Netkiller Android 手札

目录

- 自述
  - 1. 写给读者
  - 2. 作者简介
  - 3. 如何获得文档
  - 4. 打赏（Donations）
  - 5. 联系方式
- 1. Android Studio
  - 1. 卸载 Android Studio
  - 2. 代码格式化
  - 3. 设置兼容最低SDK版本
  - 4. SDK Tools
    - 4.1. 接受 License
    - 4.2. 查看 SDK 列表
    - 4.3. 按照 Android SDK
  - 5. 命令行操作
  - 6. adb 命令
    - 6.1. 设备管理
    - 6.2. Shell
      - 网络相关
        - 查看 IP 地址
        - 无线 IP 地址
        - Mac 地址
      - 内存信息
      - 查看硬件与系统属性
    - 6.3. 设备 ID
      - 获取变量
      - 设置变量
        - 显示/关闭虚拟键
    - 6.4. 查看安卓版本
      - 产品型号

6.5. Logcat
6.6. 上传文件
6.7. 下载文件
6.8. 安卓 .apk bk
6.9. 屏幕尺寸
6.10. dump 系统信息
电池信息
6.11. 解锁
2. AndroidManifest.xml
1. SDK 版本配置
2. 开启网络
3. 文件存储权限
4. 相机权限
5. GPS 定位权限
6. 全屏-无标题
7. 设置为默认开机启动
3. 设备
1. 环境变量
1.1. 扩展存储
1.2. 下载缓存目录
1.3. 数据目录
2. 配置文件
2.1. *.properties 文件
2.2. 再 AndroidManifest.xml 使用 meta-data element 定义
2.3. 再 build.gradle 文件中配置 productFlavors
2.4. 从 assets 目录读取配置文件
配置文件例子
3. 设备信息
4. 声卡
4.1. 播放
4.2. 录音
4.3. 查看声卡信息
4.4. /proc/asound 设备信息
4.5. 查看声卡当前占用设备
4.6. tinymix 设置声卡参数

- 4.7. 麦克风阵列调试
  - 录音测试
4. Activity
  1. 定义 UI
  2. 隐藏虚拟键
  3. 显式四种跳转方式
  4. 定时关闭
  5. 恢复触发
  6. 返回触发
  7. startActivity()
  8. Activity 间数据传递
    - 8.1. Intent 方式
    - 8.2. Bundle 方式
    - 8.3. Flag 属性
    - 8.4. 返回值
  9. intentActivityResultLauncher 跳转
  10. startActivityForResult 替代方案
  11. Fragment
    - 11.1. 在 Fragment 中使用 findViewById
    - 11.2. 在 Fragment 中使用 Intent 跳转
    - 11.3. Fragment 中调用 getPackageManager()
    - 11.4. 在 Fragment 中使用 runOnUiThread
5. Palette 视觉设计
  1. 样式布局
    - 1.1. 动画
    - 1.2. 声音波形图
  2. Widgets
    - 2.1. ImageView
      - 剧中效果
      - ImageView 显示 URL 图片
      - 唱片播放效果（旋转PNG图片）
        - UI 布局
        - 旋转动画效果文件
        - 启动旋转效果
  3. Legacy
    - 3.1. GardView

        3.2. GridView
7. Schedule 计划任务
    1. Time 和 TimerTask 定时刷新
    2. 使用 Runnable 和 Handler 实现定时执行
    3. TimerTask 更新 UI
8. Internationalization i18n with Android (国际化)
    1. 创建国际化文件
    2. strings.xml 文件
    3. 翻译语言
    4. 引用国际化文件
    5. 切换语言
9. 存储
    1. SharedPreferences
        1.1. 操作模式
        1.2. 保存数据
        1.3. 读取数据
        1.4. 通过 key 查询数据是否存在
        1.5. 删除数据
        1.6. 清空数据
        1.7. 对象存储
        1.8. SharedPreferences 读取物理存储文件
    2. SD Card
        2.1. SD Card 状态
        2.2. Android 11 申请 sdcard 权限
10. 网络
    1. Wifi 配置
    2. OkHttp - An HTTP & HTTP/2 client for Android and Java applications
        2.1. Gradle
        2.2. AndroidManifest.xml 开启网络访问权限
        2.3. okhttp 默认是 HTTPS 开启 HTTP
        2.4. GET
            URL 组装
            Get 异步调用
        2.5. POST
            POST Form Data

POST RAW JSON
数据流提交
2.6. http header 相关设置
设置 HTTP 头
Cookie 管理
禁用缓存
设置缓存 max-age
强制缓存
2.7. HTTP Base Auth
2.8. HttpUrl.Builder 组装 URL 地址参数
2.9. Android Activity Example
2.10. Android Oauth2 + Jwt example
2.11. HTTP/2
2.12. 异步更新 UI
11. 相机与相册
1. manifest 文件
2. layout
3. Activity
4. LED flash 做手电筒
12. 麦克风与录音
1. 开启麦克风和SD卡权限
2. layout
3. Activity
13. 多媒体开发
1. MediaPlayer
1.1. 播放Raw下的元数据
1.2. 播放assets文件夹中的音乐
1.3. 播放互联网音乐
1.4. 使用单例模式
2. VideoView 开发
2.1. 播放网络视频
2.2. MediaController 添加翻页事件
2.3. 静音播放视频
2.4. 更新进度条
2.5. 完整的例子
2.6. 循环播放

        2.7. 静音播放
    3. SoundPool
    4. 音量控制
    5. SurfaceView
    6. Vitamio
14. 定位
    1. GPS + 网络 定位
        1.1. manifest 权限配置
        1.2. layout
        1.3. Activity
    2. 只从 GPS 获取定位
15. 电话
    1. SIM 卡状态
    2. 通信录与拨打电话
    3. 发送短信
16. 消息广播
    1. 动态注册
    2. 静态注册
    3. 自定义用户消息广播
    4. 本地广播
17. Service 服务
    1. Service的基本用法
        1.1. manifest 文件
        1.2. 创建 Service
        1.3. Layout 代码
        1.4. Activity 代码
    2. Service 中启动线程
    3. Service 和 Activity 通信
        3.1. Layout
        3.2. Service
        3.3. Activity
    4. Service 和 Toast
    5. Service 中启动 Activity
    6. Service 中更新 UI
18. Notification 通知中心
    1. 文本通知

    2. 添加点击操作
19. NFC (Near field communication)
    1. AndroidManifest.xml 文件配置
    2. Loyout 文件
    3. Activity 文件
20. EventBus
    1. 添加 EventBus 依赖到项目Gradle文件
    2. 快速开始一个演示例子
        2.1. 创建 MessageEvent 类
        2.2. Layout
        2.3. Activity
    3. Sticky Events
        3.1. MainActivity
        3.2. StickyActivity
        3.3. MessageEvent
        3.4. 删除粘性事件
    4. 线程模型
    5. 配置 EventBus
    6. 事件优先级
    7. 捕获异常事件
21. 图形开发
    1. Paint
    2. AnimationDrawable
22. Android Things
    1. GPIO
23. Android MQTT
    1. build.gradle 添加依赖包
    2. AndroidManifest.xml
    3. Android Mqtt v5 例子
24. 安卓开发版
    1. rk3568
        1.1. 声卡
25. 杂项
    1. Sleep
    2. Caused by: java.net.UnknownServiceException: CLEARTEXT communication to 47.100.253.187 not

permitted by network security policy
    3. 设计模式
        3.1. 单例模式
    4. Android OS 包
        4.1. 进程ID
        4.2. handler
    5. fastjson android
        5.1. 对象转字符串
        5.2. JsonObject 转对象
        5.3. 字符串 与 json 互转
        5.4. json 转 数组
        5.5. JSON数组转List
        5.6. Map 与 Json 互转
    6. Butter Knife
26. FAQ
    1. java.net.UnknownServiceException: CLEARTEXT communication to 192.168.0.185 not permitted by network security policy
    2. Caused by: android.os.NetworkOnMainThreadException
    3. java.lang.IllegalStateException: Player is accessed on the wrong thread.
27. 讯飞云
    1. AIUI
        1.1. AIUIPlayer
        1.2. 酷我音乐
            获取音乐URL

Netkiller Android 手札

http://www.netkiller.cn/android/index.html

Mr. Neo Chan, 陈景峯(BG7NYT)

中国广东省深圳市望海路半岛城邦三期
518067
+86 13113668890

<netkiller@msn.com>

电子书最近一次更新于 2023-09-01 20:23:23

版权 © 2018-2019 Neo Chan

版权声明

转载请与作者联系，转载时请务必标明文章原始出处和作者信息及本声明。

http://www.netkiller.cn
http://netkiller.github.io
http://netkiller.sourceforge.net
微信公众号: netkiller
微信：13113668890 请注明“读者”
QQ：13721218 请注明“读者”
QQ群：128659835 请注明“读者”
知乎专栏 | 多维度架构

2018-10

我的系列文档

编程语言

Netkiller Architect 手札	Netkiller Developer 手札	Netkiller Java 手札	Netkiller Spring 手札	Netkiller PHP 手札	Netkiller Python 手札
Netkiller Testing 手札	Netkiller Cryptography 手札	Netkiller Perl 手札	Netkiller Docbook 手札	Netkiller Project 手札	Netkiller Database 手札

致读者

Netkiller 系列手札 已经被 Github 收录，并备份保存在北极地下250米深的代码库中，备份会保留1000年。

Preserving open source software for future generations

The world is powered by open source software. It is a hidden cornerstone of modern civilization, and the shared heritage of all humanity.

