tv_time.setText(kd.getTime());
16        return convertView;
17    }
```

扩展学习

尝试修改物流信息内容。

明日科技

第20章 验证码合集

实例 191 仿 12306 售票网验证码

应用场景：验证码

源码位置：光盘\Code\20\191

视频位置：光盘\Video\20\191

视频讲解

实例说明

本实例将实现仿 12306 售票网的验证码功能。运行实例，将显示如图 20.1 所示的初始界面；单击与验证码提示内容相符的图片，然后单击“登录”按钮，将显示验证成功的界面，如图 20.2 所示。

图 20.1 初始界面效果

图 20.2 验证码验证成功

关键技术

本实例主要使用 CodeDataUtils 自定义类中的 getDataList 方法，MainActivity 类中的 selectCode() 方法、onClick() 方法，实现仿 12306 售票网验证码的功能。

实现过程

（1）打开 Android Studio，新建一个项目，将其命名为“SelectImageCode”，公司域名为“mingrisoft. com”。

（2）导入资源文件：复制“光盘\Code\20\191\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）打开“Gradle:Scripts”目录中的“.build.gradle(Module:app)”文件，在该文件中添加依赖代码，并单击窗口顶部的 sync:Now 进行同步。

（4）在 SelectImageCode 项目中的“app\src\main\java”目录下创建 code 包，用于获取验证码数据，该包中包括 4 个类文件，其类名及作用如表 1.1 所示。

表 20.1　util 包中的四个类

类　名	描　述
Code	用来存储验证码数据
CodeDataUtils	用来获取验证码数据
CodeList	用来存储验证码类型和验证码集合
CodeType	验证码类型

（5）在 MainActivity 类中声明变量，为创建验证码做准备。代码如下：

代码位置：SelectImageCode\app\src\main\java\com\mingrisoft\selectimageCode\MainActivity.java

```
01  private boolean[] isSelected = new boolean[8];                          //标记
02  private int[] bgColor = {0xffff0000, 0xff000000, 0x00000000};           //背景颜色
03  private CompositeSubscription subscription = new CompositeSubscription();
04  private boolean hasData;                                                //是否有数据
05  private List<Code> allCode;                                             //所有验证图片
06  private List<CodeType> typeCode;                                        //所有类别
07  private Random mRandom = new Random();                                  //获取随机数
08  private int[] indexArray = {0, 1, 2, 3, 4, 5, 6, 7};                    //标记数组
09  private ArrayList<String> selectCode;                                   //验证信息
10  private String codeName;                                                //验证码条件
11  private ArrayList<Integer> showImageList = new ArrayList<>();           //筛选数据集合
12  private ArrayList<String> codeString = new ArrayList<>();              //验证字符串
13  private String[] codeNumberInAll = new String[2];                       //用于保存验证值
14  private ArrayList<Code> showCodeList = new ArrayList<>();               //用于显示的验证图片的集合
15  private ArrayList<Integer> indexRecordList = new ArrayList<>();         //用于记录标记位
```

在 MainActivity 类中，创建 getImageCodeData() 方法，用来初始化数据，并将资源文件的图片获取出来，并保存到集合中。代码如下：

```
/**
 * 获取验证码数据
 */
private void getImageCodeData() {
    Observable<CodeList> codeListObservable =
            Observable.create(subscriber -> {
                if (!subscriber.isUnsubscribed()) {
                    subscriber.onNext(CodeDataUtils.getDataList(this));
                }
            });
    Subscription codeListSubscription = codeListObservable
            .subscribeOn(Schedulers.io())
            .observeOn(AndroidSchedulers.mainThread())
            .subscribe(codeList -> {
                if (codeList != null) {
                    allCode = codeList.getCodeList();              //获取数据
                    typeCode = codeList.getTypeList();             //获取类别
                    Logger.t("图片总数据数量").i(allCode.size() + "");
                    Logger.t("图片类别数量").i(typeCode.size() + "");
                    hasData = true;                                //标识数据已获取
                } else {
                    hasData = false;                               //数据获取失败
                    showToast("初始化数据失败");
                }
                selectCode = new ArrayList<>();                    //保存验证数据
                imageViewBackground();                             //初始化图片的背景
                addCodeImageToWindow();                            //添加验证图片
            });
    subscription.add(codeListSubscription);
}
```

在 MainActivity 类中，创建 addCodeImageToWindow() 方法，用来向界面中添加验证图片。代码如下：

```
/**
 * 添加验证图片
 */
private void addCodeImageToWindow() {
    if (hasData == false) {
        return;
    }
    //获取随机类别
```

```
09      int typeNum = mRandom.nextInt(typeCode.size());
10      CodeType type = typeCode.get(typeNum);                    //获取验证码类别
11      codeName = type.getType();                                //获取验证码条件
12      text.setText(Html.fromHtml("请单击下面所有的:"+
13              "<font color = '#ff0000'>"
14              + codeName + "</font>"));                         //显示类别
15      saveCodeMessage(type.getCode());                          //保存数据
16      getCodeImageFromAll();                                    //获取其他干扰数据
17      setCodeShowIndex();                                       //设置图片的索引
18      showImageInWindow();                                      //在界面中显示图片
19  }
```

创建 onClick() 方法，用来执行登录验证，验证成功后弹出 Toast 提示登录成功。代码如下：

```
01  /**
02   * 登录验证
03   */
04  @OnClick(R.id.login)
05  public void onClick() {
06     if (selectCode.size() > 0){
07         //创建StringBuilder对象
08         StringBuilder sb = new StringBuilder();
09         for (int i = 0;i < selectCode.size();i++){
10             //添加指定数据的验证数据
11             sb.append(showCodeList.get(
12                     Integer.parseInt(selectCode.get(i)) - 1).getId());
13         }
14         for (int i = 0;i < codeString.size();i++){
15             if (TextUtils.equals(sb.toString(),codeString.get(i))){
16                 showToast("登录成功！");
17                 return;
18             }
19         }
20         showToast("登录失败，验证码有误！");
21     }else {
22         showToast("请选择验证码！");
23     }
24  }
```

扩展学习

请尝试给密码输入框增加提示文字。

实例 192　成语验证码

应用场景：验证码

源码位置：光盘\Code\20\192

实例说明

本实例中，用户按照文本框的提示依次单击带有文字的按钮，使上面的文字组成一个成语，以实现验证码的功能。运行实例，将显示如图 20.3 所示的初始界面，界面中会显示“输入验证码”的文本框和“验证码”的文本按钮；根据提示依次单击带有对应验证码文字的按钮，完成验证码的输入，如图 20.4 所示。

图 20.3　初始界面效果

图 20.4　输入验证码的效果

关键技术

本实例主要使用 MainActivity 类中的 getCodeWord() 自定义方法，selectWordBtn() 自定义方法、onClick() 方法，实现单击带有文字按钮的验证码的功能。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“IdiomCode”，公司域名为“mingrisoft. com”。

（2）导入资源文件：复制“光盘\Code\20\192\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）打开“Gradle:Scripts”目录中的“.build.gradle(Module:app)”文件，在该文件中添加依赖代码，并单击窗口顶部的 sync:Now 进行同步。

（4）在 MainActivity 类中声明变量，为创建验证码做准备。代码如下：
代码位置：IdiomCode\app\src\main\java\com\mingrisoft\idiomcode\MainActivity.java

```
01  private Random mRandom = new Random();                                  //用于生成随机数
02  private CompositeDisposable disposables = new CompositeDisposable();
03  private Code mCode;                                                     //数据
04  private String inputCodeWord;                                           //被用来验证的数据
05  private ArrayList<Integer> codeWordIndexInArrayList = new ArrayList<>();
06  private ArrayList<String> showWordList = new ArrayList<>();             //抽取的汉字集合
07  private ArrayList<String> selectedWord = new ArrayList<>();             //选择出来的字符串
```

创建 getWord() 方法，用来初始化验证码数据。代码如下：

```
01  /**
02   * 获取数据
03   */
04  private void getWordData() {
05      Flowable<Code> observable = Flowable.create(e ->
06              e.onNext(WordUtils.GetWordData()), BackpressureStrategy.LATEST);
07      disposables.add(observable.subscribeOn(Schedulers.io())
08              .observeOn(AndroidSchedulers.mainThread())                  //线程切换
09              .map(code -> {
10                  if (null != code) mCode = code;                         //获取数据
11                  return true;
12              }).subscribe(aBoolean -> {
13                  if (aBoolean) getCodeWord();                            //生成验证码
14              });                                                         //获取数据
15  }
```

重写 getInputCodeWordFromData() 方法，用于获取数据后生成验证码，并将验证码显示在界面中。代码如下：

```
01  /**
02   * 从数据中获取成语作为验证码
03   */
04  private void getInputCodeWordFromData() {
05      //获取随机数
06      int index = mRandom.nextInt(mCode.getStringArray().length);
07      inputCodeWord = mCode.getStringArray()[index].trim();               //文字验证码
08      showCode.setText(inputCodeWord);                                    //显示验证码
09      for (char c: inputCodeWord.toCharArray()){
10          showWordList.add(String.valueOf(c));                            //保存验证
11      }
```

```
12        //显示验证码图片
13        imageCode.setBackground(new CodeImage(this,inputCodeWord));
14    }
```

重写 onClick() 方法，用来判断输入的验证码是否有效。代码如下：

```
01  /**
02   * 验证验证码
03   */
04  @OnClick(R.id.send)
05  public void onClick() {
06      if (selectedWord.size() > 0) {
07          StringBuilder sb = new StringBuilder();
08          for (TextView textView : inputViewList) {
09              //获取输入的验证码
10              sb.append(textView.getText().toString());
11          }
12          String result = TextUtils.equals(inputCodeWord,
13                  sb.toString()) ? "发布成功！" : "验证码错误！";
14          showToast(result);
15      } else {
16          showToast("请输入验证码！");
17      }
18  }
```

扩展学习

请尝试将生成的验证码文字设置为不同的颜色。

实例 193　文字过滤验证码

应用场景：验证码

源码位置：光盘\Code\20\193

实例说明

本实例将实现文字过滤验证码的功能。运行实例，初始效果如图 20.5 所示；在输入框的下方，会出现绿色的带有下划线的验证码提示信息，按照提示信息输入验证文字，如图 20.6 所示。

明日科技

图 20.5　初始界面效果

图 20.6　输入验证码的效果

关键技术

本实例主要使用集合，以及对字符串的裁剪和拼接的方法，实现文字过滤验证码的功能。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“WordHineCode”，公司域名为“mingrisoft.com”。

（2）导入资源文件：复制“光盘\Code\20\193\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）在 MainActivity 类中声明变量，为创建验证码做准备。代码如下：

```
01  private Random mRandom = new Random();                                  //用于生成随机数
02  private CompositeDisposable disposables = new CompositeDisposable();
03  private ArrayList<Integer> codeArrayList = new ArrayList<>();          //文字
04  private ArrayList<Integer> wordArrayList = new ArrayList<>();          //存储验证码集合
05  private String[] codeWord;                                             //验证码索引
06  private Code mCode;                                                    //数据
07  private String hiitStr = "划线";                                        //提示关键字
```

创建 getWordData() 方法，在该方法中通过异步操作初始化数据。代码如下：

```
01  /**
02   * 获取数据
03   */
04  private void getWordData() {
05      Flowable<Code> observable = Flowable.create(e ->
```

```
            e.onNext(WordUtils.GetWordData()), BackpressureStrategy.LATEST);
    disposables.add(observable.subscribeOn(Schedulers.io())
            .observeOn(AndroidSchedulers.mainThread())          //线程切换
            .map(code -> {
        if (null != code) mCode = code;                         //获取数据
        return true;
    }).subscribe(aBoolean -> {
        if (aBoolean) getCodeWord();                            //生成验证码
    });                                                         //获取数据
}
```

创建 getCodeWord() 方法，在该方法中获取数据后生成验证码，并展示在界面中。代码如下：

```
/**
 * 生成验证码
 */
private void getCodeWord() {
    reset();
    //随机获取4个汉字
    getHintWordData(mCode.getCordArray(),codeArrayList);
    //随机获取4个字母
    getHintWordData(mCode.getWordArray(),wordArrayList);
    //在4个汉字中随机出来两个字作为验证码
    getCodeFromWord();
    //显示验证码输入提示文字
    //（请一次输入"龙p肥r置s应i"中"划线"的汉字）
    //拼接字符串
    String codeHintStr = addWordCode();
    String hintText = "（请依次输入"" + codeHintStr +""中"" + hiitStr +""的汉字）";
    int firstIndex = hintText.indexOf(codeWord[0]);
    int secondIndex = hintText.indexOf(codeWord[1]);
    Log.i("字体索引" , firstIndex + "--" + secondIndex);
    SpannableString hintStr = new SpannableString(hintText);
    //设置字体效果
    hintStr.setSpan(new ForegroundColorSpan(Color.parseColor("#008000")),
            18,20, SpannableString.SPAN_INCLUSIVE_INCLUSIVE);
    //设置下划线
    hintStr.setSpan(new UnderlineSpan(),18,20, SpannableString.SPAN_INCLUSIVE_INCLUSIVE);
    //设置字体效果
    hintStr.setSpan(new ForegroundColorSpan(Color.parseColor("#008000")),
            firstIndex,firstIndex+1, SpannableString.SPAN_INCLUSIVE_INCLUSIVE);
    //设置下划线
    hintStr.setSpan(new UnderlineSpan(),firstIndex,firstIndex+1,
            SpannableString.SPAN_INCLUSIVE_INCLUSIVE);
    //设置字体效果
```

明日科技

```
33      hintStr.setSpan(new ForegroundColorSpan(Color.parseColor("#008000")),
34              secondIndex,secondIndex + 1, SpannableString.SPAN_INCLUSIVE_INCLUSIVE);
35      //设置下划线
36      hintStr.setSpan(new UnderlineSpan(),secondIndex,secondIndex + 1,
37              SpannableString.SPAN_INCLUSIVE_INCLUSIVE);
38      hint.setText(hintStr);
39  }
```

创建 onClick() 方法，用来判断输入的验证码是否有效。代码如下：

```
01  /**
02   * 验证验证码
03   */
04  @OnClick(R.id.send)
05  public void onClick() {
06      if (!"".equals(input.getText().toString())){
07          if (TextUtils.equals(input.getText().toString(),codeWord[0] + codeWord[1])){
08              showToast("验证成功!");
09          }else {
10              showToast("验证码有误!");
11          }
12          getCodeWord();
13      }else {
14          showToast("请输入验证码!");
15      }
16  }
```

扩展学习

请尝试对输入框中字符长度增加限制，使输入框中输入的字符长度不超过两个字符。

实例 194 汉字转拼音验证码

应用场景：验证码

源码位置：光盘\Code\20\194

实例说明

本实例将实现汉字转拼音验证码的功能。运行实例，初始效果如图 20.7 所示，验证码文字的颜色为红色；其中有一个字被拼音所替代，按照提示文字输入正确的验证码内容，如图 20.8 所示。

明日科技

图 20.7　初始界面效果

图 20.8　输入验证码的效果

关键技术

关键技术

本实例主要使用 MainActivity 类中的 getInputCodeWordFromData() 自定义方法、getCodeWord() 自定义方法与 validation() 自定义方法，实现汉字转拼音的验证码功能。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“PinYinHintCode”，公司域名为“mingrisoft.com”。

（2）导入资源文件：复制“光盘\Code\20\194\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）打开 Gradle:Scripts 目录中的“.build.gradle(Module:app)”文件，在该文件中添加依赖代码，并单击窗口顶部的 sync:Now 进行同步。

（4）在 MainActivity 类中声明变量，为创建验证码做准备。代码如下：

代码位置：PinYinHintCode\app\src\main\java\com\mingrisoft\pinyinhintcode\MainActivity.java

```
01  private Random mRandom = new Random();                              //用于生成随机数
02  private CompositeDisposable disposables = new CompositeDisposable();
03  private String[] dataCodeArray;                                     //验证码数组
04  private String inputCodeWord;                                       //用于加载被验证的字符串
05  private ArrayList<String> showWordList = new ArrayList<>();         //抽取的汉字集合
```

创建 getWordData() 方法，在该方法中通过异步操作初始化数据。代码如下：

明日科技

```
/**
 * 获取数据
 */
private void getWordData() {
    Flowable<String[]> observable = Flowable.create(e ->
            e.onNext(WordUtils.GetWordData()), BackpressureStrategy.LATEST);
    disposables.add(observable.subscribeOn(Schedulers.io())
            .observeOn(AndroidSchedulers.mainThread())                    //线程切换
            .map(str -> {
        if (null != str) dataCodeArray = str;                             //获取数据
        return true;
    }).subscribe(aBoolean -> {
        if (aBoolean) getCodeWord();                                      //生成验证码
    });                                                                   //获取数据
}
```

创建 getCodeWord() 方法，用来获取数据后生成验证码，并展示在界面中。代码如下：

```
/**
 * 获取验证码文字
 */
private void getCodeWord() {
    inputCodeWord = null;
    showWordList.clear();
    //首先随机取一个成语作为验证码
    getInputCodeWordFromData();
    //清空输入框中的字
    editText.setText("");
    int index = mRandom.nextInt(4);
    String word = showWordList.get(index);                                //获取验证码
    String pinYin = CharacterParser.getInstance().getSelling(word);
    String showMessage = inputCodeWord.replace(word, pinYin);             //替换字符
    inputHint.setText(Html.fromHtml("请输入" + "<font color='#ff0000'>" +
            showMessage + "</font>" + "【提示：" + pinYin + "=" + word + "】"));
}
```

创建 validation() 方法，用来判断输入的验证码是否有效。代码如下：

```
/**
 * 验证
 */
@OnClick(value = R.id.validation_code)
public void validation() {
    if (!"".equals(inputCode.getText())) {
        if (inputCode.getText().equals(inputCodeWord)){
            showToast("验证成功！");
```

明日科技

```
09            }else {
10                showToast("验证失败！");
11            }
12            getCodeWord();
13        } else {
14            showToast("请输入验证码！");
15        }
16    }
```

扩展学习

请尝试增加一个按钮，用来刷新验证码。

实例 195　汉字验证码

应用场景：验证码

源码位置：光盘\Code\20\195

实例说明

本实例将实现单击文本按钮输入验证码的功能。运行实例，将显示如图 20.9 所示的初始界面，在界面中会显示验证码提示信息和选择验证码的文本按钮；根据提示信息单击文本按钮来选择验证码，如果选择的验证文字与所提示的文字吻合，则验证成功，将显示如图 20.10 所示的界面。

图 20.9　初始界面效果

图 20.10　验证码验证成功

关键技术

本实例主要使用了 MainActivity 类中的 getCodeWord() 自定义方法、selectWordBtn() 自定义方法与 onClick() 方法，实现单击文字按钮输入验证码的功能。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“WordCode”，公司域名为“mingrisoft. com”。

（2）导入资源文件：复制“光盘\Code\20\195\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）打开 Gradle:Scripts 目录中的“.build.gradle(Module:app)”文件，在该文件中添加依赖代码，并单击窗口顶部的 sync:Now 进行同步。

（4）在 MainActivity 类中声明变量，为创建验证码做准备。代码如下：

代码位置：WordCode\app\src\main\java\com\mingrisoft\wordcode\MainActivity.java

```
01  private char[] wordData;                                          //验证文字数组
02  private CompositeDisposable disposables = new CompositeDisposable();
03  private Random mRandom = new Random();                            //用于生成随机数
04  //用来储存验证码的汉字的索引值
05  private ArrayList<Integer> codeWordIndexInArrayList = new ArrayList<>();
06  private ArrayList<Integer> showWordIndexInArrayList = new ArrayList<>();
07  private ArrayList<String> showWordList = new ArrayList<>();       //抽取的汉字集合
08  private String inputCodeWord;                                     //被用来验证的数据
09  private ArrayList<String> selectedWord = new ArrayList<>();       //选择出来的字符串
```

通过异步操作初始化数据。代码如下：

```
01  /**
02   * 获取数据
03   */
04  private void getWordData() {
05      Flowable<char[]> observable = Flowable.create(e ->
06              e.onNext(WordUtils.GetWordData()), BackpressureStrategy.LATEST);
07      disposables.add(observable.subscribeOn(Schedulers.io())
08              .observeOn(AndroidSchedulers.mainThread())            //线程切换
09              .map(chars -> {
10          if (null != chars) wordData = chars;                      //获取数据
11          return true;
12      }).subscribe(aBoolean -> {
13          if (aBoolean) getCodeWord();                              //生成验证码
14      });                                                           //获取数据
15  }
```

创建 getInputCodeWordFromData() 方法，用来获取数据后生成验证码并展示在界面中。代码如下：

```
/**
 * 从数据中获取4个验证汉字组成验证码
 */
private void getInputCodeWordFromData() {
    StringBuilder sb = new StringBuilder();
    int count = 0;                                                    //设置计数器
    while (count < INPUT_CODE_COUNT) {
        //获取随机数
        int index = mRandom.nextInt(wordData.length);
        if (detection(index, codeWordIndexInArrayList)) {
            codeWordIndexInArrayList.add(index);                      //储存索引
            sb.append(wordData[index]);                               //获取文字
            count++;                                                  //计数器自增
        }
    }
    inputCodeWord = sb.toString();                                    //文字验证码
    showCode.setText(inputCodeWord);                                  //显示验证码
    imageCode.setBackground(new CodeImage(this,inputCodeWord));
}
```

创建 onClick() 方法，用来判断输入的验证码是否有效。代码如下：

```
/**
 * 验证按钮
 */
@OnClick(R.id.test)
public void onClick() {
    if (selectedWord.size() > 0) {
        StringBuilder sb = new StringBuilder();
        for (TextView textView : inputViewList) {
            //获取选择的验证文字
            sb.append(textView.getText().toString());
        }
        String result = TextUtils.equals(inputCodeWord,
                sb.toString()) ? "发布成功！" : "验证码错误！";
        showToast(result);                                            //提示验证结果
    } else {
        showToast("请输入验证码！");
    }
}
```

扩展学习

本实例在更换验证码的时候，验证码选择中的文字并没有发生改变。请尝试在切换验证码时，更改文字的显示效果。

明日科技

实例 196　图片选择验证码

应用场景：验证码

源码位置：光盘\Code\20\196

实例说明

本实例将实现图片选择验证码的功能。运行实例，将显示如图 20.11 所示的界面，在验证码图片中将以乱序显示诗句中的单个汉字；按照右侧提示中随机选取的文字按顺序单击验证码图片中的文字，会在选中的验证码上面显示序号，如图 20.12 所示。

图 20.11　生成验证码

图 20.12　选择验证码

关键技术

本实例主要使用 View 类中的 onDraw() 方法、onMeasure() 方法、onSizeChanged() 方法、onTouchEvent() 方法，从而实现图片选择验证码的功能。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“CordCard”，公司域名为“mingrisoft. com”。

（2）导入资源文件：复制“光盘\Code\20\196\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）创建 CodeImageView 类，继承 View 类，并添加构造方法，然后在构造方法中添加初始化操作，主要是创建画笔 Paint。代码如下：

代码位置：CordCard\app\src\main\java\com\mingrisoft\cordcard\view\CodeImageView.java

```
01  /**
02   * 初始化
```

```
 */
private void initialize() {
    paint = new Paint();
    paint.setStyle(Paint.Style.STROKE);                    //设置油漆桶
    paint.setDither(true);                                 //设置防抖动
    paint.setAntiAlias(true);                              //设置抗锯齿
    paint.setTextSize(textSize);                           //设置字体的大小
    paint.setTextAlign(Paint.Align.CENTER);                //设置文字对齐格式
}
```

（4）重写 onMeasure() 方法，用来测量控件在布局中的宽度和高度，根据测量数值和测量模式来给控件设置宽度和高度。

（5）重写 onDraw() 方法，用于绘制验证码。代码如下：

```
@Override
protected void onDraw(Canvas canvas) {
    super.onDraw(canvas);
    if (null != background){
        canvas.drawBitmap(background,null,new Rect(startX,startY,mWidth,mHeight),null);
    }
    if (null == mCodeDatas) {
        paint.setColor(Color.BLACK);
        canvas.drawText("请设置数据源", mWidth / 2, getBaseLine(startY, mHeight), paint);
        return;
    }

    //标记点
    int count = 0;
    for (KeyPoint point : points) {
        canvas.save();                                     //保存画布的当前状态
        //旋转画布
        canvas.rotate(point.getAngle(), point.getX(), point.getY());
        //绘制文字
        paint.setStrokeWidth(0);
        paint.setColor(Color.BLACK);
        Rect drawRect = rects.get(count);                  //获取绘制区域
        canvas.drawText(mCodeDatas.get(count), point.getX(),
                        getBaseLine(drawRect.bottom, drawRect.top), paint);
        //绘制选择的验证文字
        if (showKeyPoint) {
            //绘制坐标点
            paint.setStrokeWidth(dp2px(5));
            paint.setColor(Color.YELLOW);
            canvas.drawPoint(point.getX(), point.getY(), paint);
            //绘制文字绘制区域
```

明日科技

```
            paint.setColor(Color.YELLOW);
            paint.setStrokeWidth(dp2px(2));
            canvas.drawRect(drawRect.left, drawRect.top, drawRect.right,
                    drawRect.bottom, paint);
        }
        count++;
        canvas.restore();                                          //恢复画布的状态
    }
    //绘制覆盖层
    if (null != selectLayer && selectLayer.size() > 0) {
        for (int i = 0; i < selectLayer.size(); i++) {
            Rect rect = selectLayer.get(i);                        //获取文字绘制的区域
            Bitmap bitmap = BitmapFactory.decodeResource(getResources(),
                    bitmapRes[i]);                                 //获取图片
            Rect drawRect = new Rect(rect.left + layerOffset, rect.top + layerOffset,
                    rect.right - layerOffset, rect.bottom - layerOffset); //设置绘制区域
            canvas.drawBitmap(bitmap, null, drawRect, null);       //绘制覆盖物
        }
    }
}
```

（6）在 MainActivity 类的 onCreate() 方法中初始化控件。代码如下：

代码位置：CordCard\app\src\main\java\com\mingrisoft\codecard\MainActivity.java

```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
    ButterKnife.bind(this);
    codeView.setOnCodeSelectedListener(this);                      //设置监听获取选择回调
    getWordData();                                                 //获取验证码数据
}
```

（7）在 MainActivity 类中创建 getCodeWord() 方法，用来生成验证码，并将验证码显示在界面中。代码如下：

```
private void getCodeWord() {
    //随机从数组中获取一个字符串作为验证码数据源的索引
    int index = mRandom.nextInt(wordData.length);
    //随机获取一个图片作为验证码选择的背景图片
    int bitmapIndex = mRandom.nextInt(8) + 1;
    //随机生成验证码的数
    int codeCount = mRandom.nextInt(3) + 2;
    //获取验证码数据
    String selectCode = wordData[index];                           //根据索引获取数据源
    List<String> list = new ArrayList<>();                         //用来保存数据源
```

```
11      char[] array = selectCode.toCharArray();                    //获取数据源数组
12      for (int i = 0; i < array.length; i++) {
13          list.add(String.valueOf(array[i]));                     //获取数据源集合
14      }
15      codeView.setSelectOptionsData(list, codeCount);
16      Bitmap background = getRes("picture" + String.valueOf(bitmapIndex));
17      codeView.setCodeBackground(background);
18      //生成验证码
19      generate(list, codeCount ,background);
20  }
```

扩展学习

请尝试将生成验证码的字体颜色设置为白色。

实例 197　拖动形式的验证码

应用场景：用户登录时

源码位置：光盘\Code\20\197

实例说明

本实例将实现拖动形式的验证码功能。运行实例，将显示如图 20.13 所示的初始界面；向右拖动图中的进度条，如图 20.14 所示；当①号星星图标与②号星星图标重合完成验证，如图 20.15 所示。

图 20.13　初始界面

图 20.14　拖动图标

图 20.15　验证码图标重合

关键技术

本实例主要使用 Canvas 类中的 drawBitmap() 方法、Math 类中的 random() 方法，实现设置绘制图片及随机生成滑动验证的位置。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“SlidingVerification”，公司域名为“mingrisoft.com”。

（2）导入资源文件：复制“光盘\Code\20\197\src”文件夹中的图片资源，粘贴到“app\res\mipmao”文件夹中，用于设置滑动验证的背景图片。

（3）去掉默认的标题栏，在“app\res\values\styles”文件中将“DarkActionBar”改为“NoActionBar”。

（4）创建滑动验证类 SlidingVerificationView，用于滑动验证的实现及监听。

（5）在滑动验证类中获取滑块随机生成的坐标，关键代码如下：

```
/**
 * 随机生成滑块的x坐标和y坐标
 */
private void initUnitXY() {
    mUnitRandomX = (int) (Math.random() * (mBitmap.getWidth() - mUintWidth));    //x坐标
    mUnitRandomY = (int) (Math.random() * (mBitmap.getHeight() - mUintHeight)); //y坐标
    //防止生成的位置距离太近
    if (mUnitRandomX <= mBitmap.getWidth() / 2) {
        mUnitRandomX = mUnitRandomX + mBitmap.getWidth() / 4;
    }
    //防止生成的x坐标截图时导致异常
    if (mUnitRandomX + mUintWidth > getWidth()) {
        initUnitXY();                                                    //重新调用该方法
        return;
    }
}
```

（6）当滑块滑动时，判断滑块是否与背景图拼接成功，然后在主界面进行接口回调，关键代码如下：

```
/**
 * 验证是否拼接成功
 */
public void testPuzzle() {
    if (Math.abs(mUnitMoveDistance - mUnitRandomX) <= DEFAULT_DEVIATE) {
        if (null != mlistener) {                                         //判断监听是否为空
            mlistener.onSuccess();                                       //拼接成功
        }
    } else {
        if (null != mlistener) {                                         //判断监听是否为空
```

明日科技

```
            mlistener.onFail();                                    //拼接失败
        }
    }
}
```

（7）在 MainActivity 类中进行滑动验证，如果滑动验证成功，则弹出提示并重置滑动验证，如果验证未成功，则只弹出提示。关键代码如下：

```
/**
 * 滑块滑动时的监听
 **/
verificationView.setPuzzleListener(new SlidingVerificationView.onPuzzleListener() {
    @Override
    public void onSuccess() {
        /**
         * 滑动成功后弹出提示
         **/
        Toast.makeText(MainActivity.this, "验证成功", Toast.LENGTH_SHORT).show();
        mSeekBar.setProgress(0);                                   //使进度条回归起点
        verificationView.reSet();                                  //重置滑动验证
    }
    @Override
    public void onFail() {
        /**
         * 滑动失败时 弹出提示
         **/
        Toast.makeText(MainActivity.this, "验证失败", Toast.LENGTH_SHORT).show();
        mSeekBar.setProgress(0);                                   //使进度条回归起点
    }
});
```

扩展学习

本实例应用的图形是五角星，请尝试在布局中更改图形的样式，可改为圆形或方形。

实例 198　生成本地验证码

应用场景：多次登录页

源码位置：光盘\Code\20\198
视频位置：光盘\Video\20\198

实例说明

在 App 应用中，为防止用户频繁登录，会要求用户输入验证码。本实例将实现生成本地验证码让用户输入。运行实例，初始界面如图 20.16 所示，单击验证码位置，可以更新验证码；输入完验证码

信息，单击“登录”按钮，输入错误弹出如图 20.17 所示的错误信息，输入正确则弹出如图 20.18 所示的正确提示。

图 20.16　初始界面

图 20.17　输入错误提示

图 20.18　输入正确提示

关键技术

关键技术

本实例主要使用 ImageView 类中的 setImageBitmap() 方法、Toast 类中的 makeText() 方法、Bitmap 类中的 createBitmap() 方法、Canvas 类中的 drawText() 方法、drawLine() 方法，实现本地生成验证码并完成验证的功能。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“CodeDemo”，公司域名为“mingrisoft.com”。

（2）导入资源文件：复制“光盘\Code\20\198\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）在“app\java\com\mingrisoft\codedemo”文件夹下创建 Code 类，在该类中创建 createBitmap() 方法，用于生成验证码。代码如下：

```
01  //验证码图片
02  public Bitmap createBitmap() {
03      padding_left = 0;
04      //绘制位图
05      Bitmap bp = Bitmap.createBitmap(width, height, Bitmap.Config.ARGB_8888);
06      //创建画板
07      Canvas c = new Canvas(bp);
08      //绘制新的验证码
09      code = createCode();
```

```
10      //设置背景色
11      c.drawColor(Color.WHITE);
12      //创建画笔
13      Paint paint = new Paint();
14      paint.setAntiAlias(true);
15      //设置字号
16      paint.setTextSize(font_size);
17      //画验证码
18      for (int i = 0; i < code.length(); i++) {
19          randomTextStyle(paint);
20          randomPadding();
21          //绘制显示文字
22          c.drawText(code.charAt(i) + " ", padding_left, padding_top, paint);
23      }
24      //画线条
25      for (int i = 0; i < line_number; i++) {
26          //绘制线条
27          drawLine(c, paint);
28      }
29      c.save(Canvas.ALL_SAVE_FLAG );                    //保存
30      c.restore();
31      return bp;
32  }
```

（4）在“app\java\com\mingrisoft\codedemo”文件夹下默认生成的 MainActivity 类中，设置显示本地生成的验证码。代码如下：

```
01  //初始化显示验证码控件
02  iv_showCode = (ImageView) findViewById(R.id.iv_showCode);
03  //将验证码用图片的形式显示出来
04  iv_showCode.setImageBitmap(Code.getInstance().createBitmap());
05  //获取验证码值
06  realCode = Code.getInstance().getCode().toLowerCase();
```

绑定验证码的单击事件，实现单击后重新绘制新的验证码。代码如下：

```
01  //验证码单击事件
02  case R.id.iv_showCode:
03      //设置新的验证码
04      iv_showCode.setImageBitmap(Code.getInstance().createBitmap());
05      //获取验证码值
06      realCode = Code.getInstance().getCode().toLowerCase();
07      Log.v(TAG,"realCode"+realCode);
08      break;
```

明日科技

绑定“登录”按钮的单击事件，实现单击后验证输入的验证码是否正确。代码如下：

```
//登录按钮单击事件
case R.id.but_forgetpass_toSetCodes:
    //获取验证码值
    String phoneCode = et_phoneCode.getText().toString().toLowerCase();
    //验证码信息
    String msg = "生成的验证码："+realCode+"\n"+"输入的验证码:"+phoneCode;
    //判断验证码是否输入正确
    if (phoneCode.equals(realCode)) {
        Toast.makeText(MainActivity.this, phoneCode + "验证码正确",
                Toast.LENGTH_SHORT).show();
    } else {
        Toast.makeText(MainActivity.this,"验证码错误:"+"\n"+msg,Toast.LENGTH_LONG).show();
    }
```

扩展学习

请尝试在 Code 类中修改验证码显示的文字个数。

明日科技

第21章

启动页合集

实例 199　闪屏页

应用场景：启动手机 App 应用

源码位置：光盘\Code\21\199

实例说明

现在大多数的 App 应用都有闪屏页，即在刚刚进入指定的软件时，并不直接显示主界面，而是显示一个很漂亮的界面或者广告，闪屏页完成之后才进入主界面。本实例将实现 App 应用中的闪屏页界面。运行实例，会随机出现两张图片，如图 21.1 和图 21.2 所示；紧接着图 21.2 会出现自动放大的动画效果，如图 21.3 所示。

图 21.1　随机闪屏页（一）

图 21.2　随机闪屏页（二）

图 21.3　闪屏页实现放大效果

关键技术

本实例主要使用 Animation 类中的 loadAnimation() 方法设置图片的动画，使用 Random 类中的 nextInt() 方法实现随机获取闪屏界面图片的功能。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“Splash”，公司域名为“mingrisoft.com”。

（2）在“app\res”文件夹中创建 anim 文件夹，在 anim 文件中创建 enlarge 文件，该文件用于存放大闪屏界面的图片。

（3）在“app\res\values”文件夹中创建 arrays.xml 文件，用于加载闪屏界面的随机显示图片。

（4）修改显示界面的主题样式，去除系统默认添加的 ActionBar 标题栏，方法是打开“app\res\values”文件夹中的 styles.xml 文件，将“DarkActionBar”改为“NoActionBar”。

（5）创建 SlapActivity 类，继承自 AppCompatActivity，此界面为闪屏界面。

（6）从资源文件中取出加载的数组，代码如下：

```
TypedArray ar =getResources().obtainTypedArray(R.array.imgArray);//获取图片数组
int len = ar.length();                                            //获取数组的长度
resids = new int[len];                                            //初始化加载图片的数组
for (int i = 0; i < len; i++)
    resids[i] = ar.getResourceId(i, 0);                           //使用for循环把图片读取出来
```

（7）利用 Random 随机取出一张图片并显示出来，代码如下：

```
Random rand = new Random();                                       //初始化随机数
int pos = rand.nextInt(resids.length);                            //随机获取出小于数组长度的数
imageView.setImageResource(resids[pos]);                          //把图片显示出来
```

（8）由于图片的放大操作是耗时操作，所以开启一个线程，在线程中执行图片的放大操作。关键代码如下：

```
/**
 * 开启线程
 **/
new Thread(new Runnable() {
    @Override
    public void run() {
        try {
            imageView.startAnimation(animation);                  //给图片设置放大的动画
            Thread.sleep(5000);                                   //此界面休眠5秒
        } catch (InterruptedException e) {
            e.printStackTrace();
        }
        toSecond();                                               //执行跳转到下一个界面的方法
    }
}).start();
```

扩展学习

该程序在闪屏界面停留 5 秒之后进入主界面，尝试让闪屏界面停留 3 秒后进入主界面。

实例 200　新手引导

应用场景：引用 App 新用户操作

源码位置：光盘\Code\21\200
视频位置：光盘\Video\21\200

实例说明

视频讲解

有些 App 应用在用户第一次使用或者更新功能后，会引导用户进行相关的操作。本实例将实现一个新手引导，为用户展示一些相关的信息。运行实例，将显示如图 21.4 所示的初始界面；单击“知道了”按钮，进入第二个展示页，如图 21.5 所示；单击“知道了”，进入程序主界面，如图 21.6 所示。

图 21.4 展示页（一）

图 21.5 展示页（二）

图 21.6 主界面

关键技术

关键技术

本实例主要使用 View 类中的 invalidate() 方法、Paint 类中的 setXfermode() 方法、Display 类中的 getWidth() 方法，实现刷新视图及将重叠的部分设置为透明的功能。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“NoviceBoot”，公司域名为“mingrisoft.com”。
（2）将相关的图片资源导入到“res\mipmap”文件夹中，作为界面背景。

（3）在 MainActivity 类中获取屏幕的宽度与高度，用于设置覆盖在屏幕上方的半透明矩形的大小，关键代码如下：

```
/**
 * 获取屏幕的宽高
 **/
WindowManager wm = (WindowManager) this.getSystemService(Context.WINDOW_SERVICE);
w = wm.getDefaultDisplay().getWidth();                    //获取屏幕宽度
h = wm.getDefaultDisplay().getHeight();                   //获取屏幕高度
```

（4）设置覆盖在屏幕上方视图的透明度（注：透明度取值 0-1），然后将屏幕的宽与高传递给画布，关键代码如下：

```
v3.setAlpha((float) 0.7);                                 //设置覆盖最上方的View的透明度
v3.getWindowWH(w, h);                                     //将手机屏幕的宽高传给画布
```

（5）在自定义的 TopView 类中添加接收屏幕宽与高的方法，并重新绘制视图，关键代码如下：

```
/**
 * 根据屏幕的宽高设置矩形的大小
 **/
public void getWindowWH(int w, int h) {
    this.h = h;
    this.w = w;
    invalidate();                                         //重新绘制
}
```

（6）在 TopView 类中绘制圆形与矩形，使重合的部分变成透明，再将文字的位置及大小添加到指定的位置，关键代码如下：

```
/**
 * 设置重叠部分透明
 **/
canvas.drawCircle(s, d, 100, pCircle)                     //画圆
pMath.setXfermode(new PorterDuffXfermode(PorterDuff.Mode.XOR));  //设置重叠透明
canvas.drawRect(0, 0, w, h, pMath);                       //画矩形
pText.setTextSize(50);                                    //绘制提示文字的大小
canvas.drawText(text, x, y, pText);                       //绘制提示文字
```

（7）给覆盖在屏幕上方的按钮添加判断的监听，本实例一共提示两次，当判断为最后一次提示时，将按钮与 View 类隐藏，判断的关键代码如下：