The GitHub Arctic Code Vault is a data repository preserved in the Arctic World Archive (AWA), a very-long-term archival facility 250 meters deep in the permafrost of an Arctic mountain.

We are collaborating with the Bodleian Library in Oxford, the Bibliotheca Alexandrina in Egypt, and Stanford Libraries in California to store copies of 17,000 of GitHub's most popular and most-depended-upon projects—open source's "greatest hits"—in their archives, in museum-quality cases, to preserve them for future generations.

https://archiveprogram.github.com/arctic-vault/

自述

《Netkiller 系列 手札》是一套免费系列电子书，netkiller 是 nickname 从1999 开使用至今，“手札” 是札记，手册的含义。

2003年之前我还是以文章形式在BBS上发表各类技术文章，后来发现文章不够系统，便尝试写长篇技术文章加上章节目录等等。随着内容增加，不断修订，开始发布第一版，第二版.....

IT知识变化非常快，而且具有时效性，这样发布非常混乱，经常有读者发现第一版例子已经过时，但他不知道我已经发布第二版。

我便有一种想法，始终维护一个文档，不断更新，使他保持较新的版本不过时。

第一部电子书是《PostgreSQL 实用实例参考》开始我使用 Microsoft Office Word 慢慢随着文档尺寸增加 Word 开始表现出力不从心。

我看到PostgreSQL 中文手册使用SGML编写文档，便开始学习 Docbook SGML。使用Docbook写的第一部电子书是《Netkiller Postfix Integrated Solution》这是Netkiller 系列手札的原型。

至于“手札”一词的来历，是因为我爱好摄影，经常去一个台湾摄影网站，名字就叫“摄影家手札”。

由于硬盘损坏数据丢失 《Netkiller Postfix Integrated Solution》 的 SGML文件已经不存在; Docbook SGML存在很多缺陷 UTF-8支持不好，转而使用Docbook XML.

目前技术书籍的价格一路飙升，动则￥80，￥100，少则￥50，￥60. 技术书籍有时效性，随着技术的革新或淘汰，大批书记成为废纸垃圾。并且这些书技术内容雷同，相互抄袭，质量越来越差，甚至里面给出的例子错误百出，只能购买影印版，或者翻译的版本。

在这种背景下我便萌生了自己写书的想法，资料主要来源是我的笔记与例子。我并不想出版，只为分享，所有我制作了基于CC License 发行的系列电子书。

本书注重例子，少理论（捞干货），只要你对着例子一步一步操作，就会成功，会让你有成就感并能坚持学下去，因为很多人遇到障碍就会放弃，其实我就是这种人，只要让他看到希望，就能坚持下去。

1. 写给读者

为什么写这篇文章

有很多想法,工作中也用不到所以未能实现，所以想写出来,和大家分享.有一点写一点,写得也不好,只要能看懂就行,就当学习笔记了.

开始零零碎碎写过一些文档，也向维基百科供过稿，但维基经常被ZF封锁，后来发现sf.net可以提供主机存放文档，便做了迁移。并开始了我的写作生涯。

这篇文档是作者20年来对工作的总结,是作者一点一滴的积累起来的，有些笔记已经丢失，所以并不完整。

因为工作太忙整理比较缓慢。目前的工作涉及面比较窄所以新文档比较少。

我现在花在技术上的时间越来越少，兴趣转向摄影，无线电。也想写写摄影方面的心得体会。

写作动力:

曾经在网上看到外国开源界对中国的评价，中国人对开源索取无度，但贡献却微乎其微.这句话一直记在我心中，发誓要为中国开源事业做我仅有的一点微薄贡献

另外写文档也是知识积累，还可以增加在圈内的影响力.

人跟动物的不同,就是人类可以把自己学习的经验教给下一代人.下一代在上一代的基础上再创新,不断积累才有今天.

所以我把自己的经验写出来,可以让经验传承

没有内容的章节:

目前我自己一人维护所有文档，写作时间有限，当我发现一个好主题就会加入到文档中，待我有时间再完善章节，所以你会发现很多章节是空无内容的.

文档目前几乎是流水帐试的写作，维护量很大，先将就着看吧.

我想到哪写到哪,你会发现文章没一个中心,今天这里写点,明天跳过本章写其它的.

文中例子绝对多,对喜欢复制然后粘贴朋友很有用,不用动手写,也省时间.

理论的东西,网上大把,我这里就不写了,需要可以去网上查.

我爱写错别字,还有一些是打错的,如果发现请指正.

文中大部分试验是在Debian/Ubuntu/Redhat AS上完成.

写给读者

至读者:

我不知道什么时候，我不再更新文档或者退出IT行业去从事其他工作，我必须给这些文档找一个归宿，让他能持续更新下去。

我想捐赠给某些基金会继续运转，或者建立一个团队维护它。

我用了20年时间坚持不停地写作，持续更新，才有今天你看到的《Netkiller 手扎》系列文档，在中国能坚持20年，同时没有任何收益的技术类文档，是非常不容易的。

有很多时候想放弃，看到外国读者的支持与国内社区的影响，我坚持了下来。

中国开源事业需要各位参与，不要成为局外人，不要让外国人说：中国对开源索取无度，贡献却微乎其微。

我们参与内核的开发还比较遥远，但是进个人能力，写一些文档还是可能的。

系列文档

下面是我多年积累下来的经验总结，整理成文档供大家参考:

Netkiller Architect 手札

Netkiller Developer 手札

Netkiller PHP 手札

Netkiller Python 手札

Netkiller Testing 手札

Netkiller Cryptography 手札
Netkiller Linux 手札
Netkiller FreeBSD 手札
Netkiller Shell 手札
Netkiller Security 手札
Netkiller Web 手札
Netkiller Monitoring 手札
Netkiller Storage 手札
Netkiller Mail 手札
Netkiller Docbook 手札
Netkiller Version 手札
Netkiller Database 手札
Netkiller PostgreSQL 手札
Netkiller MySQL 手札
Netkiller NoSQL 手札
Netkiller LDAP 手札
Netkiller Network 手札
Netkiller Cisco IOS 手札
Netkiller H3C 手札
Netkiller Multimedia 手札
Netkiller Management 手札
Netkiller Spring 手札
Netkiller Perl 手札
Netkiller Amateur Radio 手札

2. 作者简介

陈景峯 (ㄔㄣˊ ㄐㄧㄥˇ ㄈㄥ)

Nickname： netkiller | English name: Neo chen | Nippon name: ちんけいほう (音訳) | Korean name: 천징봉 | Thailand name: ภูมิภาพภูเขา | Vietnam: Tr`ần Cảnh Phong

Callsign: BG7NYT | QTH: ZONE CQ24 ITU44 ShenZhen, China

程序猿，攻城狮，挨踢民工, Full Stack Developer, UNIX like Evangelist, 业余无线电爱好者（呼号：BG7NYT）,户外运动，山地骑行以及摄影爱好者。

《Netkiller 系列 手札》的作者

成长阶段

1981年1月19日(庚申年腊月十四)出生于黑龙江省青冈县建设乡双富大队第一小队

1989年9岁随父母迁居至黑龙江省伊春市，悲剧的天朝教育，不知道那门子归定，转学必须降一级，我本应该上一年级，但体制让我上学前班，那年多都10岁了

1995年小学毕业，体制规定借读要交3000两银子(我曾想过不升初中)，亲戚单位分楼告别平房，楼里没有地方放东西，把2麻袋书送给我，无意中发现一本电脑书BASIC语言，我竟然看懂了，对于电脑知识追求一发而不可收，后面顶零花钱，压岁钱主要用来买电脑书《MSDOS 6.22》《新编Unix实用大全》《跟我学Foxbase》。。。。。。

1996年第一次接触UNIX操作系统，BSD UNIX, Microsoft Xinux(盖茨亲自写的微软Unix，知道的人不多)

1997年自学Turbo C语言，苦于没有电脑，后来学校建了微机室才第一次使用QBASIC(DOS 6.22 自带命令)，那个年代只能通过软盘拷贝转播，Trubo C编译器始终没有搞到，

1997年第一次上Internet网速只有9600Bps, 当时全国兴起各种信息港域名格式是www.xxxx.info.net, 访问的第一个网站是NASA下载了很多火星探路者拍回的照片，还有“淞沪”sohu的前身

1998~2000年在哈尔滨学习计算机，充足的上机时间，但老师让我们练打字（明伦五笔/WT）打字不超过80个/每分钟还要强化训练，不过这个给我的键盘功夫打了好底。

1999年学校的电脑终于安装了光驱，在一张工具盘上终于找到了Turbo C, Borland C++与Quick Basic编译器，当时对VGA图形编程非常感兴趣，通过INT33中断控制鼠标，使用绘图函数模仿windows界面。还有操作 UCDOS 中文字库，绘制矢量与点阵字体。

2000年沉迷于Windows NT与Back Office各种技术，神马主域控制器，DHCP，WINS，IIS，域名服务器，Exchange邮件服务器，MS Proxy, NetMeeting...以及ASP+MS SQL开发；用56K猫下载了一张LINUX。ISO镜像，安装后我兴奋的24小时没有睡觉。

职业生涯

2001 年来深圳进城打工,成为一名外来务工者. 在一个4人公司做PHP开发，当时PHP的版本是2.0, 开始使用Linux Redhat 6.2.当时很多门户网站都是用FreeBSD,但很难搞到安装盘，在网易社区认识了一个网友,从广州给我寄了一张光盘，FreeBSD 3.2

2002 年我发现不能埋头苦干,还要学会"做人".后辗转广州工作了半年，考了一个Cisco CCNA认证。回到深圳重新开始，在车公庙找到一家工作做Java开发

2003 年这年最惨,公司拖欠工资16000元,打过两次官司2005才付清.