```
/**
 * 进行判断
 **/
if (i == 1) {
    /**
```

```
06          * 如果提示信息展示完成，使按钮隐藏
07          **/
08         v3.setVisibility(View.INVISIBLE);
09         nextBtn.setVisibility(View.INVISIBLE);
10         break;
11     }
```

扩展学习

本实例为每次进入后都会显示提示信息，请尝试实现在第一次进入时显示用户信息，再次进入时不再显示相关信息的功能。

实例 201	炫酷引导页 应用场景：App 引导界面	源码位置：光盘\Code\21\201

实例说明

当 App 应用安装完成后，首次打开时会出现引导界面，有些引导界面是介绍 App 应用的功能，有些引导界面是帮助用户使用 App。本实例将完成一个炫酷的 App 引导页。运行实例，打开如图 21.7 所示的初始界面；向左滑动显示第二个界面，如图21.8所示；接着向左滑动可显示第三个和第四个界面，第四个界面如图 21.9 所示，该界面中有一个进入主界面的按钮。

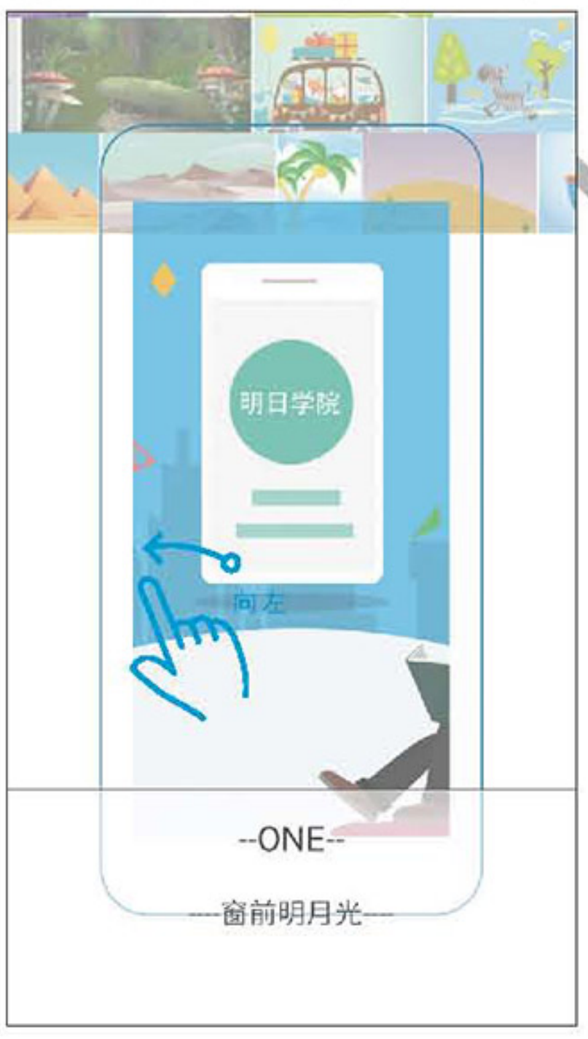


图 21.7　初始界面效果

图 21.8　滑动中的界面

图 21.9　最后一个界面

关键技术

本实例主要使用 View 类中的 setAlpha() 方法、setTranslation() 方法、setScaleY() 方法、setScaleX() 方法，实现设置图片的移动、缩放等动画的功能。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“GuidePageTwo”，公司域名为“mingrisoft.com”。

（2）在“app\res\mipmap”文件夹中粘贴相关的图片，作为引导界面的图片动画。

（3）创建 DensityUtil 类文件，用于将 dp 与 px 进行相互转换。

（4）在 MainActivity 类中为了界面显示的美观度，需要将界面设置成全屏显示，代码如下：

```
/**
 * 设置界面全屏
 **/
private void setFullscreen() {
    View screen = getWindow().getDecorView();
    int manager = View.SYSTEM_UI_FLAG_HIDE_NAVIGATION
            | View.SYSTEM_UI_FLAG_LAYOUT_FULLSCREEN
            | View.SYSTEM_UI_FLAG_LAYOUT_HIDE_NAVIGATION
            | View.SYSTEM_UI_FLAG_FULLSCREEN
            | View.SYSTEM_UI_FLAG_IMMERSIVE_STICKY;
    screen.setSystemUiVisibility(manager);
}
```

（5）将屏幕设置全屏之后，将添加到 ViewPager 中的 Fragment 进行初始化，之后还需要将初始化的 Fragmen 添加到 List 集合中，关键代码如下：

```
oneFragment = new OneFragment();                    //初始化第一个界面
twoFragment = new TwoFragment();                    //初始化第二个界面
threeFragment = new ThreeFragment();                //初始化第三个界面
fourFragment = new FourFragment();                  //初始化第四个界面
fragmentList.add(oneFragment);                      //将第一个界面添加到集合中
fragmentList.add(twoFragment);                      //将第二个界面添加到集合中
fragmentList.add(threeFragment);                    //将第三个界面添加到集合中
fragmentList.add(fourFragment);                     //将第四个界面添加到集合中
```

（6）在 MainActivity 类中为 ViewPager 编写适配器。

（7）在视图进行滑动时，每个视图对应的图片会进行相应的动画，当第一个界面滑动到第二个界面的时候图片缩小，底部的图片显示出来。关键代码如下：

```
//第一页面到第二个页面的动画
iv_initial_phone.setTranslationY(-600 * f);         //设置图片的位置
iv_initial_phone.setScaleX(0.5f * p + 0.5f);        //设置图片x的缩放比例
iv_initial_phone.setScaleY(0.5f * p + 0.5f);        //设置图片y的缩放比例
iv_initial_phone.setAlpha(200);                     //设置图片的透明度
```

明日科技

扩展学习

本实例中单击“请注册账号”后，动画将从底部升起，请尝试实现注册按钮的渐变显示功能。

实例 202	滑动隐藏引导页 应用场景：首次安装 App 时	源码位置：光盘\Code\21\202

实例说明

大多数 App 应用在首次运行时，会出现引导页，有些 App 为动态图，有些为动画，用来介绍软件的使用方法。本实例将实现引导页的动画。运行实例，将显示如图 21.10~21.13 所示的界面。

图 21.10　引导页（一）

图 21.11　引导页（二）

图 21.12　引导页（三）

图 21.13　引导页（四）

关键技术

本实例主要使用 View 类中的 setVisibility() 方法、setBackgroundResource() 方法，实现设置控件的隐藏、显示及设置界面背景的功能。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“GuidePage”，公司域名为“mingrisoft.com”。
（2）将相关的图片资源粘贴到“res\drawable”文件夹中，用作引导页的背景图片。
（3）在“app\res\values”文件夹中创建 attrs.xml 文件，用于设置圆点指示器的自定义属性。
（4）创建 GuidePageActivity 引导界面，在绑定布局之前设置界面全屏。代码如下：

```
getWindow().setFlags(WindowManager.LayoutParams.FLAG_FULLSCREEN,
                                        WindowManager.LayoutParams.FLAG_FULLSCREEN);
```

（5）在 GuidePageActivity 类中判断 App 是不是第一次加载，使用轻量级数据库进行判断，关键代码如下：

```
/**
 * 初始化轻量级数据库与编辑器
 **/
sharedPreferences = getSharedPreferences("guideTF", CONTEXT_RESTRICTED);
editor = sharedPreferences.edit();
/**
 * 判断是否为第一次登录，如果不是直接跳转到闪屏界面
 **/
if (sharedPreferences.getBoolean("guideTF", false)) {
    Intent intent = new Intent(this, MainActivity.class);
    startActivity(intent);
    finish();
}
```

（6）为引导页 GuidePageActivity 类绘制布局文件。

（7）初始化数据，目的是将图片传递给适配器，代码如下：

```
/**
 * 初始化数据
 */
private void initData() {
    pageList = new ArrayList<>();                          //初始化数组
    for (int i = 0; i < 4; i++) {
        View page = getLayoutInflater().inflate(R.layout.guide_page_item, null);
        page.setBackgroundResource(fontImages[i]);         //将图片设置到背景中
        pageList.add(page);                                //将View类添加到list中
    }
    guideAdapter = new GuidePageAdapter(pageList);         //将数据传给适配器
}
```

（8）由于用到了 ViewPager，所以需要设置适配器，并将数据传递给适配器，适配器中接收数据方法的代码如下：

```
/**
 * 构造方法
 * @param pageList 传入将要显示的控件的集合
 */
public GuidePageAdapter(ArrayList<View> pageList) {
    this.pageList = pageList;
}
```

扩展学习

请尝试实现按钮进入的效果为侧方进入。

实例 203　滑动旋转引导页

应用场景：用于首次进入手机 App

源码位置：光盘\Code\21\203

视频位置：光盘\Video\21\203

实例说明

手机 App 应用的引导页有很多样式，本实例将介绍如何实现引导页的炫酷动画。运行实例，第一张图片切换到第二张图片的动画（图片转动消失），如图 21.14 所示；第二张图片切换到第三张图片的动画（图片从手机上方移出），如图 21.15 所示；第三张图片切换到第四张图片的动画（图片从手机下方移出），如图 21.16 所示，从而实现引导页炫酷的动画效果。

图 21.14　第一张图片切换

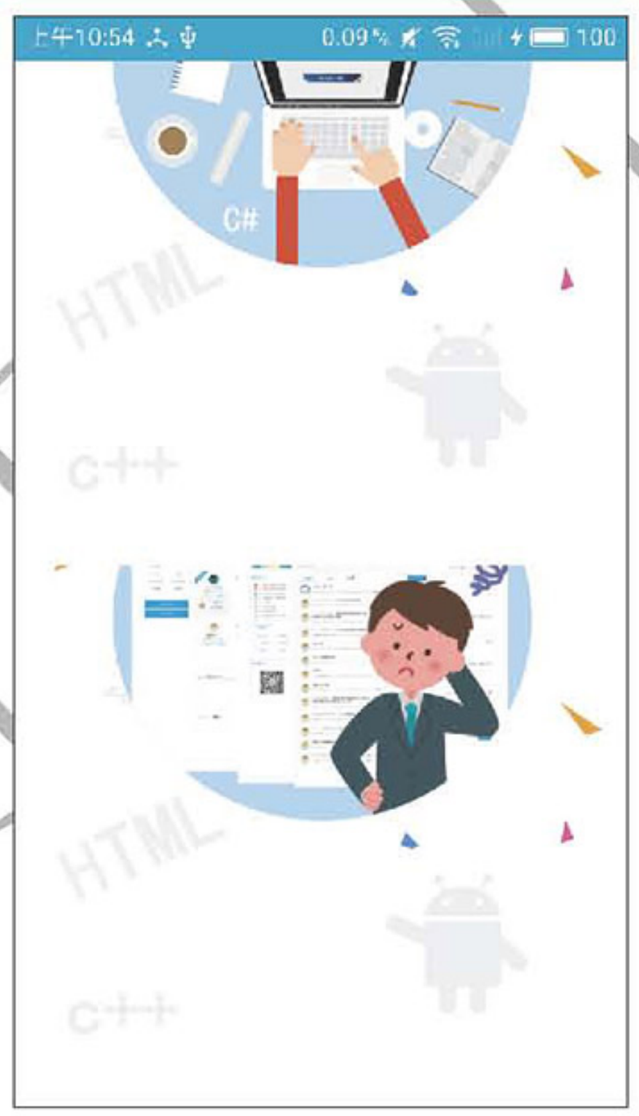


图 21.15　第二张图片切换

图 21.16　第三张图片切换

关键技术

本实例主要使用 ScrollView 类中的 scrollTo() 方法，View 类中的 setAlpha () 方法、setRotation() 方法，设置图片的滚动、透明度的变化及旋转。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“BootPageNew”，公司域名为“mingrisoft.com”。

（2）在“app\res”文件夹的 mipmap 文件夹中拷入相关的图片资源，用于设置引导页的图片。

（3）创建 MyPagerAdapter 类，继承 PagerAdapter，用于设置 ViewPager 的适配器。

（4）在 MainActivity 类中获取屏幕的宽度、高度，并设置透明图片的高度与手机屏幕的高度一致，关键代码如下：

```
/**
 * 获取屏幕的宽、高度
 **/
DisplayMetrics metric = new DisplayMetrics();
getWindowManager().getDefaultDisplay().getMetrics(metric);
width = metric.widthPixels;                          //屏幕宽度（像素）
height = metric.heightPixels;                        //屏幕高度（像素）
```

（5）给 ViewPager 绑定适配器，并设置图片的透明度，为图片设置透明之后将获取到的手机屏幕高度设置给透明的图片，关键代码如下：

```
adapter = new MyPagerAdapter(list);                  //初始化适配器
viewPager.setAdapter(adapter);                       //ViewPager绑定适配器
nullView.setAlpha(0);                                //设置图片完全透明
nullViewTwo.setAlpha(0);                             //设置图片完全透明
nullView.setMinimumHeight(height);                   //给第二张透明的图片设置高度
nullViewTwo.setMinimumHeight(height);                //给第三张透明的图片设置高度
twoView.setMinimumHeight(height);                    //给第二张图片设置高度
threeView.setMinimumHeight(height);                  //给第三张图片设置高度
```

（6）根据 ViewPager 的属性，获取滑动到不同位置时的偏移量，并设置图片的透明度、转动的角度、ScrollView 滚动的距离。关键代码如下：

```
case 0:                                              //当ViewPager滑动到第一页的时候
    oneView.setRotation(positionOffset * 360);       //设置第一张图片的旋转的角度
    oneView.setAlpha(1 - positionOffset);            //设置第一张图片的透明度
    break;
case 1:                                              //当ViewPager滑动到第二页的时候
    scrollViewTwo.scrollTo(0, 2158);                 //设置第二张图片滑动的距离
    scrollView.scrollTo(0, positionOffsetPixels * 2);
    break;
case 2:                                              //当ViewPager滑动到第三页的时候
    scrollViewTwo.scrollTo(0, 2158 - (positionOffsetPixels * 2)); //设置第三张图片滑动的距离
    break;
```

扩展学习

本实例中引导界面中的第二张图片向上滑动移出，第三张图片是向下滑动移出，请尝试将两张图片的移出效果对调一下。

第22章

加载等待合集

实例204 加载等待条

应用场景：App加载网络数据

源码位置：光盘\Code\22\204

实例说明

在进行网络请求或者访问较大的数据库时，会有一段的等待期，为了提高用户体验效果，需要在等待期间添加等待动画。运行实例，图22.1为正在加载的界面，图22.2为加载成功后的界面。

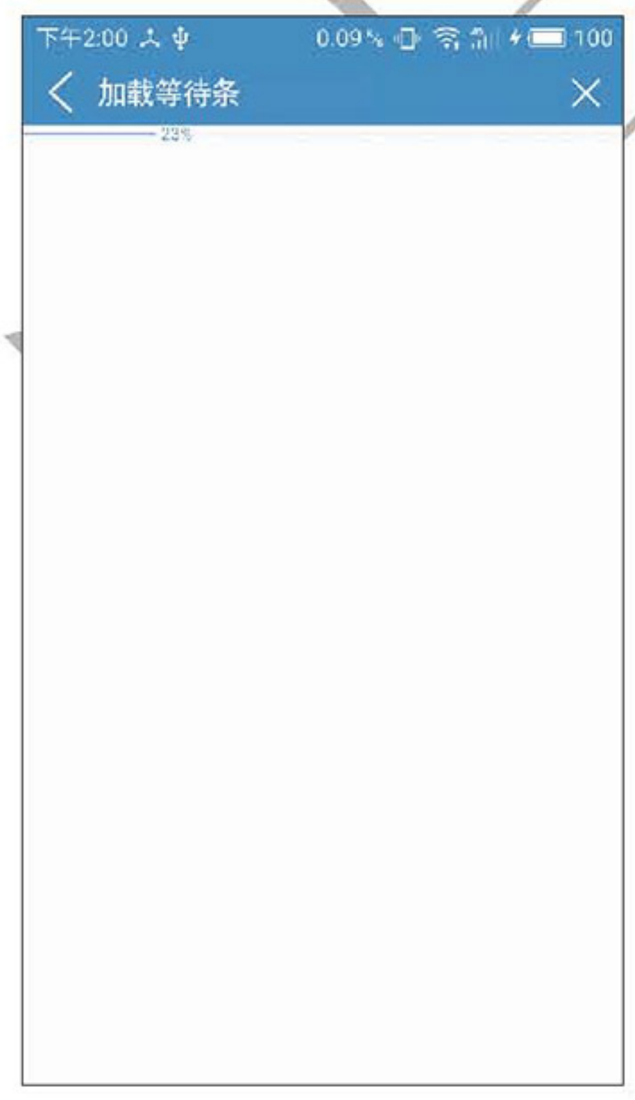


图22.1 加载中的界面

图22.2 加载成功后的界面

关键技术

本实例主要使用 Canvas 类的 drawRect() 方法、drawText() 方法、View 类的 setVisibility() 方法，实现绘制加载等待条与加载进度的功能。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“NumberProgressBar”，公司域名为“com.mingrisoft”。

（2）在“app\res\mipmap”文件夹中导入相关的图片资源。

（3）修改显示界面的主题样式，去除系统默认添加的 ActionBar 标题栏，方法是打开“app\res\values”文件夹中的 styles.xml 文件，将“DarkActionBar”修改为“NoActionBar”。

（4）创建数字进度条的类文件，命名为“NumberProgressBar”，在该类中定义一些变量及控件，关键代码如下：

```
01  //目前的进展，不能超过最大进展
02  private int mCurrentProgress = 0;
03  //进度栏颜色
04  private int mReachedBarColor;
05  //进度条的颜色
06  private int mUnreachedBarColor;
07  //进度条文字的颜色
08  private int mTextColor;
09  //文字的大小
10  private float mTextSize;
11  //进度条的高度
12  private float mReachedBarHeight;
13  //进度条空白区域的面积
14  private float mUnreachedBarHeight;
15  //数后边的百分号
16  private String mSuffix = "%";
17  //数的前缀
18  private String mPrefix = "";
```

（5）进度条是由三支画笔绘制的，分别为绘制已加载的绿色进度条、未加载的灰色进度条以及加载进度的百分比。初始化三支画笔的代码如下：

```
01  /**
02   * 初始化画笔
03   **/
04  private void initializePainters() {
05      /**
06       * 初始化已经加载完成的绿色部分的画笔
07       **/
08      mReachedBarPaint = new Paint(Paint.ANTI_ALIAS_FLAG);          //抗锯齿
```

明日科技

```
09      mReachedBarPaint.setColor(mReachedBarColor);                    //设置画笔颜色
10      /**
11       * 初始化未加载的灰色部分的画笔
12       **/
13      mUnreachedBarPaint = new Paint(Paint.ANTI_ALIAS_FLAG);          //抗锯齿
14      mUnreachedBarPaint.setColor(mUnreachedBarColor);                //设置画笔颜色
15      /**
16       * 绘制文字的画笔
17       **/
18      mTextPaint = new Paint(Paint.ANTI_ALIAS_FLAG);                  //抗锯齿
19      mTextPaint.setColor(mTextColor);                                //设置颜色
20      mTextPaint.setTextSize(mTextSize);                              //设置文字的大小
21  }
```

（6）由于此项目没有后台的数据，所以用计时器模拟网络数据的加载，在计时器中每隔 0.1 秒便给进度条加一，关键代码如下：

```
01  /**
02   * 开启计时器的方法
03   **/
04  private void startTimer() {
05      /**
06       * 计时器
07       * 1000表示这个计时器最先打开后延迟1秒执行
08       * 100表示每隔0.1秒执行一次
09       **/
10      timer.schedule(new TimerTask() {
11          @Override
12          public void run() {
13              runOnUiThread(new Runnable() {
14                  @Override
15                  public void run() {
16                      bnp.incrementProgressBy(1);                     //每次给进度+1
17                  }
18              });
19          }
20      }, 1000, 100);
21  }
```

（7）开启计时之后，在界面销毁时关闭计时器，代码如下：