2004 年开始加入分布式计算团队,目前成绩，工作仍然是Java开发并且开始使用PostgreSQL数据库。

2004-10月开始玩户外和摄影

2005-6月成为中国无线电运动协会会员,呼号BG7NYT,进了一部Yaesu FT-60R手台。公司的需要转回PHP与MySQL，相隔几年发现PHP进步很大。在前台展现方面无人能敌，于是便前台使用PHP，后台采用Java开发。

2006 年单身生活了这么多年,终于找到归宿. 工作更多是研究PHP各种框架原理

2007 物价上涨,金融危机，休息了4个月（其实是找不到工作），关外很难上439.460中继，搞了一台Yaesu FT-7800.

2008 终于找到英文学习方法， 《Netkiller Developer 手札》，《Netkiller Document 手札》

2008-8-8 08:08:08 结婚,后全家迁居湖南省常德市

2009《Netkiller Database 手札》,2009-6-13学车，年底拿到C1驾照

2010 对电子打击乐产生兴趣，计划学习爵士鼓。由于我对Linux热爱，我轻松的接管了公司的运维部，然后开发运维两把抓。我印象最深刻的是公司一次上架10个机柜，我们用买服务器纸箱的钱改善伙食。我将40多台服务器安装BOINC做压力测试，获得了中国第二的名次。

2011 平凡的一年，户外运动停止，电台很少开，中继很少上，摄影主要是拍女儿与家人，年末买了一辆山地车

2012 对油笔画产生了兴趣，活动基本是骑行银湖山绿道，

2013 开始学习民谣吉他，同时对电吉他也极有兴趣；最终都放弃了。这一年深圳开始推数字中继2013-7-6日入手Motorola

MOTOTRBO XIR P8668，Netkiller 系列手札从Sourceforge向Github迁移；年底对MYSQL UDF，Engine与PHP扩展开发产生很浓的兴趣，拾起遗忘10+年的C，写了几个mysql扩展（图片处理，fifo管道与ZeroMQ），10月份入Toyota Rezi 2.5V并写了一篇《攻城狮的苦逼选车经历》

2014-9-8 在淘宝上买了一架电钢琴 Casio Privia PX-5S pro 开始陪女儿学习钢琴，由于这家钢琴是合成器电钢，里面有打击乐，我有对键盘鼓产生了兴趣。

2014-10-2号罗浮山两日游，对中国道教文化与音乐产生了兴趣，10月5号用了半天时间学会了简谱。10月8号入Canon 5D Mark III + Canon Speedlite 600EX-RT香港过关被查。

2014-12-20号对乐谱制作产生兴趣（https://github.com/SheetMusic/Piano），给女儿做了几首钢琴伴奏曲，MuseScore制谱然后生成MIDI与WAV文件。

2015-09-01 晚饭后拿起爵士鼓基础教程尝试在Casio Privia PX-5S pro演练，经过反复琢磨加上之前学钢琴的乐理知识，终于在02号晚上，打出了简单的基本节奏，迈出了第一步。

2016 对弓箭（复合弓）产生兴趣，无奈天朝法律法规不让玩。每周游泳轻松1500米无压力，年底入 xbox one s 和 Yaesu FT-2DR, 同时开始关注功放音响这块

2017 7月9号入 Yamaha RX-V581 功放一台，连接Xbox打游戏爽翻了，入Kindle电子书，计划学习蝶泳，果断放弃运维和开发知识体系转攻区块链。

2018 从溪山美地搬到半岛城邦，丢弃了多年攒下的家底。11 月开始玩 MMDVM，使用 Yaesu FT-7800 发射，连接MMDVM中继板，树莓派，覆盖深圳湾，散步骑车通联两不误。

2019 卖了常德的房子，住了5次院，哮喘反复发作，决定停止电子书更新，兴趣转到知乎，B站

2020 准备找工作

职业生涯路上继续打怪升级

3. 如何获得文档

下载 Netkiller 手札 (epub,kindle,chm,pdf)
EPUB
https://github.com/netkiller/netkiller.github.io/tree/master/download/epub
MOBI
https://github.com/netkiller/netkiller.github.io/tree/master/download/mobi
PDF
https://github.com/netkiller/netkiller.github.io/tree/master/download/pdf
CHM
https://github.com/netkiller/netkiller.github.io/tree/master/download/chm

通过 GIT 镜像整个网站

https://github.com/netkiller/netkiller.github.com.git

$ git clone https://github.com/netkiller/netkiller.github.com.git

镜像下载

整站下载

```
wget -m http://www.netkiller.cn/index.html
```

指定下载

```
wget -m wget -m http://www.netkiller.cn/linux/index.html
```

Yum 下载文档

获得光盘介质，RPM包，DEB包，如有特别需要，请联系我

YUM 在线安装电子书

http://netkiller.sourceforge.net/pub/repo/

```
# cat >> /etc/yum.repos.d/netkiller.repo <<EOF
[netkiller]
name=Netkiller Free Books
baseurl=http://netkiller.sourceforge.net/pub/repo/
enabled=1
gpgcheck=0
gpgkey=
EOF
```

查找包

```
# yum search netkiller

netkiller-centos.x86_64 : Netkiller centos Cookbook
netkiller-cryptography.x86_64 : Netkiller cryptography
Cookbook
netkiller-docbook.x86_64 : Netkiller docbook Cookbook
netkiller-linux.x86_64 : Netkiller linux Cookbook
netkiller-mysql.x86_64 : Netkiller mysql Cookbook
netkiller-php.x86_64 : Netkiller php Cookbook
netkiller-postgresql.x86_64 : Netkiller postgresql Cookbook
netkiller-python.x86_64 : Netkiller python Cookbook
netkiller-version.x86_64 : Netkiller version Cookbook
```

安装包

```
yum install netkiller-docbook
```

4. 打赏（Donations）

If you like this documents, please make a donation to support the authors' efforts. Thank you!

您可以通过微信，支付宝，贝宝给作者打赏。

银行(Bank)

招商银行(China Merchants Bank)

开户名：陈景峰

账号：9555500000007459

微信（Wechat）

支付宝（Alipay）

PayPal Donations

https://www.paypal.me/netkiller

5. 联系方式

主站 http://www.netkiller.cn/

备用 http://netkiller.github.io/

繁体网站 http://netkiller.sourceforge.net/

联系作者

Mobile: +86 13113668890

Email: netkiller@msn.com

QQ群: 128659835 请注明“读者”

QQ: 13721218

ICQ: 101888222

注：请不要问我安装问题！

博客 **Blogger**

知乎专栏 https://zhuanlan.zhihu.com/netkiller

LinkedIn: http://cn.linkedin.com/in/netkiller

OSChina: http://my.oschina.net/neochen/

Facebook: https://www.facebook.com/bg7nyt

Flickr: http://www.flickr.com/photos/bg7nyt/

Disqus: http://disqus.com/netkiller/

solidot: http://solidot.org/~netkiller/

SegmentFault: https://segmentfault.com/u/netkiller

Reddit: https://www.reddit.com/user/netkiller/

Digg: http://www.digg.com/netkiller

Twitter: http://twitter.com/bg7nyt

weibo: http://weibo.com/bg7nyt

Xbox club

我的 xbox 上的ID是 netkiller xbox，我创建了一个俱乐部 netkiller 欢迎加入。

Radio

CQ CQ CQ DE BG7NYT:

如果这篇文章对你有所帮助,请寄给我一张QSL卡片, qrz.cn or qrz.com or hamcall.net

Personal Amateur Radiostations of P.R.China

ZONE CQ24 ITU44 ShenZhen, China

Best Regards, VY 73! OP. BG7NYT

守听频率 DMR 438.460 -8 Color 12 Slot 2 Group 46001

守听频率 C4FM 439.360 -5 DN/VW

MMDVM Hotspot:

Callsign: BG7NYT QTH: Shenzhen, China

YSF: YSF80337 - CN China 1 - W24166/TG46001

DMR: BM_China_46001 - DMR Radio ID: 4600441

第 1 章 Android Studio

1. 卸载 Android Studio

卸载 Mac Android Studio

```
rm -Rf /Applications/Android\ Studio.app
rm -Rf ~/Library/Preferences/AndroidStudio*
rm ~/Library/Preferences/com.google.android.studio.plist
rm -Rf ~/Library/Application\ Support/AndroidStudio*
rm -Rf ~/Library/Logs/AndroidStudio*
rm -Rf ~/Library/Caches/AndroidStudio*
rm -Rf ~/Library/Android*
```

删除 Projects

```
rm -Rf ~/AndroidStudioProjects
```

删除 gradle

```
rm -rf ~/.gradle
```

卸载Android Virtual Devices(AVDs) and *.keystore.

```
rm -Rf ~/.android
```

2. 代码格式化

```
Option + Command + L
```

3. 设置兼容最低SDK版本

Android 9 Pie 的设置方法

打开 build.gradle 修改 minSdkVersion 项，这里的 26 表示 Android 8.0

```
apply plugin: 'com.android.application'

android {
    compileSdkVersion 28
    defaultConfig {
        applicationId "cn.netkiller.video"
        minSdkVersion 26
        targetSdkVersion 28
        versionCode 1
        versionName "1.0"
        testInstrumentationRunner
"android.support.test.runner.AndroidJUnitRunner"
    }
    buildTypes {
        release {
            minifyEnabled false
            proguardFiles getDefaultProguardFile('proguard-
android.txt'), 'proguard-rules.pro'
        }
    }
}

dependencies {
    implementation fileTree(dir: 'libs', include: ['*.jar'])
    implementation 'com.android.support:appcompat-v7:28.0.0'
    implementation 'com.android.support.constraint:constraint-
layout:1.1.3'
    testImplementation 'junit:junit:4.12'
    androidTestImplementation
'com.android.support.test:runner:1.0.2'
    androidTestImplementation
'com.android.support.test.espresso:espresso-core:3.0.2'
```

```
}
```

4. SDK Tools

```
wget https://dl.google.com/android/repository/sdk-tools-linux-
4333796.zip
unzip sdk-tools-linux-4333796.zip
cd tools/
```

查看帮助信息

```
neo@ubuntu:~/tmp/tools$ bin/sdkmanager --help
Usage:
  sdkmanager [--uninstall] [<common args>] [--package_file=
<file>] [<packages>...]
  sdkmanager --update [<common args>]
  sdkmanager --list [<common args>]
  sdkmanager --licenses [<common args>]
  sdkmanager --version

With --install (optional), installs or updates packages.
    By default, the listed packages are installed or (if
already installed)
    updated to the latest version.
With --uninstall, uninstall the listed packages.

    <package> is a sdk-style path (e.g. "build-tools;23.0.0" or
             "platforms;android-23").
    <package-file> is a text file where each line is a sdk-
style path
                    of a package to install or uninstall.
    Multiple --package_file arguments may be specified in
combination
    with explicit paths.

With --update, all installed packages are updated to the latest
version.
```

```
With --list, all installed and available packages are printed
out.

With --licenses, show and offer the option to accept licenses
for all
     available packages that have not already been accepted.

With --version, prints the current version of sdkmanager.

Common Arguments:
    --sdk_root=<sdkRootPath>: Use the specified SDK root
instead of the SDK
                               containing this tool

    --channel=<channelId>: Include packages in channels up to
<channelId>.
                           Common channels are:
                           0 (Stable), 1 (Beta), 2 (Dev), and 3
(Canary).

    --include_obsolete: With --list, show obsolete packages in
the
                        package listing. With --update, update
obsolete
                        packages as well as non-obsolete.

    --no_https: Force all connections to use http rather than
https.

    --proxy=<http | socks>: Connect via a proxy of the given
type.

    --proxy_host=<IP or DNS address>: IP or DNS address of the
proxy to use.

    --proxy_port=<port #>: Proxy port to connect to.

    --verbose: Enable verbose output.

* If the env var REPO_OS_OVERRIDE is set to "windows",
  "macosx", or "linux", packages will be downloaded for that
OS.
```

4.1. 接受 License

```
$ yes|tools/bin/sdkmanager --licenses
```

4.2. 查看 SDK 列表

```
neo@ubuntu:~/tmp/tools$ bin/sdkmanager --list | grep
"platforms;"
Warning: File /home/neo/.android/repositories.cfg could not be
loaded.
  platforms;android-26 | 2       | Android SDK Platform 26  |
platforms/android-26/
  platforms;android-10
| 2             | Android SDK Platform 10
  platforms;android-11
| 2             | Android SDK Platform 11
  platforms;android-12
| 3             | Android SDK Platform 12
  platforms;android-13
| 1             | Android SDK Platform 13
  platforms;android-14
| 4             | Android SDK Platform 14
  platforms;android-15
| 5             | Android SDK Platform 15
  platforms;android-16
| 5             | Android SDK Platform 16
  platforms;android-17
| 3             | Android SDK Platform 17
  platforms;android-18
| 3             | Android SDK Platform 18
  platforms;android-19
| 4             | Android SDK Platform 19
  platforms;android-20
| 2             | Android SDK Platform 20
```

```
  platforms;android-21
| 2             | Android SDK Platform 21
  platforms;android-22
| 2             | Android SDK Platform 22
  platforms;android-23
| 3             | Android SDK Platform 23
  platforms;android-24
| 2             | Android SDK Platform 24
  platforms;android-25
| 3             | Android SDK Platform 25
  platforms;android-26
| 2             | Android SDK Platform 26
  platforms;android-27
| 3             | Android SDK Platform 27
  platforms;android-28
| 6             | Android SDK Platform 28
  platforms;android-7
| 3             | Android SDK Platform 7
  platforms;android-8
| 3             | Android SDK Platform 8
  platforms;android-9
| 2             | Android SDK Platform 9
```

4.3. 按照 Android SDK

最新版本 Android 9 SDK

```
bin/sdkmanager "platform-tools" "platforms;android-28"
```

按照旧版本的 Android 8 SDK

```
bin/sdkmanager "platforms;android-26"
```

5. 命令行操作

会同时生成debug和release两个包

```
./gradlew assemble
```

只生成release的包

```
./gradlew assembleRelease
```

6. adb 命令

默认情况执行 adb 会提示找不到命令

```
neo@MacBook-Pro-M2 ~ % adb
zsh: command not found: adb
```

提示

这里我使用的是 zsh shell

```
neo@MacBook-Pro-M2 ~ % open -e .zprofile

export PATH=${PATH}:~/Library/Android/sdk/platform-tools
```

现在可以正常使用了

```
neo@MacBook-Pro-M2 ~ % adb version
Android Debug Bridge version 1.0.41
Version 34.0.4-10411341
Installed as /Users/neo/Library/Android/sdk/platform-tools/adb
Running on Darwin 23.0.0 (arm64)
```

6.1. 设备管理

```
neo@MacBook-Pro-M2 ~ % adb devices
List of devices attached
```

```
0123456789ABCDEF        device
CFE6R21625003544        device
```

查看详细信息

```
neo@MacBook-Pro-M2 ~ % adb devices -l
List of devices attached
0123456789ABCDEF       device usb:1310720X
product:full_aiv8167sm3_bsp model:aiv8167sm3_bsp
device:aiv8167sm3_bsp transport_id:2
CFE6R21625003544       device usb:1114112X product:MRR-W29
model:MRR_W29 device:HWMRR-Q transport_id:1
```

6.2. Shell

```
neo@MacBook-Pro-M2 ~ % adb -s CFE6R21625003544 shell
HWMRR-Q:/ $
```

网络相关

查看 IP 地址

```
neo@MacBook-Pro-M2 ~> adb shell ifconfig
wlan0     Link encap:Ethernet  HWaddr c0:84:7d:2b:3c:24
          UP BROADCAST MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:0 TX bytes:0
```

```
lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope: Host
          UP LOOPBACK RUNNING  MTU:65536  Metric:1
          RX packets:340 errors:0 dropped:0 overruns:0 frame:0
          TX packets:340 errors:0 dropped:0 overruns:0
carrier:0
          collisions:0 txqueuelen:1
          RX bytes:27313 TX bytes:27313

eth0      Link encap:Ethernet  HWaddr 86:7a:05:cc:ae:72
          inet6 addr: fe80::847a:5ff:fecc:ae72/64 Scope: Link
          UP BROADCAST MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:6 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:0 TX bytes:508
          Interrupt:42
```

无线 IP 地址

只查看无线 IP 地址

```
adb shell ifconfig wlan0
```

Mac 地址

```
rk3568_r:/ $ cat /sys/class/net/wlan0/address
30:7b:c9:0f:12:b9
```

查看 MAC 地址

```
neo@MacBook-Pro-M2 ~> adb shell cat
/sys/class/net/wlan0/address
c0:84:7d:2b:3c:24
```

内存信息

```
neo@MacBook-Pro-M2 ~> adb shell cat /proc/meminfo
MemTotal:        2043916 kB
MemFree:          844392 kB
MemAvailable:    1334032 kB
Buffers:            6376 kB
Cached:           609984 kB
SwapCached:            0 kB
Active:           562872 kB
Inactive:         346688 kB
Active(anon):     297196 kB
Inactive(anon):   116108 kB
Active(file):     265676 kB
Inactive(file):   230580 kB
Unevictable:         256 kB
Mlocked:             256 kB
HighTotal:       1564672 kB
HighFree:         661800 kB
LowTotal:         479244 kB
LowFree:          182592 kB
SwapTotal:        520908 kB
SwapFree:         520908 kB
Dirty:                 0 kB
Writeback:             0 kB
AnonPages:        293468 kB
Mapped:           348972 kB
Shmem:            120132 kB
Slab:             194608 kB
SReclaimable:     172360 kB
SUnreclaim:        22248 kB
KernelStack:        5376 kB
PageTables:        12448 kB
```

```
NFS_Unstable:          0 kB
Bounce:                0 kB
WritebackTmp:          0 kB
CommitLimit:     1542864 kB
Committed_AS:   25076844 kB
VmallocTotal:     499712 kB
VmallocUsed:           0 kB
VmallocChunk:          0 kB
CmaTotal:          16384 kB
CmaFree:           14540 kB
```