```
01  @Override
02  protected void onDestroy() {
03      super.onDestroy();
04      timer.cancel();                                                 //结束计时
05  }
```

扩展学习

本实例中加载等待条显示百分比，请尝试不显示进度条中的文字信息。

实例 205	小球加载等待 应用场景：加载数据进行等待	源码位置：光盘\Code\22\205

实例说明

当 App 应用与后台的数据交互时，需要长时间的进行数据解析，在等待的这段时间内可以在界面中添加一个加载动画。本实例将介绍如何实现加载等待的动画。运行实例，显示加载等待的动画，如图 22.3~22.5 所示。

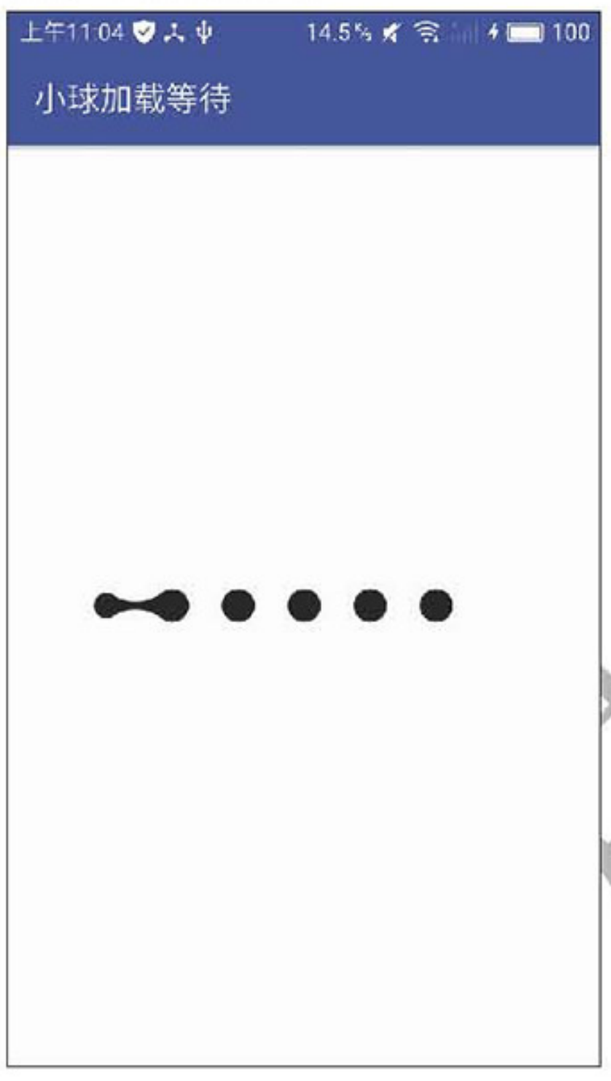


图 22.3　加载等待效果（一）

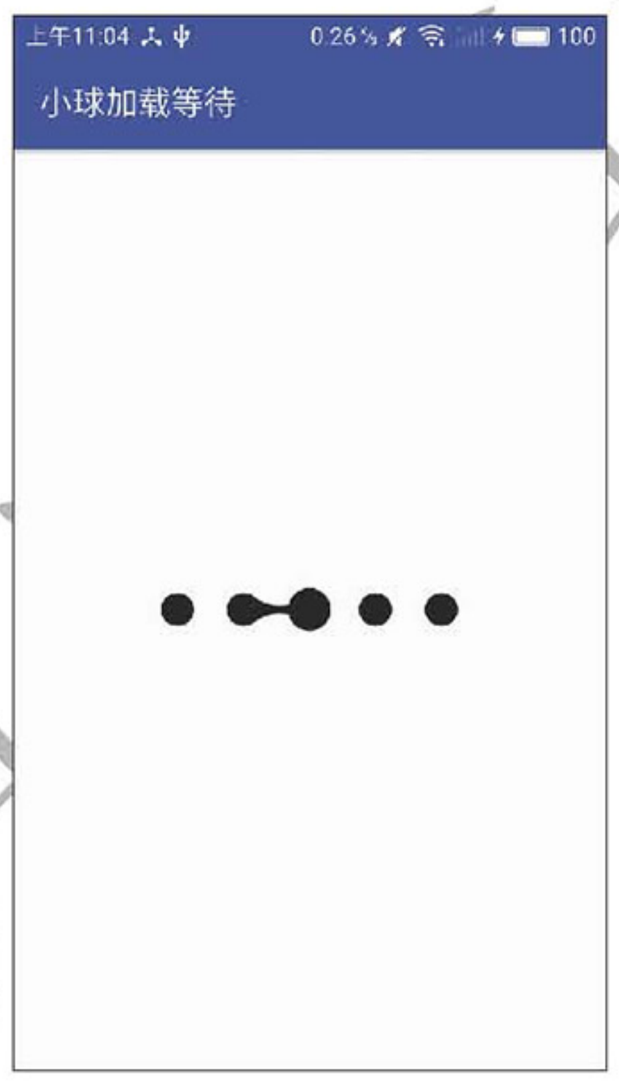


图 22.4　加载等待效果（二）

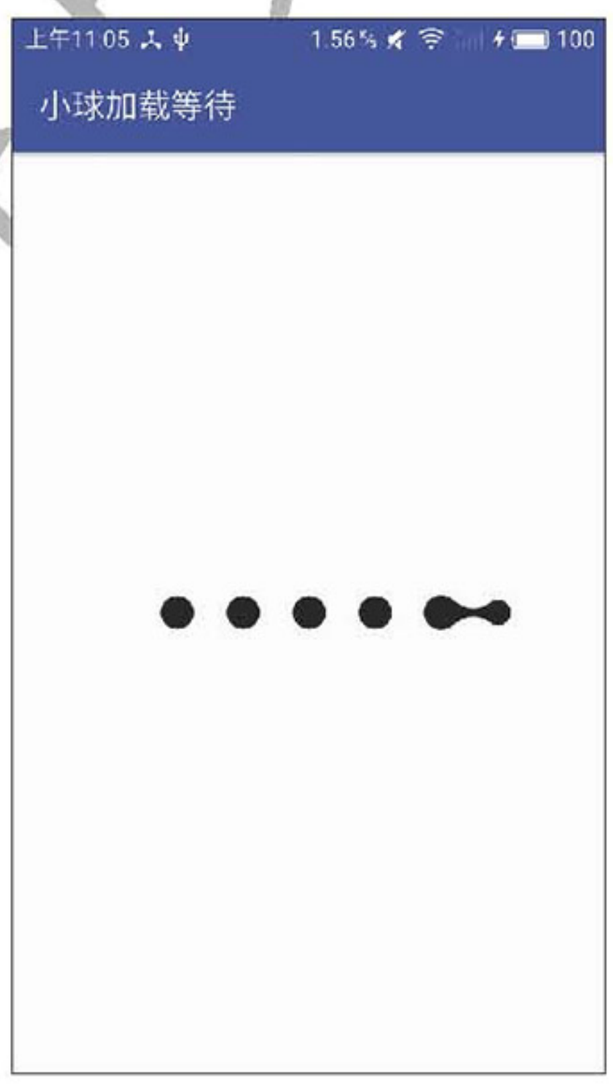


图 22.5　加载等待效果（三）

关键技术

关键技术

本实例主要使用 Math 类中的 cos() 方法、sin () 方法、acos() 方法和 sqrt() 方法，实现绘制贝塞尔曲线。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“LoadingView”，公司域名为“mingrisoft.com”。

（2）创建 MetaballView 类，主要用于显示加载等待的动画，在该类中需要声明动画的路径集合。关键代码如下：

```
private ArrayList<Circle> circlePaths = new ArrayList<>();          //圆点路径的集合
```

（3）在 MetaballView 类中添加一个方法，用于设置动画的风格，判断是否需要填充效果，关键代码如下：

```
01  /**
02   * 主界面使用此方法设置动画的样式
03   **/
04  public void setPaintMode(int mode) {
05      //如果主界面传过来的是0，则显示动画无填充效果
06      //否则动画有填充效果
07      paint.setStyle(mode == 0 ? Paint.Style.STROKE : Paint.Style.FILL);
08  }
```

（4）当 App 应用进行界面的加载时，开启小圆形来回波动的动画，实现动画的相关属性及动画效果的关键代码如下：

```
01  private void startAnimation() {
02      wa = new MoveAnimation();                                    //初始化移动的动画
03      wa.setDuration(2500);                                        //动画时间2.5秒
04      wa.setInterpolator(new AccelerateDecelerateInterpolator());  //设置不同速度的补值
05      wa.setRepeatCount(Animation.INFINITE);                       //设置动画的重复
06      wa.setRepeatMode(Animation.REVERSE);                         //设置动画的重复模式
07      startAnimation(wa);                                          //开始动画
08  }
```

（5）在布局文件中引用刚刚创建好的 MetaballView 布局，代码如下：

```
01  <com.mingrisoft.loadingview.MetaballView
02      android:id="@+id/meta_ball"
03      android:layout_width="wrap_content"
04      android:layout_height="wrap_content"
05      android:layout_centerInParent="true" />
```

（6）在 MainActivity 类中直接调用 LoadingView 类就可以。

扩展学习

本实例中加载动画中的小球为黑色，请尝试使其变为绿色。

明日科技

实例 206 图形变换加载等待

应用场景：数据加载

源码位置：光盘\Code\22\206

实例说明

App 应用进行后台大数据加载或者遍历手机本地的数据等耗时的操作时，添加一个炫酷的加载等待效果，可以提高加载等待的美观度。本实例将介绍一个加载等待的动画。运行实例，将显示加载等待的动画，如图 22.6~22.8 所示。

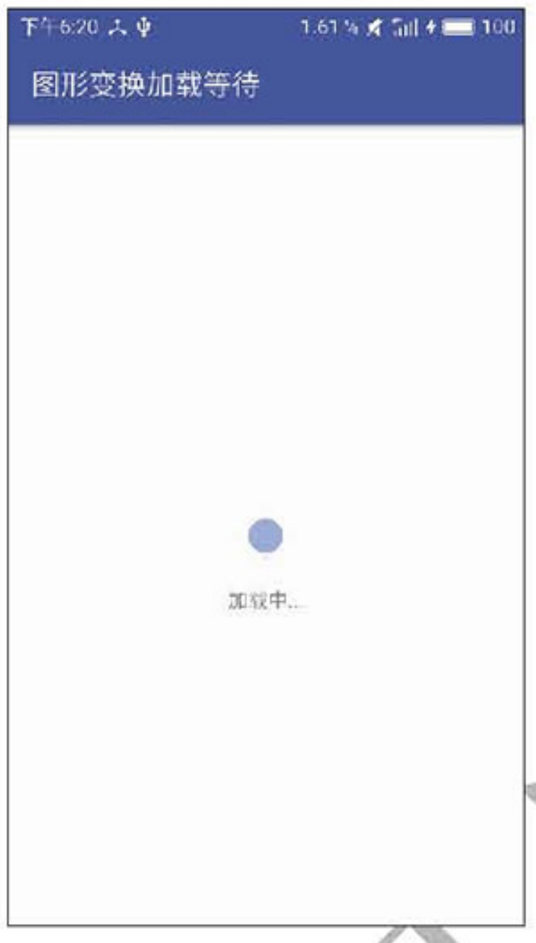


图 22.6 加载等待效果（一）

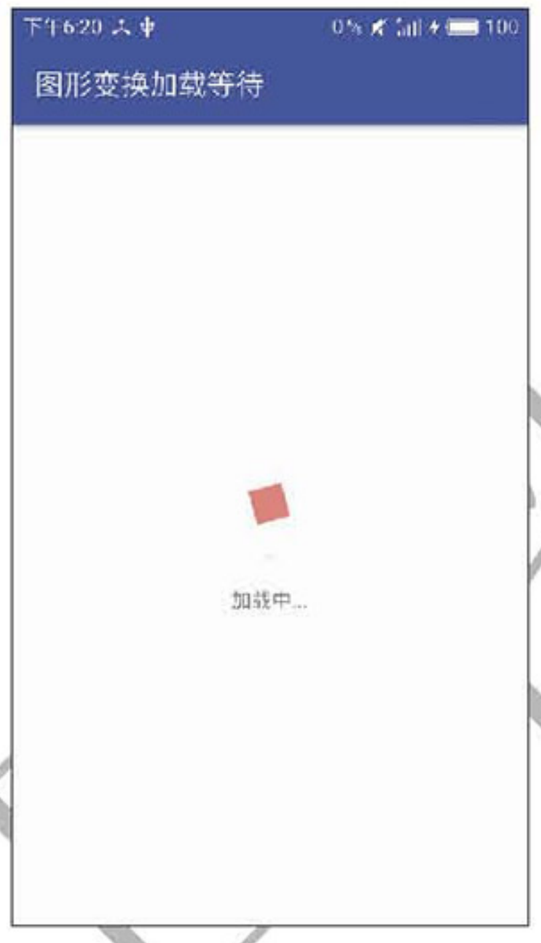


图 22.7 加载等待效果（二）

图 22.8 加载等待效果（三）

关键技术

本实例主要使用 Path 类中的 moveTo() 方法、cubicTo () 方法、lineTo() 方法、quadTo() 方法，设置动画效果。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“LoadingView”，公司域名为“mingrisoft.com”。

（2）在“app\res\values”文件夹中创建加载动画的颜色与属性类。

（3）修改显示界面的主题样式，去除系统默认添加的 ActionBar 标题栏，方法是打开“app\res\values”文件夹中的 styles.xml 文件，将“DarkActionBar”修改为“NoActionBar”。

（4）创建 ShapeLoad 类，用于设置加载时的动画，在该类中设置构造方法，用于布局界面的调用。

（5）加载等待动画执行时，为了使界面美观大气，在图形的下方显示阴影，阴影的面积随图形位置的变化而变化，关键代码如下：

```
01  /**
02   * 当图形进行上抛时，为使界面有立体感，设置图形下方的阴影变换的动画
03   **/
04  ObjectAnimator scaleIndication = ObjectAnimator.ofFloat(mIndicationIm, "scaleX", 0.2f, 1);
```

（6）以绘制矩形为例，使用了 Path 类中的 moveTo() 方法与 lineTo() 方法，最后根据路径与画笔的颜色绘制出图形，关键代码如下：

```
01  Path path = new Path();                                          //初始化路径
02  /**
03   * 根据path.lineTo()方法与Path.moveTo()方法绘制矩形
04   **/
05  path.moveTo(relativeXFromView(0f), relativeYFromView(0f));
06  path.lineTo(relativeXFromView(1f), relativeYFromView(0f));
07  path.lineTo(relativeXFromView(1f), relativeYFromView(1f));
08  path.lineTo(relativeXFromView(0f), relativeYFromView(1f));
09  path.close();                                                    //关闭路径
10  mAnimPercent = 0;
11  canvas.drawPath(path, mPaint);                                   //在画布上根据路径与画笔绘制出图像
```

（7）在布局文件中绘制出刚刚创建好的 LoadingView 布局，代码如下：

```
01  <com.mingrisoft.smsforge.LoadingView
02      android:id="@+id/loadView"
03      android:layout_width="wrap_content"
04      android:layout_height="wrap_content"
05      android:layout_centerInParent="true"
06      app:loadingText="加载中..." />
```

（8）在 ViewActivity 类中直接调用图形切换的界面。

扩展学习

本实例中加载的动画为矩形、圆形、三角形互相切换，请尝试修改为只有圆形上下跳的动画效果。

实例 207　加载等待弹窗

应用场景：加载界面效果

源码位置：光盘\Code\22\207

实例说明

本实例中要实现一个加载等待弹窗的效果。运行实例，将加载初始界面，如图 22.9 所示；在单击“测试”按钮后会弹出弹窗，随后在弹窗中播放加载等待的动画，如图 22.10 所示。

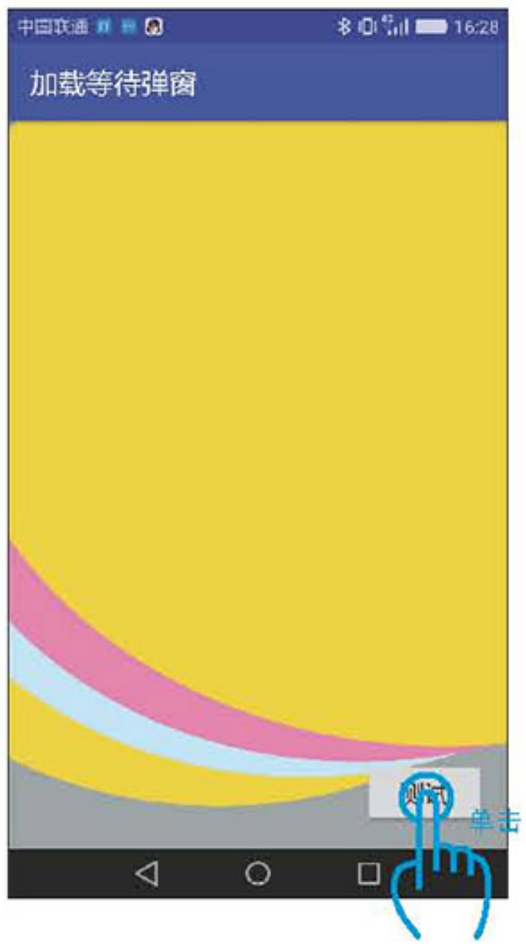


图 22.9　界面的默认效果

图 22.10　菜单打开的效果

关键技术

本实例中主要使用 Dialog，通过自定义 Dialog 来实现加载等待弹窗的效果。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“LoadingDialog”，公司域名为“mingrisoft.com”。

（2）在 styles.xml 文件中设置 Dialog 的效果。代码如下：

代码位置：LoadingDialog\app\src\main\res\values\styles.xml

```
<style name="MyDialog" parent="android:style/Theme.Dialog">
    <!--背景颜色和透明度-->
    <item name="android:windowBackground">@android:color/transparent</item>
    <!--是否去除标题 -->
    <item name="android:windowNoTitle">true</item>
    <!--是否去除边框-->
    <item name="android:windowFrame">@null</item>
    <!--是否浮现在activity之上-->
    <item name="android:windowIsFloating">true</item>
    <!--是否模糊-->
    <item name="android:backgroundDimEnabled">false</item>
</style>
```

（3）创建 LoadingDialog 类，在 onCreate() 方法中设置 Dialog 的布局效果，并获取 AnimationDrawable 对象，用来管理加载动画的播放和停止。代码如下：

代码位置：LoadingDialog\app\src\main\java\com\mingrisoft\loadingdialog\LoadingDialog.java

```
setCanceledOnTouchOutside(false);                    //单击外侧不关闭Dialog
setContentView(R.layout.loading_dialog);             //加载布局
//获取动画
animation = (AnimationDrawable) ((ImageView)
        findViewById(R.id.loading)).getDrawable();
```

根据 LoadingDialog 的生命周期控制动画的播放和停止。代码如下：

```
@Override
protected void onStart() {
    super.onStart();
    Log.i("LoadingDialog","onStart: ");
    animation.start();                               //播放动画
}

@Override
protected void onStop() {
    super.onStop();
    Log.w("LoadingDialog","onStop: ");
    animation.stop();                                //停止动画
}
```

（4）在 MainActivity 类中调用加载等待弹窗的界面。（代码位置：LoadingDialog\app\src\main\java\com\mingrisoft\loadingdialog\MainActivity.java）

扩展学习

请尝试更改弹窗的背景颜色。

实例 208　加载、错误和完成三种动画

应用场景：加载效果

源码位置：光盘\Code\22\208

实例说明

加载、错误和完成三种动画是通过同步“net.steamcrafted:load-toast:1.0.8”依赖代码所实现的运行效果。运行实例，如图 22.11 所示，单击界面中的“加载”按钮，将显示加载进度的进度条动画；单击“错误”按钮，将显示如图 22.12 所示的错误提示动画；单击“完成”按钮，将显示如图 22.13 所示的完成进度加载的提示动画。

明日科技

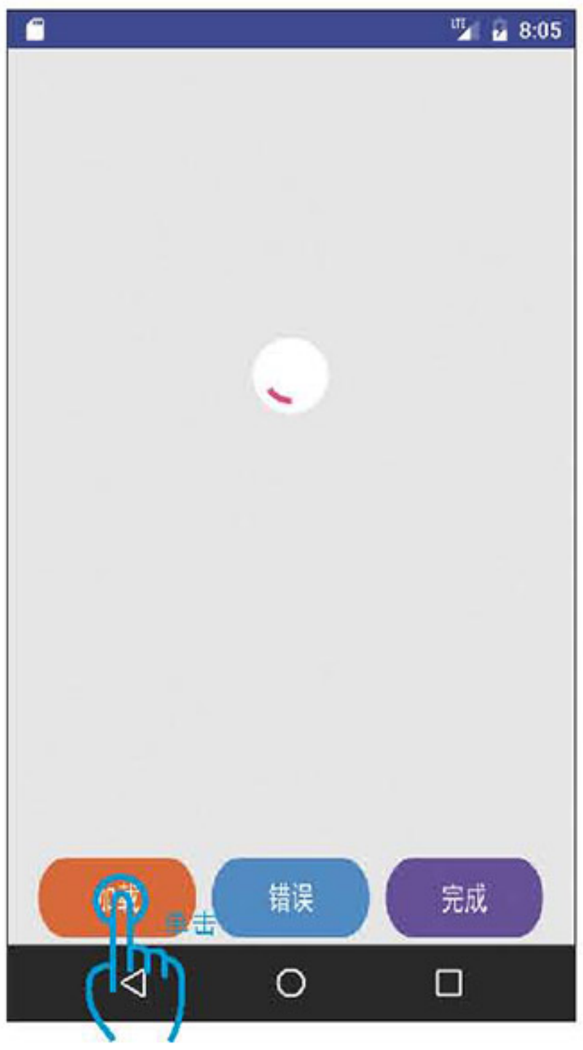


图 22.11　加载进度的进度条动画

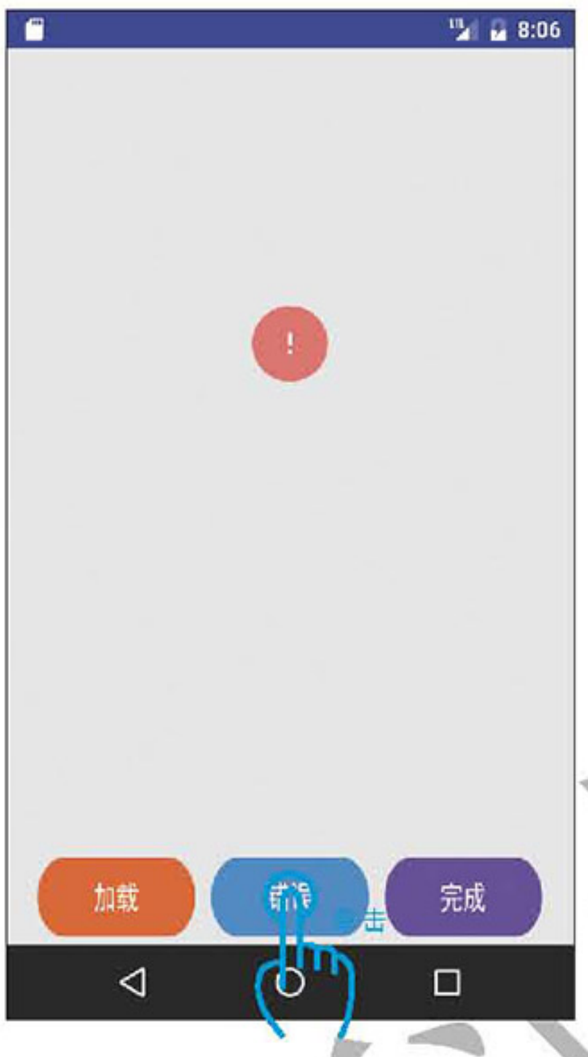


图 22.12　错误提示动画

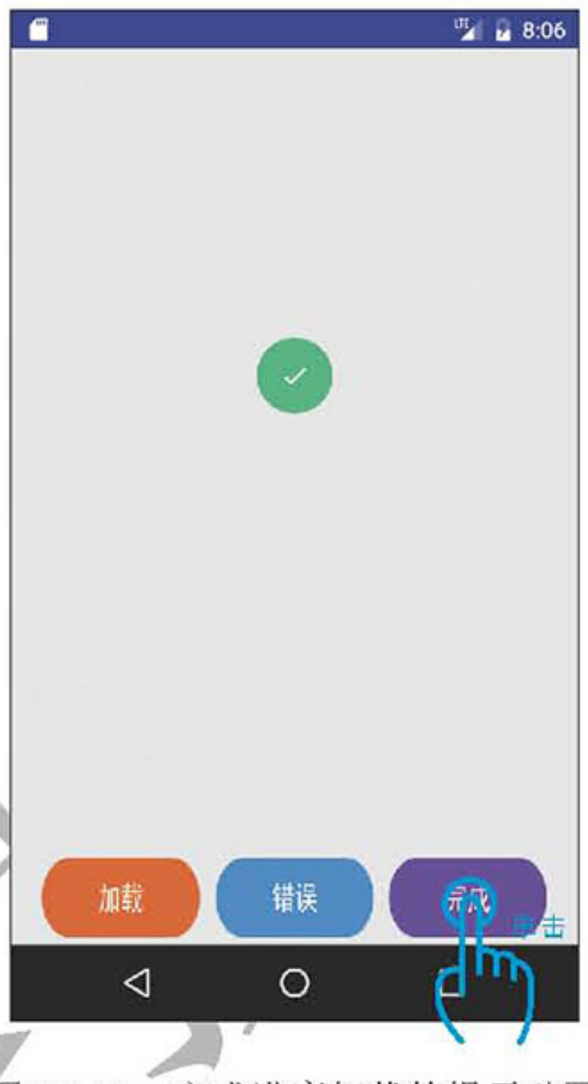


图 22.13　完成进度加载的提示动画

关键技术

本实例主要使用 LoadToastView 类中的 show() 方法、ViewHelper 类中的 setTranslationX() 方法、ViewPropertyAnimator 类中的 animate() 方法，实现加载、错误和完成三种动画的效果。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“LoadToast”，公司域名为“mingrisoft.com”。

（2）导入资源文件：复制“光盘\Code\22\208\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）打开“Gradle Scripts”目录中的 build.gradle(Module:app) 文件，在该文件中添加依赖代码并单击窗口顶部的“Sync Now”进行同步。代码如下：

```
compile 'net.steamcrafted:load-toast:1.0.8'
```

（4）在“app\src\main\java\com.mingrisoft.loadtoast”目录中创建 LoadToast 类，在该类中通过 show() 自定义方法，实现显示加载、错误和完成动画效果。代码如下：

```
01  public LoadToast show() {
02      loadToastView.show();                                    //显示加载进度的控件
03      //设置Toast显示在屏幕中间的位置
04      ViewHelper.setTranslationX(loadToastView, (parentView.getWidth() - loadToastView.getWidth()) / 2);
05      //设置Toast位置显示的y坐标
```

```
06      ViewHelper.setTranslationY(loadToastView, -loadToastView.getHeight() + mTranslationY);
07      //设置显示Toast进入的动画
08      ViewPropertyAnimator.animate(loadToastView)
09              .alpha(1f)                                          //设置透明度为完全显示
10              .translationY(50 + mTranslationY)                   //设置Toast进入动画滑动的距离
11              .setInterpolator(new DecelerateInterpolator())      //设置动画减速
12              .setDuration(300)                                   //持续的时间
13              .setStartDelay(0).start();                          //启动动画
14      return this;
15  }
```

通过 vanishingAnimation() 自定义方法实现 Toast 提示的消失动画。代码如下：

```
01  private void vanishingAnimation() {
02      //设置图标消失动画的延迟时间与透明度
03      ViewPropertyAnimator.animate(loadToastView).setStartDelay(1000).alpha(0f)
04              .translationY(-loadToastView.getHeight() + mTranslationY) //设置位置
05              .setInterpolator(new AccelerateInterpolator())      //设置动画加速
06              .setDuration(300).start();                          //设置动画的持续时间并启动
07  }
```

（5）打开 MainActivity 类，在默认生成的 onCreate() 方法中，创建 LoadToast 对象，然后设置 Toast 加载提示的起始位置，最后设置 3 个按钮所对应的单击事件。

扩展学习

本实例实现了 Toast 加载、错误与完成的运行效果。请尝试在 LoadToast 类中分别创建 setText() 方法、setTextColor() 方法、setBackgroundColor() 方法和 setProgressColor() 方法，实现改变加载进度条运行效果的功能。

明日科技

第23章 抢红包服务

实例 209 微信抢红包

应用场景：微信

源码位置：光盘\Code\23\209

实例说明

微信抢红包是通过获取红包通知中的文字进行判断，符合红包的通知文字将打开红包通知所对应的界面进行抢红包的操作。本实例将实现微信自动抢红包的功能。运行实例，将显示效果如图 23.1 所示的界面，自动显示开启服务对话框；单击如图 23.2 所示的无障碍界面中的微信抢红包服务菜单，将开启微信抢红包服务，如图 23.3 所示。

图 23.1 开启服务对话框

图 23.2 显示无障碍界面

图 23.3 开启微信抢红包服务

关键技术

本实例主要使用 AccessibilityEvent 类中的 getPackageName() 方法、PendingIntent 类中的 send() 方法、AccessibilityNodeInfo 类中的 performAction() 方法，实现微信自动抢红包的效果。

负责声明：本实例仅供学习使用，抢红包的速度取决于手机的性能与网络，如果微信进行更新时，需要对该实例进行优化，否则无法正常使用。禁止使用本实例参与赌博活动与商业用途。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“WeChatGrabRedPackage”，公司域名为“mingrisoft.com”。

（2）导入资源文件：复制“光盘\Code\23\209\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）在“app\src\main\java”目录中创建 service 包，用于处理 Accessibility 服务与 Notification 服务事件，该包中包含 3 个类文件。其中 GrabRedPacketService 类中的 onAccessibilityEvent() 方法用于获取微信红包通知的类名称，并判断是否为微信 App。代码如下：

```
01  public void onAccessibilityEvent(AccessibilityEvent event) {
02      //获取当前界面通知中应用的包名
03      String pkn = String.valueOf(event.getPackageName());
04      //判断当前界面信息是不是空的
05      if(accessbilities != null && !accessbilities.isEmpty()) {
06          //遍历当前界面中的信息
07          for (Accessbility job : accessbilities) {
08              //判断包名是否相同
09              if(pkn.equals(job.getTargetPackageName()) && job.isEnable()) {
10                  //传递获取到的信息
11                  job.onReceiveAccessbility(event);
12              }
13          }
14      }
15  }
```

（4）在“app\src\main\java”目录中创建 accessbility 包，用于保存微信抢红包的信息处理文件，该包中包含 3 个类文件。其中，WechatAccessbility 类中的 openHongBaoNotification() 自定义方法用于实现打开微信红包的通知栏。代码如下：

```
01  @TargetApi(Build.VERSION_CODES.JELLY_BEAN)
02  private void openHongBaoNotification(Notification notification) {
03      //判断是否收到红包
04      isReceivingHongbao = true;
05      //将微信的通知栏消息打开
06      PendingIntent pendingIntent = notification.contentIntent;
```

明日科技

```
07      //判断是否为锁屏或者是黑屏状态
08      boolean lock = NotificationUtil.isLockScreen(getContext());
09
10      if (!lock) {
11          //单击打开通知
12          NotificationUtil.send(pendingIntent);
13      } else {
14          //显示通知
15          NotificationUtil.showNotify(getContext(), String. valueOf(notification.tickerText),
16                  pendingIntent);
17      }
18      //播放声音与振动
19      NotificationUtil.playEffect(getContext(), getConfig());
20  }
```

通过 handleChatListHongBao() 自定义方法实现领取聊天界面中的红包。代码如下：

```
01  @TargetApi(Build.VERSION_CODES.JELLY_BEAN_MR2)
02  private void handleChatListHongBao() {
03      //获取当前窗口信息
04      AccessibilityNodeInfo nodeInfo = getService().getRootInActiveWindow();
05      if (nodeInfo == null) {
06          return;
07      }
08      //找到领取红包信息并保存
09      List<AccessibilityNodeInfo> list = nodeInfo.
10              findAccessibilityNodeInfosByText("领取红包");
11      if (list != null && list.isEmpty()) {
12          //从消息列表查找红包
13          AccessibilityNodeInfo node = AccessibilityUtil.
14                  findNodeInfosByText(nodeInfo, "[微信红包]");
15          if (node != null) {
16              //红包可以领取标记
17              isReceivingHongbao = true;
18              //单击微信红包
19              AccessibilityUtil.performClick(nodeInfo);
20          }
21      } else if (list != null) {
22          if (isReceivingHongbao) {
23              //最新的红包领起
24              AccessibilityNodeInfo node = list.get(list.size() - 1);
25              //单击最新的微信红包
26              AccessibilityUtil.performClick(node);
27              //设置红包已经领取完标记
28              isReceivingHongbao = false;
```

```
        }
    }
}
```

通过 handleLuckyMoneyReceive() 自定义方法实现拆红包的功能。代码如下：

```
@TargetApi(Build.VERSION_CODES.JELLY_BEAN)
private void handleLuckyMoneyReceive() {
    //获取当前窗口的信息
    AccessibilityNodeInfo nodeInfo = getService().getRootInActiveWindow();
    if (nodeInfo == null) {
        return;
    }

    AccessibilityNodeInfo targetNode = null;
    //获取微信打开红包后的事件
    int event = getConfig().getWechatAfterOpenHongBaoEvent();
    int wechatVersion = getWechatVersion();                          //获取微信版本
    if (event == Config.DISMANTLE_RED_PACKET) {                      //拆红包
        if (wechatVersion < USE_ID_MIN_VERSION) {                    //当前微信版本小于7.0
            //使用文字匹配
            targetNode = AccessibilityUtil.findNodeInfosByText(nodeInfo, "拆红包");
        } else {
            //设置拆红包按钮的ID
            String buttonId = "com.tencent.mm:id/b43";
            if (wechatVersion == 700) {
                buttonId = "com.tencent.mm:id/b2c";
            }
            if (buttonId != null) {
                //通过ID查找界面信息与按钮信息
                targetNode = AccessibilityUtil.findNodeInfosById(nodeInfo, buttonId);
            }
            if (targetNode == null) {                                //通过组件查找
                targetNode = AccessibilityUtil.
                        findNodeInfosByClassName(nodeInfo, BUTTON_CLASS_NAME);
            }
        }
    }
    //如果拆红包对话框存在，则单击拆开红包
    if (targetNode != null) {
        //拆红包按钮
        final AccessibilityNodeInfo n = targetNode;
        //单击拆红包按钮
        AccessibilityUtil.performClick(n);
    }
}
```

明日科技

（5）在“app\src\main\java\com.mingrisoft.wechatgrabredpackage”包中设置微信抢红包的主界面显示功能。

（6）打开“app\manifests”目录中的 AndroidManifest.xml 文件，在该文件中添加服务代码。

扩展学习

本实例实现了微信自动抢红包的功能。请尝试在 MainActivity 类的 onCreate() 方法中设置主界面 ActionBar 所显示的应用版本号。

实例 210　QQ 抢红包

应用场景：自动执行事件

源码位置：光盘\Code\23\210

实例说明

QQ 自动抢红包是通过获取 QQ 聊天窗口中的文字对红包进行判断，本实例将实现一个 QQ 自动抢红包的功能。运行实例，单击“设置抢红包服务”按钮，如图 23.4 所示；找到无障碍界面，然后单击无障碍界面中的 QQ 自动抢红包菜单，如图 23.5 所示；开启 QQ 自动抢红包服务，如图 23.6 所示。

图 23.4　单击“抢红包服务”按钮

图 23.5　显示无障碍界面

图 23.6　开启 QQ 自动抢红包服务

关键技术

关 键 技 术

本实例主要使用 AccessibilityNodeInfo 类中的 findAccessibilityNodeInfosByText() 方法、getChildCount() 方法、getChild() 方法，AccessibilityRecord 类中的 getSource() 方法，实现 QQ 自动抢红包的功能。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“QQAUtomaticallyGrabRed”，公司域名为“mingrisoft. com”。

（2）导入资源文件：复制“光盘\Code\23\210\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）在“app\res”目录中创建一个 xml 包，在该包中创建一个名称为“service_config.xml”的资源文件，用于 QQ 自动抢红包的后台服务。（代码位置：app\res\layout\service_config.xml）

（4）在“app\src\main\java\com.mingrisoft.qqautomaticallygrabred”包中创建一个 HongBao 类，该类继承自 AccessibilityService，用于实现抢红包服务。首先设置需要拦截的红包名称。代码如下：

```
private String[] filter = new String[]{"恭喜发财"};  //红包名字的数组
```

然后通过 onAccessibilityEvent() 方法，实现获取当前界面的信息源，代码如下：

```
01  public void onAccessibilityEvent(AccessibilityEvent event) {
02      //获取当前界面的信息
03      accessibilityNodeInfo = event.getSource();
04      //信息为空直接返回
05      if (accessibilityNodeInfo == null) {
06          return;
07      }
08      //执行是否打开红包的方法
09      isOpen(accessibilityNodeInfo);
10  }
```

通过 findByText() 自定义方法查找指定名称的红包，代码如下：

```
01  private List<AccessibilityNodeInfo> findByText(AccessibilityNodeInfo rootNodeInfo) {
02      //遍历红包名称，并查找该名称的红包
03      for (String name : filter) {
04          List<AccessibilityNodeInfo> list =
05                  rootNodeInfo.findAccessibilityNodeInfosByText(name);
06          //如果指定红包名称的信息不是空的  就返回这条信息
07          if (list != null && !list.isEmpty()) {
08              return list;
09          }
10      }
11      return null;
12  }
```

通过 openRed() 自定义方法实现打开红包的功能，代码如下：

```
01  private void openRed(AccessibilityNodeInfo rootNodeInfo) {
02      //获取当前界面中子控件的数量
03      int child = rootNodeInfo.getChildCount();
04      //遍历当前界面中的每个子控件
```

明日科技

```
05      for (int i=0;i<child;i++) {
06          //获取每个子控件
07          AccessibilityNodeInfo info = rootNodeInfo.getChild(i);
08          //判断子控件是否为空
09          if (info == null) {
10              continue;
11          }
12          if (info.getChildCount() > 0) {
13              openRed(info);                                    //设置新的数据
14          } else {                                              //根据文字判断是否有没拆开的红包
15              if ("单击拆开".equals(info.getText())){           //如果有新的红包则进行拆红包
16                  rootNodeInfo.performAction(AccessibilityNodeInfo.ACTION_CLICK);
17              }
18          }
19      }
20  }
```

（5）打开 MainActivity 类，在该类中设置开启服务按钮的单击事件，用于开启 QQ 自动抢红包服务的界面跳转。

（6）打开“app\manifests”目录中的 AndroidManifest.xml 文件，在该文件中添加服务代码。

扩展学习

本实例实现了 QQ 自动抢名称为“恭喜发财”红包的功能。请尝试修改 HongBao 类中的红包名称数组，将常用红包词汇添加在数组中（如添加“生日快乐”词汇），实现自动抢其他名称的红包功能。

明日科技

第24章

App 应用开发必备模块

实例 211　登录注册

应用场景：需要进入身份验证的 App 应用时

源码位置：光盘\Code\24\211

视频位置：光盘\Video\24\211

视频讲解

实例说明

App 应用中通常都会有登录与注册界面，也有一些展示类的 App 应用允许以游客的身份浏览，但是涉及到发布信息或者评论信息时，均需要身份验证，即登录注册。本实例将实现登录注册界面。运行实例，将显示如图 24.1 所示的登录界面；单击“注册”按钮时，跳转到如图 24.2 所示的注册界面。

图 24.1　登录界面

图 24.2　注册界面

关键技术

本实例主要使用 Thread 线程的 Runnable() 方法、TextUtils 类中的 isEmpty() 方法，实现判断输入内容是否为空，以及等待注册的线程。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“EnterAndRegister”，公司域名为“mingrisoft.com”。

（2）导入资源：复制“光盘\Code\24\211\src”文件中的图片资源，粘贴到“app\res\drawable”文件中，并在该文件中创建输入框及按钮的属性文件。

（3）在“app\res\values”文件夹中的 styles.xml 文件中编写顶部状态栏的属性。

（4）在 login_enter_activity 布局文件中绘制登录界面。

（5）在 login_register_activity 布局文件中绘制注册界面。

（6）在 EnterActivity 类文件中，判断用户名与密码是否为空的关键代码如下：