查看硬件与系统属性

```
neo@MacBook-Pro-M2 ~> adb shell cat /system/build.prop

# begin build properties
# autogenerated by buildinfo.sh
ro.build.id=NHG47K
ro.build.display.id=rk3288-userdebug 7.1.2 NHG47K
eng.server22.20230423.034518 test-keys
ro.build.version.incremental=eng.server22.20230423.034518
ro.build.version.sdk=25
ro.build.version.preview_sdk=0
ro.build.version.codename=REL
ro.build.version.all_codenames=REL
ro.build.version.release=7.1.2
ro.build.version.security_patch=2017-04-05
ro.build.version.base_os=
ro.build.date=Sun Apr 23 03:45:18 UTC 2023
ro.build.date.utc=1682221518
ro.build.type=userdebug
ro.build.user=server22
ro.build.host=server-zysj-03
ro.build.tags=test-keys
ro.build.flavor=rk3288-userdebug
ro.product.model=rk3288
ro.product.brand=Android
ro.product.name=rk3288
```

```
ro.product.device=rk3288
ro.product.board=rk30sdk
# ro.product.cpu.abi and ro.product.cpu.abi2 are obsolete,
# use ro.product.cpu.abilist instead.
ro.product.cpu.abi=armeabi-v7a
ro.product.cpu.abi2=armeabi
ro.product.cpu.abilist=armeabi-v7a,armeabi
ro.product.cpu.abilist32=armeabi-v7a,armeabi
ro.product.cpu.abilist64=
ro.product.manufacturer=rockchip
ro.product.locale.language=zh
ro.product.locale.region=CN
persist.sys.timezone=Asia/Shanghai
ro.wifi.channels=
ro.board.platform=rk3288
# ro.build.product is obsolete; use ro.product.device
ro.build.product=rk3288
# Do not try to parse description, fingerprint, or thumbprint
ro.build.description=rk3288-userdebug 7.1.2 NHG47K
eng.server22.20230423.034518 test-keys
ro.build.fingerprint=Android/rk3288/rk3288:7.1.2/NHG47K/server0
4230345:userdebug/test-keys
ro.build.characteristics=tablet
# end build properties
#
# from device/rockchip/rk3288/system.prop
#
#
# system.prop
#

# modify by alvin, support for 4G patch.
rild.libpath=/system/lib/libreference-ril.so
rild.libargs=-d /dev/ttyUSB3
# Default ecclist
ro.ril.ecclist=112,911
ro.opengles.version=196610
wifi.interface=wlan0
# modify by alvin, support for 4G patch.
#rild.libpath=/system/lib/libril-rk29-dataonly.so
#rild.libargs=-d /dev/ttyACM0
persist.tegra.nvmmlite = 1
ro.audio.monitorOrientation=true

#NFC
```

```
debug.nfc.fw_download=false
debug.nfc.se=false

#add Rockchip properties here
ro.rk.screenoff_time=2147483647
ro.rk.screenshot_enable=true
ro.rk.def_brightness=200
ro.rk.homepage_base=http://m.baidu.com/?from=844&vit=fps
ro.rk.install_non_market_apps=false
sys.hwc.compose_policy=0
sys.wallpaper.rgb565=0
sf.power.control=2073600
sys.rkadb.root=0
ro.sf.fakerotation=false
ro.sf.hwrotation=0
ro.rk.MassStorage=false
ro.rk.systembar.voiceicon=true
ro.rk.systembar.tabletUI=false
ro.rk.LowBatteryBrightness=false
ro.tether.denied=false
#repair by alvin, surport change system density value.
sys.resolution.changed=true
ro.default.size=100
#persist.sys.timezone=
ro.product.usbfactory=rockchip_usb
wifi.supplicant_scan_interval=15
ro.factory.tool=0
ro.kernel.android.checkjni=0
#set default lcd density to Rockchip tablet
ro.sf.lcd_density=240
ro.adb.secure=0
ro.rk.displayd.enable=false

#/*add by yfc for show vendor id*/
ro.source.code.version = 220

#add by alvin for hdmi rotation
ro.same.orientation=true
ro.orientation.einit=0
ro.rotation.external=true

#add by alvin, support for config camera rotation.
ro.camera.param.degree=0
ro.camera.back=0
ro.camera.place=0
```

```
# add by alvin for system 4k ui
# default main framebuffer resolution
persist.sys.framebuffer.main=1536x2048
sys.hwc.device.primary=HDMI-A

#
# ADDITIONAL_BUILD_PROPERTIES
#
ro.target.product=tablet
dalvik.vm.heapstartsize=16m
dalvik.vm.heapgrowthlimit=480m
dalvik.vm.heapsize=520m
dalvik.vm.heaptargetutilization=0.75
dalvik.vm.heapminfree=512k
dalvik.vm.heapmaxfree=8m
ro.config.ringtone=Ring_Synth_04.ogg
ro.config.notification_sound=pixiedust.ogg
ro.carrier=unknown
ro.config.alarm_alert=Alarm_Classic.ogg
ro.rksdk.version=RK30_ANDROID7.1.2-SDK-v1.00.00
camera2.portability.force_api=1
persist.sys.strictmode.visual=false
ro.rk.bt_enable=true
ro.rk.flash_enable=true
ro.rk.hdmi_enable=true
ro.factory.hasUMS=false
persist.sys.usb.config=mtp,adb
testing.mediascanner.skiplist=/mnt/shell/emulated/Android/
ro.factory.hasGPS=false
ro.factory.storage_suppntfs=true
ro.factory.without_battery=false
ro.rk.screenoff_time=2147483647
ro.com.widevine.cachesize=16777216
ro.enable.optee=true
ro.product.first_api_level=23
ro.boot.noril=true
keyguard.no_require_sim=true
ro.com.android.dataroaming=true
ril.function.dataonly=1
ro.config.enable.remotecontrol=false
ro.udisk.visible=true
ro.safemode.disabled=true
ro.wallpaper.fixsize=true
ro.hwui.texture_cache_size=72
ro.hwui.layer_cache_size=48
```

```
ro.hwui.r_buffer_cache_size=8
ro.hwui.path_cache_size=32
ro.hwui.gradient_cache_size=1
ro.hwui.drop_shadow_cache_size=6
ro.hwui.texture_cache_flushrate=0.4
ro.hwui.text_small_cache_width=1024
ro.hwui.text_small_cache_height=1024
ro.hwui.text_large_cache_width=2048
ro.hwui.text_large_cache_height=1024
ro.hwui.disable_scissor_opt=true
ro.rk.screenshot_enable=true
sys.status.hidebar_enable=false
persist.sys.ui.hw=true
ro.product.version=1.0.0
ro.product.ota.host=www.rockchip.com:2300
ro.sys.sdcardfs=true
persist.sys.dalvik.vm.lib.2=libart.so
dalvik.vm.isa.arm.variant=cortex-a15
dalvik.vm.isa.arm.features=default
dalvik.vm.lockprof.threshold=500
net.bt.name=Android
dalvik.vm.stack-trace-file=/data/anr/traces.txt
ro.expect.recovery_id=0x182fbd9a6eea8693a3aeac4bfab86ba6271f55d
100000000000000000000000
```

获取指定的属性 adb shell getprop net.bt.name

```
neo@MacBook-Pro-M2 ~> adb shell getprop net.bt.name
Android
```

6.3. 设备 ID

获取变量

```
```

```
neo@MacBook-Pro-M2 ~> adb shell settings get secure android_id
95c27630f4559e58
```

设置变量

```
RK3566:/ # settings put global policy_control
immersive.full=*
RK3566:/ # settings put global policy_control
immersive.status=*
RK3566:/ # settings put global policy_control
immersive.navigation=*
```

显示/关闭虚拟键

设置属性值为0表示一直打开虚拟按键，属性值为1 表示隐藏虚拟按键

```
adb root
adb remount
adb shell
$ getprop qemu.hw.mainkeys
$ setprop qemu.hw.mainkeys 0
$ stop
$ start
```

```
setprop persist.qemu.hw.mainkeys 0
```

6.4. 查看安卓版本

```
neo@MacBook-Pro-M2 ~> adb shell getprop
ro.build.version.release
7.1.2
```

产品型号

```
neo@MacBook-Pro-M2 ~> adb shell getprop ro.product.model
rk3288
```

6.5. Logcat

```
neo@MacBook-Pro-M2 ~> adb logcat
neo@MacBook-Pro-M2 ~> adb -s CFE6R21625003544 logcat
```

6.6. 上传文件

```
adb push libtinyalsa.so /system/lib/
```

```
neo@MacBook-Pro-M2 ~> adb push netkiller.wav /sdcard/
```

6.7. 下载文件

```
neo@MacBook-Pro-M2 tmp % adb pull /sdcard/file.wav
/sdcard/file.wav: 1 file pulled, 0 skipped. 0.0 MB/s (44 bytes
in 0.002s)
```

6.8. 安卓 .apk bk

```
        -l : 锁定应用程序
        -t : 允许测试包
        -d : 允许降级覆盖安装
        -p : 部分应用安装
        -g : 为应用程序授予所有运行时的权限
```

```
neo@MacBook-Pro-M2 ~> adb install netkiller.apk
```