```
01  currentUsername = editPerson.getText().toString().trim();          //去除空格，获取手机号
02  currentPassword = editCode.getText().toString().trim();            //去除空格，获取密码
03
04  if (TextUtils.isEmpty(currentUsername)) {                           //判断手机号是否为空
05      Toast.makeText(this, "手机号不能为空", Toast.LENGTH_SHORT).show();
06      return;
07  }
08  if (TextUtils.isEmpty(currentPassword)) {                           //判断密码是否为空
09      Toast.makeText(this, "密码不能为空", Toast.LENGTH_SHORT).show();
10      return;
11  }
```

（7）在 RegisterActivity 类文件中，判断注册时需填写的手机号是否为空的逻辑代码如下：

```
01  if (TextUtils.isEmpty(username)) {                                  //当手机号没有输入时
02      Toast.makeText(this, "手机号不能为空！", Toast.LENGTH_SHORT).show();
03      editTextP.requestFocus();                                       //让输入框失去焦点
04      return;
05  } else if (TextUtils.isEmpty(password)) {                           //当验证码没有输入时
06      Toast.makeText(this, "验证码不能为空！", Toast.LENGTH_SHORT).show();
07      editSMS.requestFocus();                                         //让输入框失去焦点
08      return;
09  } else if (TextUtils.isEmpty(confirm_password)) {                   //当注册密码没有输入时
10      Toast.makeText(this, "密码不能为空！", Toast.LENGTH_SHORT).show();
11      editTextCT.requestFocus();                                      //让输入框失去焦点
12      return;
13  }
```

扩展学习

本实例在登录界面中输入的密码不可见，请尝试在布局文件中删除 EditText 的一个属性，使输入的密码可见。

实例 212　视频登录

应用场景：App 应用登录界面

源码位置：光盘\Code\24\212

视频位置：光盘\Video\24\212

实例说明

视频登录主要是将登录界面的背景设置为视频，并在视频的上面添加输入用户名与密码的输入框，以及登录与注册的按钮。注意背景是一直播放的视频，不是 GIF 格式文件。运行实例，将显示如图 24.3 所示的刚打开的登录界面，单击“登录”按钮，将显示如图 24.4 所示等待输入用户名与密码的界面。

图 24.3　刚打开登录界面

图 24.4　等待输入用户名密码界面

关键技术

本实例主要使用 VideoView 类中的 setVideoPath() 方法，View 类中的 setVisibility() 方法，实现设置显示视频的文件路径及设置控件的显示与隐藏效果。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“EnterView”，公司域名为“mingrisoft.com”。

（2）在“app\res\drawable”文件夹中创建按钮与输入框的属性。

（3）在“app\res”文件夹中创建 raw 文件，用于加载登录界面的视频。

（4）创建 MyVideoView 类，用于加载视频文件，并且在布局中调用该类显示视频文件。关键代码如下：

```
/**
 * 初始化，只有传递两个参数时才能在布局中调用
 **/
public MyVideoView(Context context, AttributeSet attrs) {
    super(context, attrs);
}
```

（5）在 MainActivity 的布局文件中调用 MyVideoView 类，并设置视频文件大小为填充整个界面，关键代码如下：

```
<mingrisoft.com.enterview.MyVideoView
    android:id="@+id/videoView"
    android:layout_width="match_parent"
    android:layout_height="match_parent" />
```

（6）自定义动画，用于显示用户名与密码的输入框，代码如下：

```
ObjectAnimator anim = ObjectAnimator.ofFloat(appName, "alpha", 0,1);      //设置透明度
anim.setDuration(4000);                                                   //动画时间
anim.setRepeatCount(1);                                                   //设置动画重复（如果大于0，永不重复）
anim.setRepeatMode(ObjectAnimator.REVERSE);
anim.start();                                                             //开始动画
anim.addListener(new AnimatorListenerAdapter() {
    @Override
    public void onAnimationEnd(Animator animation) {
        appName.setVisibility(View.INVISIBLE);                            //动画结束后消失"UserEnter"
        editAll.setVisibility(View.VISIBLE);                              //显示待输入的框
    }
});
```

（7）在 MainActivity 中加载视频文件，此时需要将较大的资源文件转换成字节流进行读取，本实例中的视频文件需要用字节的形式读取并显示出来，关键代码如下：

```
File videoFile;                                                           //定义文件
try {
    FileOutputStream fos = openFileOutput(VIDEO_NAME, MODE_PRIVATE);      //输出字节文件流
    InputStream in = getResources().openRawResource(R.raw.welcome_video);//输入字节文件流
    byte[] buff = new byte[1024];                                         //转换成byte字节
    int len = 0;
    while ((len = in.read(buff)) != -1) {                                 //开始转换
```

```
        fos.write(buff, 0, len);
    }
} catch (FileNotFoundException e) {                          //抛出没有找到文件异常
    e.printStackTrace();
} catch (IOException e) {                                    //抛出io流异常
    e.printStackTrace();
}
videoFile = getFileStreamPath(VIDEO_NAME);                   //获取视频文件的路径
if (!videoFile.exists())
    throw new RuntimeException("视频文件不存在，确定有welcome_video.mp4源文件吗？");
return videoFile;                                            //返回视频文件
```

（8）在 MainActivity 类中需要将视频的播放充满手机屏幕，沉浸在状态栏下方，可以通过 Window.setFlags() 方法添加指定标签实现，代码如下：

```
/**
 * 使播放视频充满屏幕，包括状态栏下方也充满视频
 **/
if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.KITKAT) {
    Window window = getWindow();
    window.setFlags(
            WindowManager.LayoutParams.FLAG_TRANSLUCENT_STATUS,
            WindowManager.LayoutParams.FLAG_TRANSLUCENT_STATUS);
}
```

扩展学习

本实例输入框中光标的颜色为粉色，为了搭配界面色彩，请尝试将输入框的状态改为白色，即修改“values/color”中的颜色。

实例 213　设置界面

应用场景：App 应用的设置界面

源码位置：光盘\Code\24\213

实例说明

手机的 App 应用都会有设置界面，通过它可以设置声音、提示清除缓存等。本实例将实现一个设置界面。运行实例，将显示如图 24.5 所示的界面，关闭声音与扬声器的界面如图 24.6 所示，关闭消息通知如图 24.7 所示。

明日科技

图 24.5　设置界面

图 24.6　关闭声音与扬声器

图 24.7　关闭消息通知

关键技术

本实例主要使用 Context 类中的 obtainStyledAttributes() 方法，检索上下文主题中的样式属性信息。另外，还使用 TypedArray 类中的 getDrawable() 方法，获取自定义开关按钮的属性。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“AppSetActivity”，公司域名为“mingrisoft.com”。

（2）在“app\res”文件夹中的 mipmap 文件夹，粘贴相关的图片资源。

（3）在“app\res\values”文件夹中创建 attrs 文件，用于设置开关按钮的属性。

（4）修改显示界面的主题样式，去除系统默认添加的 ActionBar 标题栏。方法是将“app\res\values\styles”文件中的“DarkActionBar”修改为“NoActionBar”。

（5）创建 switch_button.xml 布局文件并设置其父布局为 FrameLayout，用来包裹两张图片。

（6）创建 SwitchButton 类，该类继承自 view，用于设置开关按钮的状态。

（7）创建好 SwitchButton 类后，在该类的初始化方法中添加的重要代码如下：

```
01  //通过调用obtainStyledAttributes方法来获取一个TypeArray，然后由该TypeArray对属性进行设置
02  TypedArray typedArray = context.obtainStyledAttributes(attrs, R.styleable.SwitchButton);
03  //取出打开的图片
04  Drawable openDrawable = typedArray.getDrawable(R.styleable.SwitchButton_switchOpenImage);
05  //取出关闭的图片
```

```
06  Drawable closeDrawable = typedArray.getDrawable(R.styleable.SwitchButton_switchCloseImage);
07  int switchStatus = typedArray.getInt(R.styleable.SwitchButton_switchStatus, 0);
08  //调用结束后务必调用recycle()方法，否则这次的设定会对下次的使用造成影响
09  typedArray.recycle();
```

（8）判断 SwitchButton 的开关状态以及打开关闭的执行方法，代码如下：

```
01  /**
02   * 开关是否为打开状态
03   */
04  public boolean isSwitchOpen() {
05      return openImage.getVisibility() == View.VISIBLE;
06  }
07
08  /**
09   * 打开开关
10   */
11  public void openSwitch() {
12      openImage.setVisibility(View.VISIBLE);                    //显示打开的图片
13      closeImage.setVisibility(View.INVISIBLE);                 //隐藏关闭的图片
14  }
15  /**
16   * 关闭开关
17   */
18  public void closeSwitch() {
19      openImage.setVisibility(View.INVISIBLE);                  //隐藏打开的图片
20      closeImage.setVisibility(View.VISIBLE);                   //显示关闭的图片
21  }
```

（9）单击 SwitchButton 按钮时执行的操作，以声音按钮为例，关键代码如下：

```
01  case R.id.switch_sound:                                       //单击声音的开关按钮
02      if (switch_sound.isSwitchOpen()) {                        //判断开关是否已打开
03          switch_sound.closeSwitch();                           //如果打开执行关闭操作
04      } else {
05          switch_sound.openSwitch();                            //如果未打开执行打开操作
06      }
07      break;
```

扩展学习

本实例在关闭新消息提醒时，使“声音与震动”的按钮隐藏，请尝试在关闭新消息提醒的按钮时，不隐藏“声音与震动”按钮，只是关闭“声音与震动”的按钮。

明日科技

实例 214　上拉加载、下拉刷新

应用场景：App 应用获取网络数据

源码位置：光盘\Code\24\214

实例说明

App 应用中通常会包含上拉加载、下拉刷新的操作，这样的操作方便用户获取服务器的新数据。本实例将实现新闻 App 应用中的上拉加载、下拉刷新的功能。运行实例，将显示如图 24.8 所示的初始界面；手指从上向下滑动时，将显示如图 24.9 所示的为下拉刷新的界面；手指从下向上滑动时，将显示如图 24.10 为正在刷新的界面。

图 24.8　初始界面

图 24.9　下拉刷新界面

图 24.10　上拉加载界面

关键技术

关键技术

本实例主要使用 AnimationUtils 类中的 loadAnimation() 方法，View 类中的 startAnimation() 方法、clearAnimation() 方法，实现刷新动画的功能。

实现过程

（1）打开 Android Studio，新建一个项目，并将其命名为“RefreshAndLoading”，公司域名为“mingrisoft.com”。

（2）在“app\res”文件夹中创建 anim 文件夹，并在该文件夹中创建 rotating.xml 文件，用于加载界面刷新时的动画。

（3）导入资源：复制“光盘\Code\24\214\src”文件中的图片资源，粘贴到“app\res\mipmap”文件中。

（4）创建一个刷新监听类，命名为 MyListener，并让这个监听类实现自定义 Layout 的刷新方法，以便在主界面中调用该监听。关键代码如下：

```
/**
 * 刷新方法
 **/
@Override
public void onRefresh(final PullToRefreshLayout pullToRefreshLayout) {
    //下拉刷新操作
    new Handler() {
        @Override
        public void handleMessage(Message msg) {
            pullToRefreshLayout.refreshFinish(PullToRefreshLayout.SUCCEED);
        }
    }.sendEmptyMessageDelayed(0, 3000);
}
```

（5）定义一个接口，用来判断 ListView 是否可以下拉，关键代码如下：

```
/**
 * 判断是否可以下拉，如果不需要下拉功能可以直接return false
 * @return true表示可以下拉，否则返回false
 */
boolean canPullDown();
```

（6）在自定义的 ListView 中实现下拉刷新，关键代码如下：

```
@Override
public boolean canPullDown()
{
    if (getCount() == 0)
    {
        //没有item的时候也可以下拉刷新
        return true;
    } else if (getFirstVisiblePosition() == 0 && getChildAt(0).getTop() >= 0){
        //滑到ListView的顶部了
        return true;
    } else
        return false;
}
```

（7）判断 ListView 已经滑动到底部时，实现加载更多的功能，关键代码如下：

```
/**
 * listview滚动监听
 **/
listView.setOnScrollListener(new AbsListView.OnScrollListener() {
    @Override
    public void onScrollStateChanged(AbsListView view, int scrollState) {
        //当不滚动时
        if (scrollState == AbsListView.OnScrollListener.SCROLL_STATE_IDLE) {
            //判断是否滚动到底部
            if (view.getLastVisiblePosition() == view.getCount() - 1) {
                //加载更多功能的代码
                try {
                    Thread.sleep(1000);    //睡眠1s
                } catch (InterruptedException e) {
                    e.printStackTrace();
                }
                /**
                 * 添加加载出来的数据
                 **/
                for (int i = 0; i < 10; i++) {
                    items.add(items.size(), "这里是第" + (items.size() + i) + "条数据");
                }
                adapter.addData(items);    //把数据用适配器显示出来
            }
        }
    }
}
```

扩展学习

本实例中获取网络图片并进行缓存，请尝试设置缓存的图片位置。

实例 215　动态显示或隐藏标题栏

应用场景：界面效果

源码位置：光盘\Code\24\215

实例说明

本实例将实现当滑动列表时，根据滑动的方向来控制标题栏的显示或者隐藏，即在向下滑动列表时显示标题栏，向上滑动时隐藏标题栏。运行实例，将显示如图 24.11 所示的初始界面；向上滑动时隐藏标题栏，如图 24.12 所示的隐藏标题栏的界面。

明日科技

图 24.11　初始界面

图 24.12　隐藏标题样式的效果

关键技术

本实例中主要使用 ListView 的滑动触摸监听来对标题栏的显示和隐藏进行控制，使用属性动画来实现标题栏显示和隐藏的动画效果。

实现过程

（1）打开 Android Studio，新建一个项目，并将其命名为“HideQrShowTiTle”，公司域名为“mingrisoft.com”。

（2）导入资源文件：复制“光盘\Code\24\215\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）在MainActivity类的onCreate()方法中获取控件的对象实例，并设置ListView中要加载的数据。代码如下：

```
01  //获取滑动的判断值
02  touchSlop = ViewConfiguration.get(this).getScaledTouchSlop();
03  toolbar = (Toolbar) findViewById(R.id.toolbar);
04  setSupportActionBar(toolbar);                                    //使用ToolBar替代ActionBar
05  listView = (ListView) findViewById(R.id.listview);
06  View header = new View(this);                                    //获取一个View的对象
07  //设置View的大小与ActionBar的大小相同
08  header.setLayoutParams(new AbsListView.LayoutParams(AbsListView.LayoutParams.MATCH_PARENT,
09          (int)getResources().getDimension(R.dimen.abc_action_bar_default_height_material)));
10  listView.addHeaderView(header);                                  //为列表添加头部
11  //添加测试数据
```

```
for (int i = 0; i < mStr.length; i++) {
    mStr[i] = "Item " + i;
}
//绑定适配器
listView.setAdapter(new ArrayAdapter<>(MainActivity.this,
        android.R.layout.simple_expandable_list_item_1, mStr));
//设置触摸监听
listView.setOnTouchListener(this);
```

在触摸监听的回调方法中获取滑动的距离并做判断，控制标题栏的显示和隐藏。代码如下：

```
    @Override
    public boolean onTouch(View v, MotionEvent event) {
        switch (event.getAction()) {
            case MotionEvent.ACTION_DOWN:
                firstY = event.getY();                          //获取起始的y坐标
                break;
            case MotionEvent.ACTION_MOVE:
                currentY = event.getY();                        //获取滑动的y坐标
                //判断列表的滑动是向上还是向下
                direction = currentY - firstY > touchSlop ? 0:1;
                toolbarAnim(direction);                         //隐藏标题栏
                break;
        }
        return false;
    }
```

创建 toolbarAnim() 方法，用来设置标题栏的显示或隐藏动画效果。代码如下：

```
/**
 * 标题栏的显示或隐藏的动画
 * @param flag
 */
private void toolbarAnim(int flag) {
    if (animator != null && animator.isRunning()) {
        animator.cancel();                                      //取消动画
    }
    //设置属性动画是标题栏显示或者隐藏
    animator = ObjectAnimator.ofFloat(toolbar, "translationY",
            toolbar.getTranslationY(),flag == 0 ? 0 : -toolbar.getHeight());
    animator.start();                                           //开始动画
}
```

扩展学习

请尝试去掉列表中的滚动条，使列表在滑动时不显示滚动条。

实例 216　仿今日头条 TAB 效果

应用场景：界面效果

源码位置：光盘\Code\24\216

实例说明

本实例要使用到一个第三方指示器开源库叫 MagicIndicator，MagicIndicator 库非常强大，可以满足大部分的指示器效果，使用起来也非常简单，可以轻松地实现今日头条的 TAB 效果。运行实例，将显示打开界面的初始效果，如图 24.13 所示；界面滑动中的效果如图 24.14 所示。

图 24.13　界面的默认效果

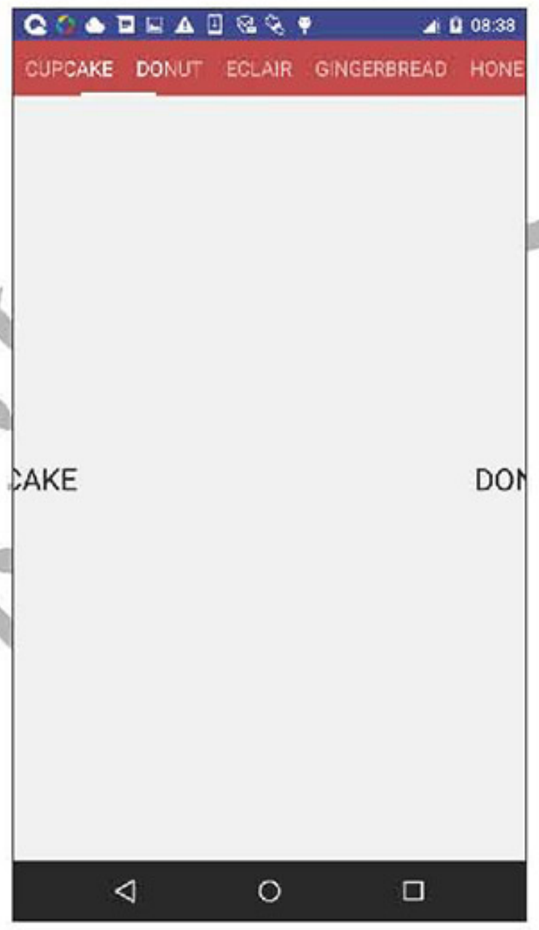


图 24.14　滑动中的效果

关键技术

关键技术

本实例主要使用 MagicIndicator 来实现仿今日头条的 TAB 效果（MagicIndicato 的下载地址为：https://github.com/hackware1993/MagicIndicator）。想要实现 TAB 效果，只需要设置 MagicIndicator 对象和 CommonNavigator 对象即可。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“ThirdTab”，公司域名为“mingrisoft. com”。

（2）导入资源文件：复制“光盘\Code\24\216\src”中的所有文件夹，粘贴到“app\es”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）在 MainActivity 类的 onCreate() 方法中初始化 ViewPager 对象、MagicIndicator 对象和 Common Navigator 对象。代码如下：

代码位置：ThirdTab\app\src\main\java\com\mingrisoft\ThirdTab\MainActivity.java

```
mViewPager = (ViewPager) findViewById(R.id.view_pager);
mViewPager.setAdapter(mExamplePagerAdapter);                          //创建适配器
mMagicIndicator = (MagicIndicator) findViewById(R.id.magic_indicator1);
mMagicIndicator.setBackgroundColor(Color.parseColor("#d43d3d"));      //设置背景颜色
mCommonNavigator = new CommonNavigator(this);                         //创建对象
mCommonNavigator.setSkimOver(true);
```

创建适配器，设置导航条的效果。代码如下：

```
private CommonNavigatorAdapter adapter = new CommonNavigatorAdapter() {  //设置选项卡的数据
    @Override
    public int getCount() {
        return mDataList.size();
    }
    @Override
    public IPagerTitleView getTitleView(Context context, final int index) {
        ClipPagerTitleView clipPagerTitleView = new ClipPagerTitleView(context);
        clipPagerTitleView.setText(mDataList.get(index));                 //设置Tab内容
        clipPagerTitleView.setTextColor(Color.parseColor("#f2c4c4"));     //设置字体颜色
        clipPagerTitleView.setClipColor(Color.WHITE);                     //设置选中颜色
        clipPagerTitleView.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                mViewPager.setCurrentItem(index                           //设置ViewPager显示界面
            }
        });
        return clipPagerTitleView;
    }

    @Override
    public IPagerIndicator getIndicator(Context context) {
        return null;
    }
};
```

将 ViewPager 与 MagicIndicator 关联起来。代码如下：

```
mMagicIndicator.setNavigator(mCommonNavigator);                   //设置指示器
ViewPagerHelper.bind(mMagicIndicator, mViewPager);                //绑定ViewPager
```

扩展学习

请尝试为指示器增加下划线跟随滚动的效果。

实例 217　选择头像的注册

应用场景：注册页面

源码位置：光盘\Code\24\217

实例说明

注册功能是 App 应用中常用的，本实例将实现可以选择头像的注册功能。运行实例，运行效果如图 24.15 所示；单击头像，进入如图 24.16 所示的头像选择页面；选择头像后，自动跳转到注册页面，并填写注册信息，如图 24.17 所示。

图 24.15　初始效果

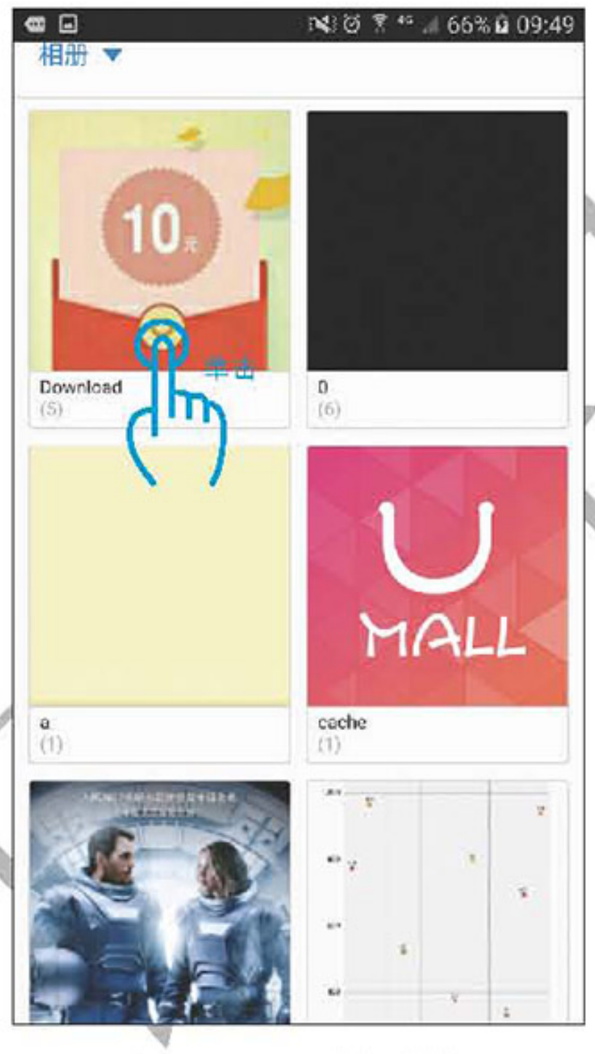


图 24.16　选择头像

图 24.17　填写完信息后的效果

关键技术

本实例主要使用 View 类中的 setOnClickListener() 方法、Uri 类中的 fromFile() 方法、File 类中的 File() 方法、FragmentActivity 类中的 startActivityForResult() 方法，实现选择头像功能。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“CutIV”，公司域名为“mingrisoft.com”。

（2）导入资源文件：复制“光盘\Code\24\217\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）在“app\java\com\mingrisoft\cutiv”文件夹下默认生成的 MainActivity 类中，为头像控件绑定单击事件，实现单击后打开图库。代码如下：

```
picture.setOnClickListener(new OnClickListener() {
    @Override
    public void onClick(View v) {
       Intent intent = new Intent(Intent.ACTION_PICK,
        android.provider.MediaStore.Images.Media.EXTERNAL_CONTENT_URI);
        //此处调用了图片选择器
        //如果直接写intent.setDataAndType("image/*");调用的是系统图库
        intent.setDataAndType(MediaStore.Images.Media.EXTERNAL_CONTENT_URI, "image/*");
        //开启意图
        startActivityForResult(intent, SHOW_PICTURE);
    }
});
```

（4）在 MainActivity 类中重写 onActivityResult() 方法接收回调信息，用于接收在图库中选择的图片信息，并设置到头像上。代码如下：

```
//接收回调信息
@Override
protected void onActivityResult(int requestCode, int resultCode, Intent data) {
    switch (requestCode) {
        case SHOW_PICTURE:
            if (data != null) {                          //判断数据是否为空
                Uri url1 = data.getData();               //获得图片的Uri
                try {
                    String[] proj = {MediaStore.Images.Media.DATA};
                    //Android多媒体数据库的封装接口
                    Cursor cursor = managedQuery(url1, proj, null, null, null);
                    //获得用户所选择的图片的索引值
                    int column_index = cursor.getColumnIndexOrThrow(MediaStore.Images.Media.DATA);
                    //将光标移至开头，这个很重要，不小心很容易引起越界
                    cursor.moveToFirst();
                    //根据索引值获取图片路径
                    String path = cursor.getString(column_index);
                    //获取bitmap对象
                    bm1 = Bimp.revitionImageSize(path);
                    //设置图片
                    picture.setImageBitmap(bm1);
                    //设置图片占满控件
                    picture.setScaleType(ImageView.ScaleType.FIT_XY);
                } catch (IOException e) {
                    e.printStackTrace();
                }
            }
            break;
        default:
```

```
30                  break;
31          }
32  }
```

扩展学习

尝试加入选择图片后剪辑图片的功能。

实例 218　明日首页轮播图

应用场景：应用的首页轮播图

源码位置：光盘\Code\24\218
视频位置：光盘\Video\24\218

实例说明

循环播放热点图片，或在 App 应用页面上方自动滚动广告图是 App 应用最基本的功能。本实例将介绍如何实现轮播图片，并且在图片下方加上小圆点指示器，标记当前位置以及显示当前文字。运行实例，初始效果如图 24.18 所示；每隔 2 秒后切换轮播图，如图 24.19 所示。

图 24.18　初始效果

图 24.19　切换后效果

关键技术

本实例主要使用 ViewPager 实现轮播图片，以及使用 ScheduledExecutorService+Handler 实现自动轮播播放。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“LBTViewPage”，公司域名为“mingrisoft.com”。

（2）导入资源文件：复制“光盘\Code\24\218\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）在“app\res\layout”目录下默认生成的 android_main.xml 布局文件中，写入 ViewPager 控件。代码如下：

```
<!--轮播图控件-->
<android.support.v4.view.ViewPager
    android:id="@+id/vp"
    android:layout_width="match_parent"
    android:layout_height="match_parent"/>
```

（4）在“app\java\com\mingrisoft\lbtviewpage”文件夹下默认生成的 MainActivity 类中，写入自定义 ViewPagerAdapter 数据适配器。代码如下：

```
//自定义Adapter
private class ViewPagerAdapter extends PagerAdapter {
    @Override
    public int getCount() {
        return images.size();
    }
    @Override
    public boolean isViewFromObject(View arg0, Object arg1) {
        return arg0 == arg1;
    }
    @Override
    public void destroyItem(ViewGroup view, int position, Object object) {
        //TODO Auto-generated method stub
        view.removeView(images.get(position));
    }
    @Override
    public Object instantiateItem(ViewGroup view, int position) {
        //TODO Auto-generated method stub
        //添加图片控件到轮播图控件中
        view.addView(images.get(position));
        return images.get(position);
    }
}
```

利用线程执行轮播动画，主要是注册线程，实现每 2 秒执行一次切换图片方法。代码如下：

```
//初始化定时线程
scheduledExecutorService = Executors.newSingleThreadScheduledExecutor();
//设定执行线程计划，初始2秒延迟，每次任务完成后延迟2秒再执行一次任务
scheduledExecutorService.scheduleWithFixedDelay(
```

```
        new ViewPageTask(),
        2,
        2,
        TimeUnit.SECONDS);
```

利用轮播图切换图片监听，切换后设置图片和文字。代码如下：

```
    //轮播图滑动事件监听
    mViewPaper.setOnPageChangeListener(new ViewPager.OnPageChangeListener() {
        //页面切换后触发
        @Override
        public void onPageSelected(int position) {
            //设置轮播图文字
            title.setText(titles[position]);
            //设置当前小点图片
            dots.get(position).setBackgroundResource(R.drawable.dot_focused);
            //设置前一个小点图片
            dots.get(oldPosition).setBackgroundResource(R.drawable.dot_normal);
            //记录小点图片ID
            oldPosition = position;
            //记录图片位置
            currentItem = position;
        }
        @Override
        public void onPageScrolled(int arg0, float arg1, int arg2) {
        }
        @Override
        public void onPageScrollStateChanged(int arg0) {
        }
    });
```

扩展学习

本实例实现默认显示 5 张图片，请尝试添加或者减少图片。

实例 219　微信底部选项卡

应用场景：滑动变色选项卡

源码位置：光盘\Code\24\219

实例说明

在 App 应用中滑动切换主显示区域时，底部或者顶部会跟随着相应的图标，并且该图标会随着切换程度渐变颜色。本实例将实现一个通过左右滑动切换界面的主显示区域并渐变底部图标颜色的效果。

运行实例，效果如图 24.20 所示；从右向左滑动，切换与图标对应的显示区域，如图 24.21 所示。

图 24.20　从右向左滑动

图 24.21　切换与图标对应的显示区域

关键技术

本实例主要使用 View 类中的 onDraw() 方法、自定义 ChangeIcon 类，实现仿微信底部选项卡的运行效果。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“WeiMain”，公司域名为“mingrisoft. com”。

（2）导入资源文件：复制“光盘\Code\24\219\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）在默认生成的 activity_main.xml 布局文件中，首先设置 ViewPager 控件，用于显示主视图，然后设置自定义图标所在的布局管理器，用于显示渐变色的图标。（代码位置：app\res\layout\activity_main.xml）

（4）在 java 目录下的 mingrisoft.com.weimain 包中创建名称为“TabFragment”的 Java 类，该类继承自 Fragment，并且在该类中使用 onCreateView() 方法实现显示在主视图中的文字。（代码位置：app\java\mingrisoft.com.weimain\TabFragment）

（5）首先在“app\src\main\java”目录中创建 view 包，然后在该包中创建名称为“ChangeIcon”的 Java 类，该类继承自 View，用于实现底部图标文字渐变色的自定义控件。（代码位置：WeiMain\app\src\main\java\view\ChangeIcon.java）

在 ChangeIcon 类中，创建 drawColorText() 方法，用于实现绘制可变色的图标文字。代码如下：

```
01  private void drawColorText(Canvas canvas, int alpha) {
02      mTextPaint.setColor(mColor);                          //设置文字画笔颜色
03      mTextPaint.setAlpha(alpha);                           //设置文字画笔的透明度
04      //绘制起始的x坐标
```

```
05        int x = getMeasuredWidth() / 2 - mTextBound.width() / 2;
06        //绘制起始的y坐标
07        int y = mIconRect.bottom + mTextBound.height();
08        //绘制变色文字
09        canvas.drawText(mText, x, y, mTextPaint);
10    }
```

创建 setupGradientIcon() 方法，用于实现绘制可变色的图标。代码如下：

```
01  private void setupGradientIcon(int alpha) {
02      //设置渐变位图范围
03      mBitmap = Bitmap.createBitmap(getMeasuredWidth(), getMeasuredHeight(),
04              Bitmap.Config.ARGB_8888);
05      mCanvas = new Canvas(mBitmap);                                  //设置渐变画布
06      mPaint = new Paint();                                           //设置画笔
07      mPaint.setColor(mColor);                                        //设置画笔颜色
08      mPaint.setAntiAlias(true);                                      //开启抗锯齿
09      mPaint.setDither(true);                                         //开启仿抖动
10      mPaint.setAlpha(alpha);                                         //设置透明度
11      mCanvas.drawRect(mIconRect, mPaint);                            //绘制纯色图标
12      //设置两层交集显示上层底色
13      mPaint.setXfermode(new PorterDuffXfermode(PorterDuff.Mode.DST_IN));
14      mPaint.setAlpha(255);                                           //画笔设置为不透明
15      //根据图标尺寸绘制纯色
16      mCanvas.drawBitmap(mIconBitmap, null, mIconRect, mPaint);
17  }
```

（6）打开 MainActivity 类，在该类中获取 ChangeIcon 自定义控件，并配置 ViewPager 滚动视图。为 ViewPager 控件设置滑动事件，在该事件的 onPageScrolled() 方法中，根据左右滑动的下标来设置图标的透明度。代码如下：

```
01  @Override
02  public void onPageScrolled(int position, float positionOffset,
03                             int positionOffsetPixels) {
04      if (positionOffset > 0) {                                       //如果偏移值大于0
05              ChangeIcon left = mTabIndicators.get(position);         //获取向左滑下标
06          ChangeIcon right = mTabIndicators.get(position+1);          //获取向右滑下标
07          left.setIconAlpha(1 - positionOffset);                      //设置左侧图标透明，右侧变色
08          right.setIconAlpha(positionOffset);                         //设置右侧图标透明，左侧变色
09      }
10  }
```

扩展学习

本实例将 MainActivity 类中的字符串文字显示在主视图当中，请尝试在 TabFragment 类中通过获取到的文字，让主视图显示出图片来。

实例 220　退出高德地图的提示

应用场景：Toast 提示

源码位置：光盘\Code\24\220

实例说明

Toast 类通常用于显示一些快速提示的信息，本实例将模拟一个退出高德地图时弹出的 Toast 提示信息。运行实例，将显示如图 24.22 所示的界面；单击模拟器或真机的返回按钮，将弹出再按一次退出高德地图的消息提示框，如图 24.23 所示。

图 24.22　单击“返回”按钮

图 24.23　弹出 Toast 消息提示框

关键技术

本实例主要使用 Toast 类中的 makeText() 方法、Timer 类中的 schedule() 方法，实现退出高德地图显示提示信息的效果。

实现过程

（1）打开 Android Studio，新建一个项目，命名为“ToastMessage”，公司域名为“mingrisoft. com”。

（2）导入资源文件：复制“光盘\Code\24\220\src”中的所有文件夹，粘贴到“app\res”目录下，单击“Overwrite for all”按钮，覆盖所有自动生成的资源文件。

（3）打开 MainActivity 类，在该类中重写 onKeyDown() 方法，并且在该方法中判断是否单击了

“返回”按钮。代码如下：

```
01  /**
02   * 该方法判断是否单击了返回按钮
03   */
04  @Override
05  public boolean onKeyDown(int keyCode, KeyEvent event) {
06      if (keyCode == KeyEvent.KEYCODE_BACK) {
07          exit();                                          //调用退出方法
08      }
09      return false;
10  }
```

创建 exit() 方法，通过记录退出状态，来实现两秒内再次单击“返回”按钮时退出当前界面，并显示提示信息的功能。代码如下：