6.9. 屏幕尺寸

```
neo@MacBook-Pro-M2 ~> adb shell wm size
Physical size: 1536x2048

neo@MacBook-Pro-M2 ~> adb shell wm density
Physical density: 240
```

```
neo@MacBook-Pro-M2 ~> adb shell dumpsys window displays
WINDOW MANAGER DISPLAY CONTENTS (dumpsys window displays)
  Display: mDisplayId=0
    init=1536x2048 240dpi cur=1536x2048 app=1536x1964
rng=1536x1416-2048x1928
    deferred=false layoutNeeded=false

  Application tokens in top down Z order:
    mStackId=1
    mDeferDetach=false
    mFullscreen=true
    mBounds=[0,0][1536,2048]
      taskId=56
        mFullscreen=true
        mBounds=[0,0][1536,2048]
        mdr=false
        appTokens=[AppWindowToken{1c392c token=Token{ff4ab7e
ActivityRecord{379f939 u0
com.wc.holoos/.player.PlayClockActivity t56}}}]
        mTempInsetBounds=[0,0][0,0]
          Activity #0 AppWindowToken{1c392c token=Token{ff4ab7e
ActivityRecord{379f939 u0
com.wc.holoos/.player.PlayClockActivity t56}}}
          windows=[Window{73ec1eb u0
com.wc.holoos/com.wc.holoos.player.PlayClockActivity}]
          windowType=2 hidden=false hasVisible=true
          app=true voiceInteraction=false
          allAppWindows=[Window{73ec1eb u0
com.wc.holoos/com.wc.holoos.player.PlayClockActivity}]
          task={taskId=56 appTokens=[AppWindowToken{1c392c
token=Token{ff4ab7e ActivityRecord{379f939 u0
com.wc.holoos/.player.PlayClockActivity t56}}}] mdr=false}
           appFullscreen=true requestedOrientation=1
          hiddenRequested=false clientHidden=false
reportedDrawn=true reportedVisible=true
          numInterestingWindows=1 numDrawnWindows=1
inPendingTransaction=false allDrawn=true (animator=true)
          startingData=null removed=false firstWindowDrawn=true
mIsExiting=false
```

```
    mStackId=0
    mDeferDetach=false
    mFullscreen=true
    mBounds=[0,0][1536,2048]
      taskId=55
        mFullscreen=true
        mBounds=[0,0][1536,2048]
        mdr=false
        appTokens=[AppWindowToken{56cdf68 token=Token{362805a
ActivityRecord{b30e805 u0 com.wc.holoos/.MainActivity t55}}}]
        mTempInsetBounds=[0,0][0,0]
          Activity #0 AppWindowToken{56cdf68
token=Token{362805a ActivityRecord{b30e805 u0
com.wc.holoos/.MainActivity t55}}}
          windows=[Window{90133ba u0
com.wc.holoos/com.wc.holoos.MainActivity}]
          windowType=2 hidden=true hasVisible=true
          app=true voiceInteraction=false
          allAppWindows=[Window{90133ba u0
com.wc.holoos/com.wc.holoos.MainActivity}]
          task={taskId=55 appTokens=[AppWindowToken{56cdf68
token=Token{362805a ActivityRecord{b30e805 u0
com.wc.holoos/.MainActivity t55}}}] mdr=false}
           appFullscreen=true requestedOrientation=1
          hiddenRequested=true clientHidden=true
reportedDrawn=false reportedVisible=false
          mAppStopped=true
          numInterestingWindows=1 numDrawnWindows=1
inPendingTransaction=false allDrawn=true (animator=true)
          startingData=null removed=false firstWindowDrawn=true
mIsExiting=false


    DimLayerController
      Task=55
        dimLayer=shared, animator=null, continueDimming=false
        mDimSurface=Surface(name=DimLayerController/Stack=0)
mLayer=110999 mAlpha=0.0
        mLastBounds=[-384,-512][1920,2560] mBounds=[-384,-512]
[1920,2560]
        Last animation:  mDuration=200 mStartTime=7877723
curTime=9020147
         mStartAlpha=0.6 mTargetAlpha=0.0
      Task=56
        dimLayer=shared, animator=null, continueDimming=false
```

```
        mDimSurface=Surface(name=DimLayerController/Stack=0)
mLayer=110999 mAlpha=0.0
        mLastBounds=[-384,-512][1920,2560] mBounds=[-384,-512]
[1920,2560]
        Last animation:  mDuration=200 mStartTime=7877723
curTime=9020147
         mStartAlpha=0.6 mTargetAlpha=0.0
      Stack=1
        dimLayer=shared, animator=null, continueDimming=false
        mDimSurface=Surface(name=DimLayerController/Stack=0)
mLayer=110999 mAlpha=0.0
        mLastBounds=[-384,-512][1920,2560] mBounds=[-384,-512]
[1920,2560]
        Last animation:  mDuration=200 mStartTime=7877723
curTime=9020147
         mStartAlpha=0.6 mTargetAlpha=0.0
      Stack=0
        dimLayer=shared, animator=null, continueDimming=false
        mDimSurface=Surface(name=DimLayerController/Stack=0)
mLayer=110999 mAlpha=0.0
        mLastBounds=[-384,-512][1920,2560] mBounds=[-384,-512]
[1920,2560]
        Last animation:  mDuration=200 mStartTime=7877723
curTime=9020147
         mStartAlpha=0.6 mTargetAlpha=0.0

    DockedStackDividerController
      mLastVisibility=false
      mMinimizedDock=false
      mAdjustedForIme=false
      mAdjustedForDivider=false
```

6.10. dump 系统信息

```
neo@MacBook-Pro-M2 ~> adb shell dumpsys
```

电池信息

```
neo@MacBook-Pro-M2 ~> adb shell dumpsys battery
Current Battery Service state:
  AC powered: true
  USB powered: false
  Wireless powered: false
  Max charging current: 0
  Max charging voltage: 0
  Charge counter: 0
  status: 2
  health: 2
  present: true
  level: 100
  scale: 100
  voltage: 0
  temperature: 424
  technology:
```

6.11. 解锁

```
neo@MacBook-Pro-M2 ~ % adb root
neo@MacBook-Pro-M2 ~ % adb reboot bootloader
neo@MacBook-Pro-M2 ~ % fastboot flashing unlock
neo@MacBook-Pro-M2 ~ % fastboot getvar unlocked
neo@MacBook-Pro-M2 ~ % adb disable-verity
neo@MacBook-Pro-M2 ~ % adb reboot
neo@MacBook-Pro-M2 ~ % adb root
neo@MacBook-Pro-M2 ~ % adb remount
```

第 2 章 AndroidManifest.xml

1. SDK 版本配置

```
<uses-sdk
        android:minSdkVersion="26"
        android:targetSdkVersion="28" />
```

2. 开启网络

```
<uses-permission android:name="android.permission.INTERNET" />
```

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

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

                <category
android:name="android.intent.category.LAUNCHER" />

            </intent-filter>
        </activity>

    </application>
    <uses-permission android:name="android.permission.INTERNET"
/>
</manifest>
```

3. 文件存储权限

```
        <uses-permission
android:name="android.permission.MOUNT_UNMOUNT_FILESYSTEMS"
/>
    <uses-permission
android:name="android.permission.WRITE_EXTERNAL_STORAGE" />
```

sdcard

```
    <uses-permission
android:name="android.permission.READ_EXTERNAL_STORAGE" />
    <uses-permission
android:name="android.permission.WRITE_EXTERNAL_STORAGE" />
    <uses-permission
android:name="android.permission.MANAGE_EXTERNAL_STORAGE" />
```

4. 相机权限

```
        <uses-permission
android:name="android.permission.CAMERA" />
    <uses-feature android:name="android.hardware.camera" />
```

5. GPS 定位权限

```
<uses-permission
android:name="android.permission.ACCESS_COARSE_LOCATION" />
<uses-permission
android:name="android.permission.ACCESS_FINE_LOCATION" />
<uses-permission
android:name="android.permission.ACCESS_WIFI_STATE" />
<uses-permission
android:name="android.permission.ACCESS_NETWORK_STATE" />
<uses-permission
android:name="android.permission.CHANGE_WIFI_STATE" />
<uses-permission android:name="android.permission.INTERNET"
/>
```

6. 全屏-无标题

首先定义主题

```
<resources xmlns:tools="http://schemas.android.com/tools">
    <style name="Theme.Main.Fullscreen"
parent="Theme.AppCompat.Light">
        <item name="windowActionBar">false</item>//无ActionBar
        <item name="windowNoTitle">true</item> //无标题
        <item name="android:windowFullscreen">true</item> //全
屏
        <item name="android:navigationBarColor"
tools:targetApi="lollipop">@android:color/transparent</item>
        <item
name="android:windowBackground">@drawable/fo2</item>//背景图
        <item name="android:windowDrawsSystemBarBackgrounds"
tools:targetApi="lollipop">true</item>
    </style>
</resources>
```

Layout

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/title"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    tools:context="cn.aigcsst.demo.MainActivity">

    <ImageView
        android:id="@+id/imageView"
```

```
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:scaleType="matrix"
        app:srcCompat="@drawable/fo2"
        tools:ignore="MissingConstraints" />
</androidx.constraintlayout.widget.ConstraintLayout>
```

AndroidManifest.xml 切换主题

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

    <uses-permission
android:name="android.permission.INTERNET" />
    <uses-permission
android:name="android.permission.ACCESS_NETWORK_STATE" />
    <uses-permission
android:name="android.permission.ACCESS_WIFI_STATE" />
    <uses-permission
android:name="android.permission.CHANGE_NETWORK_STATE" />
    <uses-permission
android:name="android.permission.READ_PHONE_STATE" />
    <uses-permission
android:name="android.permission.ACCESS_FINE_LOCATION" />
    <uses-permission
android:name="android.permission.READ_EXTERNAL_STORAGE" />
    <uses-permission
android:name="android.permission.WRITE_EXTERNAL_STORAGE" />
    <uses-permission
android:name="android.permission.RECORD_AUDIO" />

    <application
        android:name="cn.aigcsst.demo.ContextHolder"
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="Demo"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
```

```
        android:theme="@style/Theme.Main.Fullscreen"
        android:usesCleartextTraffic="true">
        <activity
            android:name="cn.aigcsst.demo.MainActivity"
            android:exported="true">
            <intent-filter>
                <action
android:name="android.intent.action.MAIN" />
                <category
android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>


    </application>

</manifest>
```

7. 设置为默认开机启动

```
<category android:name="android.intent.category.HOME" />
<category android:name="android.intent.category.DEFAULT" />
```

```
        <activity
            android:name="cn.aigcsst.album.MainActivity"
            android:exported="true">
            <intent-filter>
                <action
android:name="android.intent.action.MAIN" />

                <category
android:name="android.intent.category.LAUNCHER" />
                <category
android:name="android.intent.category.HOME" />
                <category
android:name="android.intent.category.DEFAULT" />
            </intent-filter>
        </activity>
```

第 3 章 设备

1. 环境变量

1.1. 扩展存储

状态

```
Environment.getExternalStorageState().equals(Environment.MEDI
A_MOUNTED);
```

绝对路径

```
Environment.getExternalStorageDirectory().getAbsolutePath();
```

```
Log.i("Environment", "getExternalStorageDirectory(): " +
Environment.getExternalStorageDirectory().toString());

Log.i("Environment","getExternalStoragePublicDirectory(Enviro
nment.DIRECTORY_PICTURES): " +
Environment.getExternalStoragePublicDirectory(Environment.DIR
ECTORY_PICTURES).toString());
```

1.2. 下载缓存目录

```
Log.i("Environment", "getDownloadCacheDirectory(): "+
Environment.getDownloadCacheDirectory().toString());
```

1.3. 数据目录

```
Log.i("Environment", "getRootDirectory(): " +
Environment.getRootDirectory().toString());
Log.i("Environment", "getDataDirectory(): " +
Environment.getDataDirectory().toString());
```

2. 配置文件

2.1. *.properties 文件

创建 properties 文件 res/config/development.properties

```
api_url=https://api.netkiller.cn/v1/
api_key=123456
```

```
package cn.netkiller.app;

import android.content.Context;
import android.content.res.Resources;
import android.util.Log;

import java.io.IOException;
import java.io.InputStream;
import java.util.Properties;

public final class Config {
    private static final String TAG = "Config";

    public static String getKey(Context context, String name)
{
        Resources resources = context.getResources();

        try {
            InputStream rawResource =
resources.openRawResource(R.config.development);
            Properties properties = new Properties();
            properties.load(rawResource);
            return properties.getProperty(name);
        } catch (Resources.NotFoundException e) {
            Log.e(TAG, "Unable to find the config file: " +
e.getMessage());
```

```
        } catch (IOException e) {
            Log.e(TAG, "Failed to open config file.");
        }

        return null;
    }
}
```

```
String apiUrl = Config.getKey(this, "api_url");
String apiKey = Config.getKey(this, "api_key");
```

2.2. 再 AndroidManifest.xml 使用 meta-data element 定义

```
...
<application ...>
    ...
        ...
    <meta-data android:name="api_url"
android:value="https://api.netkiller.cn/v1/"/>
    <meta-data android:name="api_key"
android:value="123456"/>
</application>
```

```
public static String getMetaData(Context context, String
name) {
    try {
        ApplicationInfo ai =
context.getPackageManager().getApplicationInfo(context.getPac
kageName(), PackageManager.GET_META_DATA);
        Bundle bundle = ai.metaData;
```

```
        return bundle.getString(name);
    } catch (PackageManager.NameNotFoundException e) {
        Log.e(TAG, "Unable to load meta-data: " +
e.getMessage());
    }
    return null;
}
```

```
String apiUrl = getMetaData(this, "api_url");
String apiKey = getMetaData(this, "api_key");
```

2.3. 再 build.gradle 文件中配置 productFlavors

```
productFlavors {
    prod {
        buildConfigField 'String', 'API_URL',
'"https://api.netkiller.cn/v1/"'
        buildConfigField 'String', 'API_KEY', '"123456"'
    }
}
```

引用 config 方法

```
String apiUrl = BuildConfig.API_URL;
String apiKey = BuildConfig.API_KEY;
```

2.4. 从 assets 目录读取配置文件

```
import java.io.InputStream;
import java.util.Properties;

import android.content.Context;

public class Config {
        public static Properties getProperties(Context c){
                Properties properties = new Properties();
                try {
                        //方法一：通过activity中的context攻取
setting.properties的FileInputStream
                        InputStream in =
c.getAssets().open("appConfig.properties");
                //方法二：通过class获取setting.properties的
FileInputStream
                //InputStream in =
PropertiesUtill.class.getResourceAsStream("/assets/setting.pr
operties "));
                properties.load(in);
                } catch (Exception e1) {
                        e1.printStackTrace();
                }
                return props;
        }
}
```

```
Properties properties =
Config.getProperties(context.getApplicationContext());
serverUrl = properties.getProperty("serverUrl");
Log.i("URL", serverUrl);
```

配置文件例子

```
package cn.netkiller.album.config;

import cn.netkiller.album.MainApp;

import java.io.IOException;
import java.io.InputStream;
import java.io.InputStreamReader;
import java.nio.charset.StandardCharsets;
import java.util.Properties;

public class Config {
    private static final String TAG = "Config";
    private Properties properties = null;
    public Config() {
        try {
            properties = new Properties();
            InputStream inputStream =
MainApp.getContext().getAssets().open("config.properties");
            InputStreamReader isr = new
InputStreamReader(inputStream, StandardCharsets.UTF_8);
            properties.load(isr);
            isr.close();
        } catch (IOException e) {
            e.printStackTrace();
        }
    }


    public String getServerURI() {
        return
properties.getProperty("mqtt.serveruri").trim();
    }

    public String getUsername() {
        return
properties.getProperty("mqtt.username").trim();
    }

    public String getPassword() {
        return
properties.getProperty("mqtt.password").trim();
    }
```

```
    public String getProjectName() {
        return properties.getProperty("project.name").trim();
    }

    public String getProjectScenes() {
        return
properties.getProperty("project.scenes").trim();
    }

    public String getProjectGroup() {
        return
properties.getProperty("project.group").trim();
    }

    public String getTopicBroadcast() {

        return
properties.getProperty("mqtt.topic.broadcast").trim();
    }

    public String getTopicRegister() {
        return
properties.getProperty("mqtt.topic.register").trim();
    }
    public String getTopicNotification(){
        return
properties.getProperty("mqtt.topic.notification").trim();
    }
}
```

3. 设备信息

设备 ID

```
String androidId =
Settings.System.getString(getContentResolver(),
"android_id");
```

4. 声卡

```
rk3288:/ $ cd /dev/snd/
rk3288:/dev/snd $ ls
controlC0 pcmC0D0c pcmC0D0p pcmC0D1p timer
```

4.1. 播放

播放测试

```
aiv8167sm3_bsp:/storage/emulated/0 # tinyplay zai.wav
Playing sample: 1 ch, 16000 hz, 16 bit
```

```
tinyplay netkiller.wav -D 0 -d 0 -r 48000 -c 2

aiv8167sm3_bsp:/storage/emulated/0 # tinyplay zai.wav -D 0 -d 0
-r 48000
Playing sample: 1 ch, 16000 hz, 16 bit
```

```
neo@MacBook-Pro-M2 tmp % adb shell tinyplay /sdcard/zai.wav
Playing sample: 1 ch, 16000 hz, 16 bit
```

4.2. 录音

录音测试

```
tinycap netkiller.wav -D 0 -d 0 -c 1 -r 48000
```

```
-D  card        声卡

-d  device     设备

-c  channels  通道

-r  rate   采样率

-b  bits   pcm 位宽

-p  period_size   一次中断的帧数

-n  n_periods     周期数

例子:  tinycap /sdcard/test.pcm -D 0 -d 0 -c 4 -r 48000 -b 32 -p
768 -n 10

声卡0；设备0；四通道；48K采样率；32位位宽；一帧数据存储大小；采样n次
```

查看录音设备

```
neo@MacBook-Pro-M2 tmp % adb shell ls "/dev/snd/pcmC*c"
/dev/snd/pcmC0D10c
/dev/snd/pcmC0D1c
/dev/snd/pcmC0D2c
/dev/snd/pcmC0D4c
/dev/snd/pcmC0D6c
/dev/snd/pcmC0D8c
/dev/snd/pcmC0D9c
```

默认录音参数 Capturing sample: 2 ch, 44100 hz, 16 bit

```
neo@MacBook-Pro-M2 tmp % adb shell tinycap /sdcard/file.pcm -D
0 -d 2 -T 5
Capturing sample: 2 ch, 44100 hz, 16 bit
Captured 0 frames
```

指定参数

```
neo@MacBook-Pro-M2 tmp % adb pull /sdcard/file.pcm
/sdcard/file.wav: 1 file pulled, 0 skipped. 14.6 MB/s (1851436
bytes in 0.121s)
neo@MacBook-Pro-M2 tmp % adb shell tinycap /sdcard/file.wav -D
0 -d 2 -c 2 -r 48000 -b 16 -T 5
Capturing sample: 2 ch, 48000 hz, 16 bit
Captured 0 frames
```

下载录音文件

```
neo@MacBook-Pro-M2 tmp % adb pull /sdcard/file.pcm
/sdcard/file.wav: 1 file pulled, 0 skipped. 7.0 MB/s (966700
bytes in 0.132s)
```

4.3. 查看声卡信息

device 0 表示录音设备

```
aiv8167sm3_bsp:/storage/emulated/0 # tinypcminfo -D 0 -d 0
Info for card 0, device 0:

PCM out:
      Access:   0x000009
   Format[0]:   0x000444
   Format[1]:   00000000
 Format Name:   S16_LE, S24_LE, S32_LE
   Subformat:   0x000001
        Rate:   min=8000Hz      max=192000Hz
    Channels:   min=1           max=2
 Sample bits:   min=16          max=32
 Period size:   min=32          max=32768
Period count:   min=2           max=256

PCM in:
cannot open device '/dev/snd/pcmC0D0c'
Device does not exist.
```

device 1 表示录音设备

```
aiv8167sm3_bsp:/storage/emulated/0 # tinypcminfo -D 0 -d 1
Info for card 0, device 1:

PCM out:
cannot open device '/dev/snd/pcmC0D1p'
Device does not exist.