```
01  /**
02   * 该方法实现两秒内单击两次返回按钮后实现退出功能
03   */
04  private void exit() {
05      if (isExit == false) {                               //如果没有退出
06          isExit = true;                                   //则准备退出
07          //显示提示信息
08          Toast.makeText(this, "再按一次退出高德地图", Toast.LENGTH_SHORT).show();
09          timer = new Timer();                             //实例化计时器
10          timer.schedule(new TimerTask() {
11              @Override
12              public void run() {
13                  isExit = false;                          //取消退出
14              }
15          }, 2000);                                        //两秒内再次单击返回按钮，将退出该应用
16      } else {
17          finish();                                        //退出当前界面
18          System.exit(0);                                  //释放内存
19      }
20  }
```

扩展学习

本实例在 Toast 消息提示框中实现了退出的提示信息。其实，Toast 消息提示框还可以显示一些变量值，请尝试让其显示当前的退出状态值。

附录

搭建 Android 开发环境

一、开发环境概述

进行 Android 应用开发需要具备两方面的环境要求，下面将分别从两个方面进行介绍。

1. 系统需求

要进行 Android 应用开发，需要有合适的系统环境。表 1 列出了进行 Android 开发所必需的系统环境需求。

表 1　进行 Android 开发所必需的系统环境需求

操作系统	系统版本	内　存	屏幕分辨率
Windows	Microsoft Windows 7/8/10（32 位或 64 位）	最小 4G，推荐 8G	1280×800
Mac OS	Mac OS X 10.10 (Yosemite) 或更高版本，最高为 10.12 (macOS Sierra)	最小 4G，推荐 8G	1280×800
Linux	Linux GNOME 或 KDE（K 桌面环境）	最小 4G，推荐 8G	1280×800

2. 软件需求

要进行 Android 应用开发，除了要有合适的系统环境，还需要有一些软件的支持。通常情况下，我们需要如图 1 所示的这些软件支持。

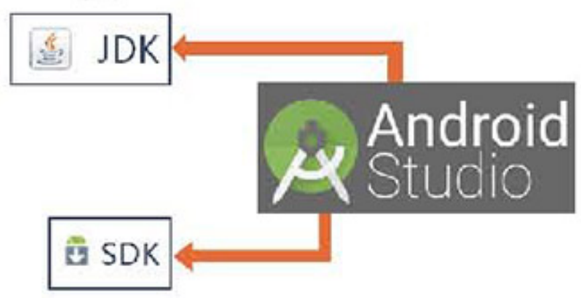


图 1　进行 Android 应用开发所需的软件

在进行 Android 应用开发时，首先需要有 JDK 和 Android SDK 的支持。之后还需要准备合适的开发工具，目前 Android 官方网站推荐的是使用 Android Studio 进行开发。从 Android Studio 2.2 开始，安装 Android Studio 时会自动安装 JDK 和 Android SDK。

说明： JDK（全称为Java Development Kit）是Java开发工具包，包括运行Java程序所必须的JRE（全称为Java Runtime Environment）环境及开发过程中常用的库文件；Android SDK是Android开发工具包，它包括了Android开发相关的API。

二、集成Android开发环境的下载

通常情况下，为了提高开发效率，需要使用相应的开发工具。在 Android 发布初期，推荐使用的开发工具是 Eclipse。2015 年 Android Studio 正式版推出后，Google 公司推荐的 Android 开发工具从 Eclipse 更改为 Android Studio。而且在 Android 的官方网站中，也提供了集成 Android 开发环境的工具包。在该工具包中，不仅包含了开发工具 Android Studio，还包括 JDK 以及最新版本的 Android SDK。下载并安装该工具包后，就可以成功地搭建好 Android 的开发环境。

说明： Android Studio是基于IntelliJ IDEA的Android开发环境。实际上，IntelliJ IDEA是一款非常优秀的Java IDE工具，只是由于它是一款商业工具软件，而且技术文档也少之又少，所以应用并不是很广泛。现在，Google在IntelliJ IDEA的基础上推出的Android Studio则是完全免费的，同时又有Google的技术，相信Android Studio一定会成为开发Android应用的最佳工具。

在 Android 的官方网站中，可以很方便地下载到集成 Android 开发环境的工具包。对于 Android 的官方网址，主要有两个，一个是英文版的“http://www.android.com/”；另一个是中文版的“https://developer.android.google.cn/”。下面以中文版官方网站为例介绍集成 Android 开发环境的下载步骤。

（1）打开浏览器（推荐使用 Google Chrome 浏览器），进入中文版 Android 官方网站，地址是“https://developer.android.google.cn/”，如图 2 所示。

图 2　中文版 Android 官方网站首页

注意 Android 官方网站如有打不开的情况下，可以通过云盘（https://pan.baidu.com/s/1dFIQCCT）下载已经配置好的工具包，或者使用国内的代理 IP。

（2）单击“获取 Android Studio”超链接，进入到下载 Android Studio 和 SDK 工具页面。在该页面中，可以直接单击“下载 ANDROID STUDIO 2.3.1 FOR WINDOWS（1,876 MB）”按钮，下载包含 Android SDK 的完整版的开发工具，如图 3 所示。也可以向下滚动页面，找到“选择其他平台”的表格，下载包含 Android SDK 的完整版的开发工具。

（3）单击“下载 ANDROID STUDIO 2.3.1 FOR WINDOWS（1,876 MB）”按钮，将进入到接受许可协议页面，在该页面中，选中“我已阅读并同意上述条款及条件”复选框，此时“下载 ANDROID STUDIO FOR WINDOWS”按钮将变为可用状态，如图 4 所示。

图 3　下载 Android Studio 和 SDK 工具页面

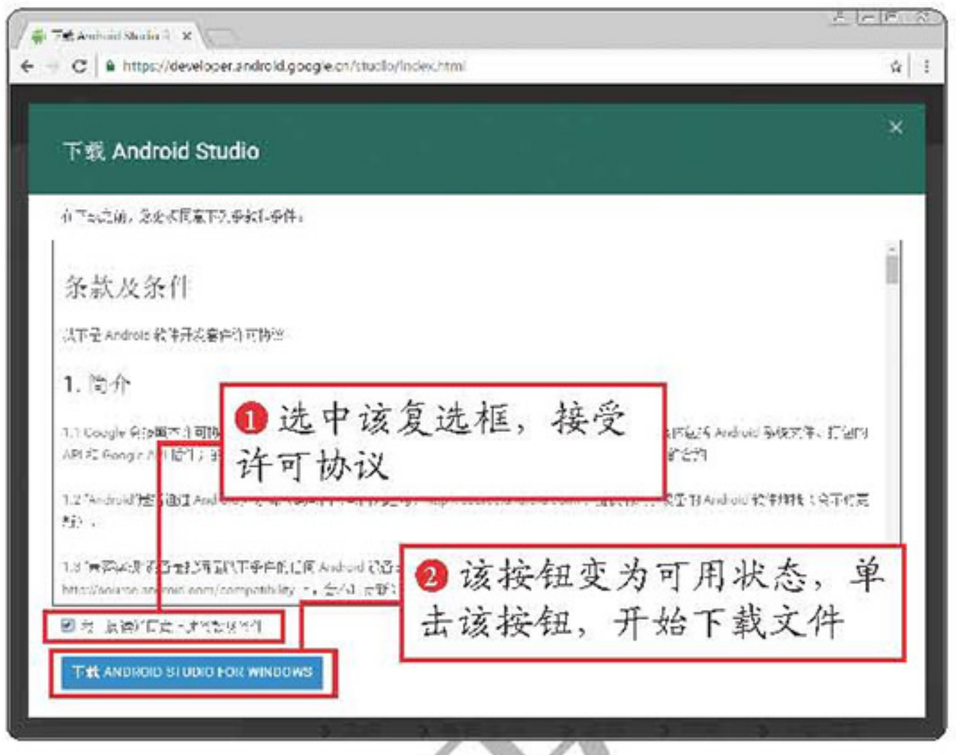


图 4　接受许可协议页面

（4）单击“下载 ANDROID STUDIO FOR WINDOWS”按钮，将开始下载 Windows 系统下的 Android 集成开发环境，同时页面会跳转到“安装说明”页面，在该页面中，可以查看安装 Android Studio 的基本步骤。如图 5 所示。

图 5　正在下载 Android 集成开发环境

（5）下载完成后，浏览器会自动提示“此类型的文件可能会损害您的计算机。您仍然要保留 android-studio-….exe 吗？”，此时，单击“保留”按钮，保留该文件即可。

（6）下载完成后，将得到一个名称为“android-studio-bundle-162.3871768-windows.exe”的安装文件。

三、集成Android开发环境的安装

在 Windows 操作系统下，安装 Android 集成开发环境前，需要先检测电脑的 BIOS 中 Intel Virtualization Technology 是否启用，如果没有启用需要先启用它。

> **说明：**每台电脑的BIOS版本不同，开启状态的位置也会有所不同，这里不进行具体介绍，请读者根据自己电脑BIOS版本找到相应的开启位置即可。如果不清楚电脑如何开启Intel Virtualization Technology，可以到网络上搜索具体方法。

集成 Android 开发环境的安装步骤如下。

（1）双击下载后得到的安装文件 android-studio-bundle-162.3871768-windows.exe，将会弹出“打开文件 - 安全警告”对话框，单击“运行”按钮，将显示如图 6 所示的加载进度框。

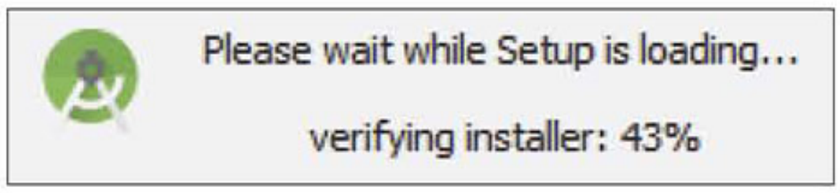


图 6　加载进度框

（2）加载完成后，将进入到如图 7 所示的安装欢迎对话框。

说明：如果以前安装过旧版本的Android Studio，将弹出卸载旧版本对话框，在该对话框中，勾选“Uninstall the previous version”复选框（表示删除旧版本），如图8所示。然后单击“Next”按钮，将打开如图7所示的欢迎安装页面。

图 7　欢迎安装页面

图 8　是否卸载旧版本

（3）在如图 7 所示的欢迎安装页面中，单击“Next”按钮，将打开选择安装组件对话框，在该对话框中采用默认设置，如图 9 所示。

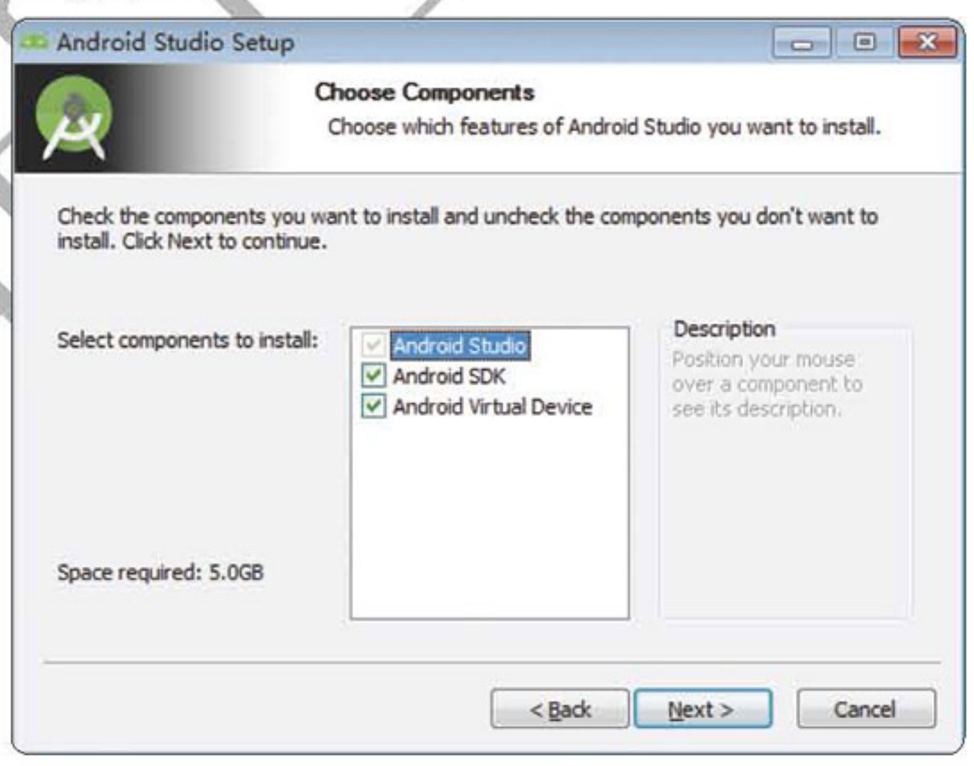


图 9　选择安装组件对话框

（4）单击“Next”按钮，将打开如图 10 所示的接受 Android SDK 的许可协议对话框，单击“I Agree”按钮，接受许可协议。

（5）将进入到配置安装路径对话框，在该对话框中，指定 Android Studio 的安装路径（F:\android\AndroidStudio\2.3.1\as）和 Android SDK 的路径（F:\android\AndroidStudio\2.3.1\sdk），如图 11 所示。

图 10　接受 Android SDK 的许可协议对话框

图 11　配置安装路径对话框

注意

在配置 Android Studio 和 Android SDK 的安装路径时，均需要指定一个独立的文件夹，并且这两个文件还必须是同级的，而且 Android SDK 的保存文件夹只能是 sdk。另外，在配置 SDK 安装路径时，不能包含“?”“%”“*”“:”“|”“"”“<”“>”“!”“;”空格和中文等字符，而在配置 Android Studio 的安装路径时，则可以有空格。

（6）单击“Next”按钮，将进入到选择开始菜单文件夹对话框，在该对话框中，选择 Android Studio 的快捷方式创建在开始菜单中的文件夹下，默认为新创建的 Android Studio 文件夹，这里采用默认设置，如图 12 所示。

（7）单击“Install”按钮，将显示如图 13 所示的安装进度对话框，此时需要等待一段时间。

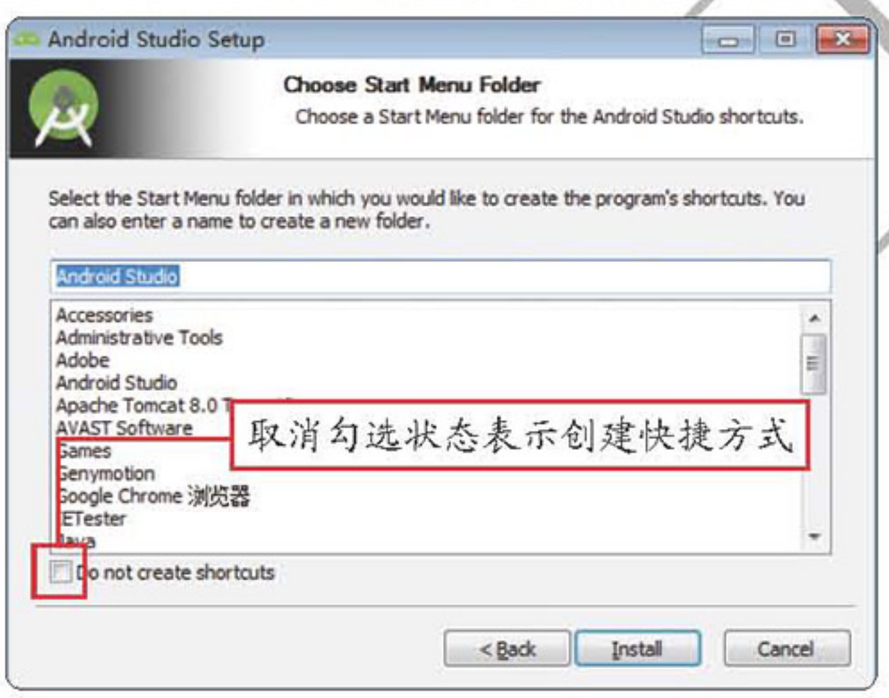


图 12　选择开始菜单文件夹对话框

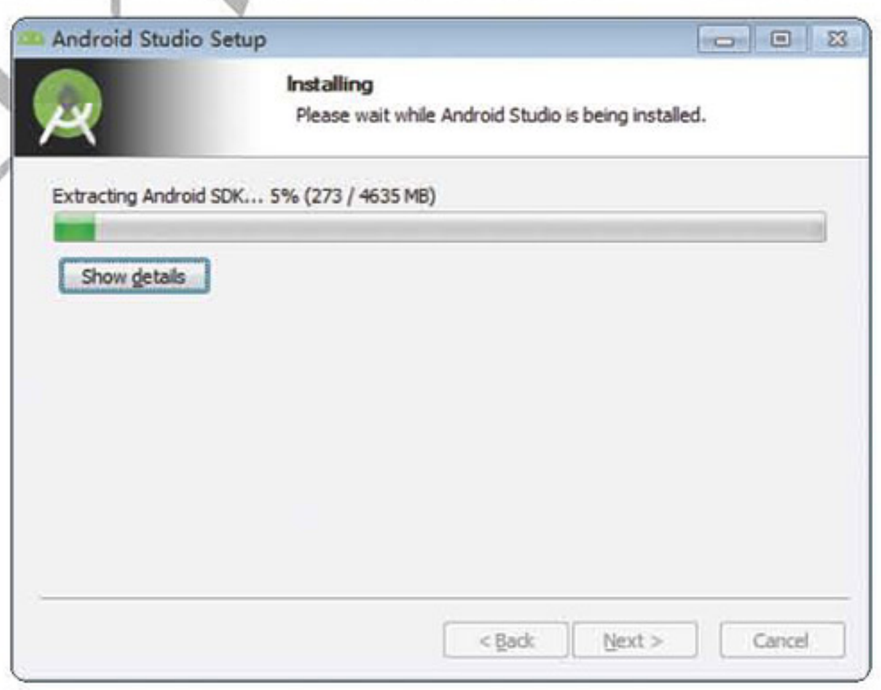


图 13　安装进度对话框

说明：在图12中，勾选“Do not create shortcuts”复选框，将不创建快捷方式。

（8）安装完成后，将显示如图 14 所示的安装完成对话框。单击“Next”按钮，将弹出如图 15 所示的对话框。在该对话框中，直接单击“Finish”按钮完成 Android Studio 的安装，并且开启 Android Studio。

说明：在图14所示的对话框中，也可以取消“Start Android Studio”复选框的勾选状态，然后再单击“Finish”按钮，完成Android Studio的安装。

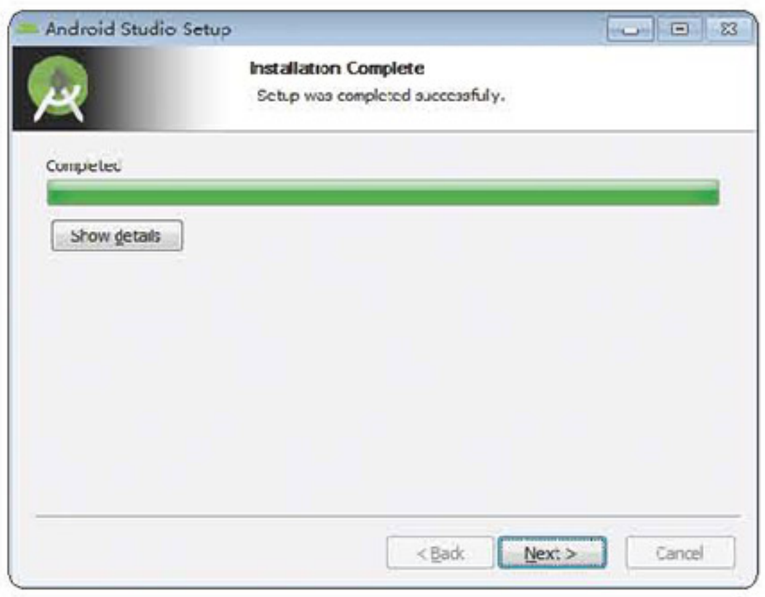


图 14　安装成功对话框

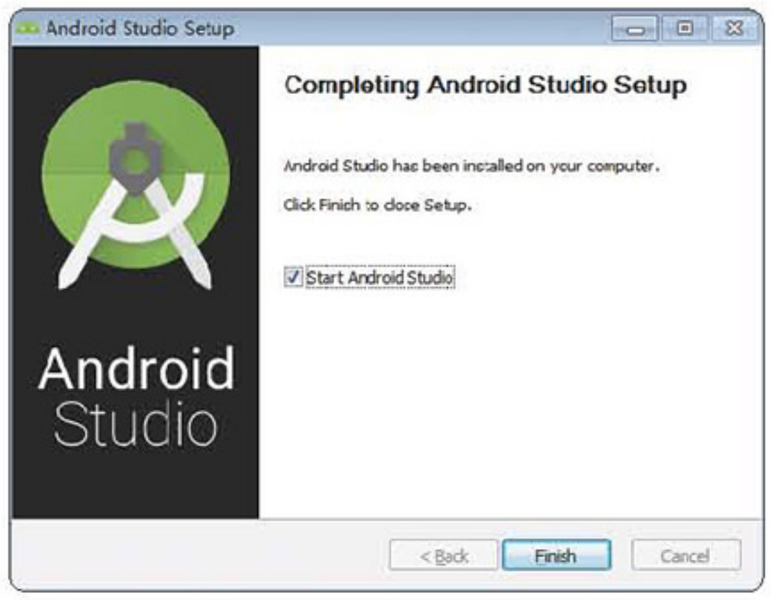


图 15　安装完成对话对话框

（9）启动 Android Studio，将弹出如图 16 所示的对话框（如果电脑中以前安装过 Android Studio，可能出现如图 17 所示的带 3 个单选按钮的对话框），该对话框用于指定是否从以前版本的 Android Studio 导入设置。这里都选中“I do not have a previous version of Studio or I do not want to import my settings”单选按钮，不导入任何设置。

图 16　询问是否导入设置对话框

图 17　以前安装过 Android Studio 弹出的询问对话框

（10）单击 OK 按钮，继续启动 Android Studio，此时会弹出如图 18 所示的对话框，该对话框用于询问是否设置代理，如果您有有效的代理地址，可以单击 Setup Proxy 按钮，添加代理地址，否则直接单击 Cancel（取消）按钮。这里直接单击 Cancel 按钮。

图 18　询问是否设置代理

（11）显示如图 19 所示的欢迎对话框。如果想要检测需要更新或者安装的组件或者改变默认的主题方案，可以单击“Next”按钮，否则直接单击“Cancel”按钮，这里单击“Cancel”按钮，不进行安装。

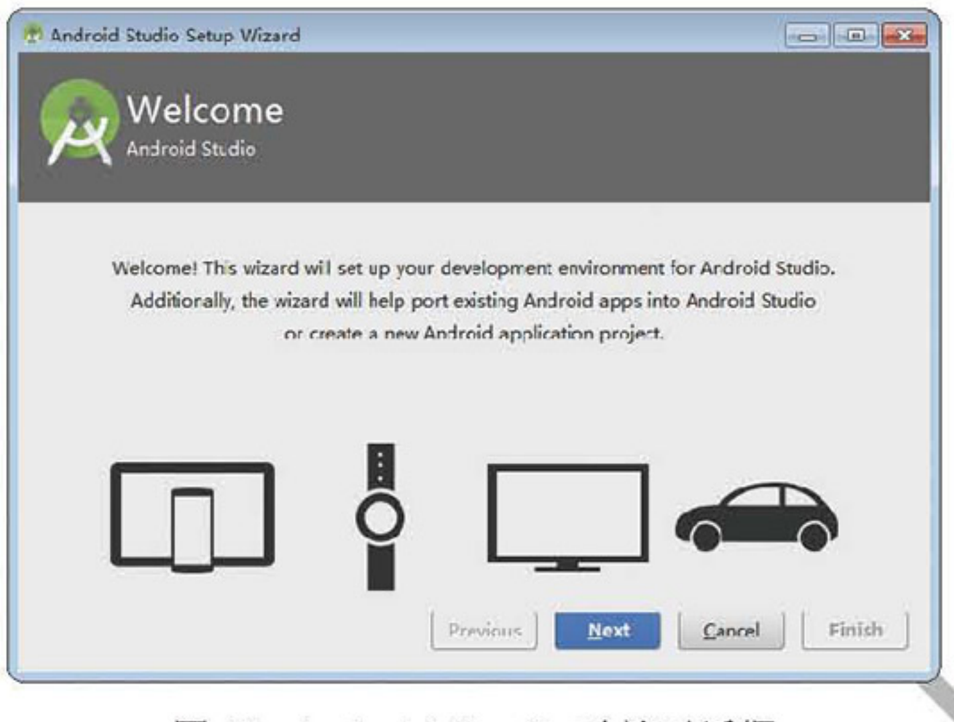


图 19　Android Studio 欢迎对话框

（12）在弹出的跳过安装向导对话框中，选中“Re-run the setup wizard on the next Android Studio startup(Recommended)”单选按钮，表示跳过安装向导，下次启动 Android Studio 时再次显示安装向导；选中“Do not re-run the setup wizard”单选按钮表示跳过安装向导，不再提示。这里选中后者。如图 20 所示。

图 20　跳过安装向导对话框

（13）单击“OK”按钮，显示如图 21 所示的欢迎对话框，这时就表示 Android Studio 已经安装完毕，并且启动成功。

图 21　欢迎对话框

和外星人一起学

训练·应用

扫描后输入学习码，
获得在线课程资源

激活学习码网址：
http://www.mingrisoft.com/key.html

责任编辑：张宏亮
美术设计：明日科技

分类建议：计算机/程序设计

ISBN 978-7-5692-0453-7

定价：89.80元

（附2DVD，含视频讲解，实例源码，资源文件及代码查错器）