PCM in:
      Access:   0x000009
   Format[0]:   0x000444
   Format[1]:   00000000
 Format Name:   S16_LE, S24_LE, S32_LE
   Subformat:   0x000001
        Rate:   min=8000Hz      max=192000Hz
    Channels:   min=1           max=2
 Sample bits:   min=16          max=32
```

```
 Period size:   min=32          max=8192
Period count:   min=2           max=256
```

如果不知道设备编号，可以使用 /proc/asound/cards 替代

```
aiv8167sm3_bsp:/storage/emulated/0 # tinypcminfo -D
/proc/asound/cards
Info for card 0, device 0:

PCM out:
      Access:   0x000009
   Format[0]:   0x000444
   Format[1]:   00000000
 Format Name:   S16_LE, S24_LE, S32_LE
   Subformat:   0x000001
        Rate:   min=8000Hz      max=192000Hz
    Channels:   min=1           max=2
 Sample bits:   min=16          max=32
 Period size:   min=32          max=32768
Period count:   min=2           max=256

PCM in:
cannot open device '/dev/snd/pcmC0D0c'
Device does not exist.
```

4.4. /proc/asound 设备信息

```
aiv8167sm3_bsp:/storage/emulated/0 # ls -l /proc/asound
card0
cards
devices
hwdep
mtsndcard
oss
```

```
pcm
seq
timers
version
```

查看当前的声卡

```
aiv8167sm3_bsp:/storage/emulated/0 # cat /proc/asound/cards
 0 [mtsndcard      ]: mt-snd-card - mt-snd-card
                      mt-snd-card
```

```
aiv8167sm3_bsp:/storage/emulated/0 # cat /proc/asound/pcm
00-00: MultiMedia1_PLayback (*) :  : playback 1
00-01: MultiMedia_Capture (*) :  : capture 1
00-02: TDM_Capture (*) :  : capture 1
00-03: HMDI_PLayback (*) :  : playback 1
00-04: DL1_AWB_Record (*) :  : capture 1
00-05: MultiMedia2_PLayback (*) :  : playback 1
00-06: VOIP_Call_BT_Capture (*) :  : capture 1
00-07: MRGRX_PLayback (*) :  : playback 1
00-08: MRGRX_CAPTURE (*) :  : capture 1
00-09: BTCVSD_Capture snd-soc-dummy-dai-9 :  : playback 1 :
capture 1
00-10: BTCVSD_Playback snd-soc-dummy-dai-10 :  : playback 1 :
capture 1
```

```
aiv8167sm3_bsp:/storage/emulated/0 # cat /proc/asound/pcm
00-00: MultiMedia1_PLayback (*) :  : playback 1
00-01: MultiMedia_Capture (*) :  : capture 1
00-02: TDM_Capture (*) :  : capture 1
00-03: HMDI_PLayback (*) :  : playback 1
00-04: DL1_AWB_Record (*) :  : capture 1
```

```
00-05: MultiMedia2_PLayback (*) :  : playback 1
00-06: VOIP_Call_BT_Capture (*) :  : capture 1
00-07: MRGRX_PLayback (*) :  : playback 1
00-08: MRGRX_CAPTURE (*) :  : capture 1
00-09: BTCVSD_Capture snd-soc-dummy-dai-9 :  : playback 1 :
capture 1
00-10: BTCVSD_Playback snd-soc-dummy-dai-10 :  : playback 1 :
capture 1
```

4.5. 查看声卡当前占用设备

```
aiv8167sm3_bsp:/storage/emulated/0 # lsof |grep pcm
audio@2.0   253 audioserve   16u      CHR              116,3
0t0       8644 /dev/snd/pcmC0D0p
omx@1.0-s   316 mediacodec  mem       REG             179,14
57968        603 /vendor/lib/libMtkOmxAdpcmDec.so
m.netkiller  2956     u0_a51  mem   unknown
/dev/ashmem//data/user/0/cn.netkiller.ui/files/msclib/1691060
275989tts.pcm (deleted)
m.netkiller  2956     u0_a51   54u      CHR
116,5       0t0       9451 /dev/snd/pcmC0D2c
```

4.6. tinymix 设置声卡参数

查看所有参数

```
rk3288:/storage/emulated/0 # tinymix
Mixer name: 'rockchip,rt5640-codec'
Number of controls: 125
ctl     type    num     name
value
0       BOOL    1       Mono Playback Switch
```

```
Off
1       INT     2       Mono DAC Playback Volume
175 175
2       BOOL    2       Speaker Channel Switch
Off Off
3       INT     2       Speaker Playback Volume
31 31
4       BOOL    2       HP Channel Switch
Off Off
5       INT     2       HP Playback Volume
31 31
6       BOOL    2       OUT Playback Switch
Off Off
7       BOOL    2       OUT Channel Switch
Off Off
8       INT     2       OUT Playback Volume
31 31
9       BOOL    2       DAC2 Playback Switch
On On
10      INT     2       DAC1 Playback Volume
175 175
11      INT     1       IN1 Boost
0
12      INT     1       IN2 Boost
0
13      INT     1       IN3 Boost
0
14      INT     2       IN Capture Volume
23 23
15      BOOL    2       ADC Capture Switch
On On
16      INT     2       ADC Capture Volume
47 47
17      INT     2       Mono ADC Capture Volume
47 47
18      INT     2       ADC Boost Gain
0 0
19      ENUM    1       Class D SPK Ratio Control
2.77x
20      ENUM    1       ADC IF1 Data Switch
Normal
21      ENUM    1       DAC IF1 Data Switch
Normal
22      ENUM    1       ADC IF2 Data Switch
Normal
```

```
23      ENUM    1       DAC IF2 Data Switch
Normal
24      BYTE    128     ELD
00000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000
0000
25      INT     1       AUDIO MODE
0
26      BOOL    1       RECMIXL HPOL Switch
Off
27      BOOL    1       RECMIXL INL Switch
Off
28      BOOL    1       RECMIXL BST3 Switch
Off
29      BOOL    1       RECMIXL BST2 Switch
Off
30      BOOL    1       RECMIXL BST1 Switch
Off
31      BOOL    1       RECMIXL OUT MIXL Switch
Off
32      BOOL    1       RECMIXR HPOR Switch
Off
33      BOOL    1       RECMIXR INR Switch
Off
34      BOOL    1       RECMIXR BST3 Switch
Off
35      BOOL    1       RECMIXR BST2 Switch
Off
36      BOOL    1       RECMIXR BST1 Switch
Off
37      BOOL    1       RECMIXR OUT MIXR Switch
Off
38      ENUM    1       Stereo ADC2 Mux
DMIC1
39      ENUM    1       Stereo ADC1 Mux
ADC
40      ENUM    1       Mono ADC L2 Mux
DMIC L1
41      ENUM    1       Mono ADC L1 Mux
ADCL
42      ENUM    1       Mono ADC R1 Mux
ADCR
43      ENUM    1       Mono ADC R2 Mux
```

```
DMIC R1
44      BOOL    1       Stereo ADC MIXL ADC1 Switch
Off
45      BOOL    1       Stereo ADC MIXL ADC2 Switch
Off
46      BOOL    1       Stereo ADC MIXR ADC1 Switch
Off
47      BOOL    1       Stereo ADC MIXR ADC2 Switch
Off
48      BOOL    1       Mono ADC MIXL ADC1 Switch
Off
49      BOOL    1       Mono ADC MIXL ADC2 Switch
Off
50      BOOL    1       Mono ADC MIXR ADC1 Switch
Off
51      BOOL    1       Mono ADC MIXR ADC2 Switch
Off
52      ENUM    1       DAI select
1:2|2:1
53      ENUM    1       SDI select
IF1
54      BOOL    1       DAC MIXL Stereo ADC Switch
Off
55      BOOL    1       DAC MIXL INF1 Switch
On
56      BOOL    1       DAC MIXR Stereo ADC Switch
Off
57      BOOL    1       DAC MIXR INF1 Switch
On
58      BOOL    1       Mono DAC MIXL DAC L1 Switch
Off
59      BOOL    1       Mono DAC MIXL DAC L2 Switch
On
60      BOOL    1       Mono DAC MIXL DAC R2 Switch
Off
61      BOOL    1       Mono DAC MIXR DAC R1 Switch
Off
62      BOOL    1       Mono DAC MIXR DAC R2 Switch
On
63      BOOL    1       Mono DAC MIXR DAC L2 Switch
Off
64      BOOL    1       DIG MIXL DAC L1 Switch
Off
65      BOOL    1       DIG MIXL DAC L2 Switch
Off
```

```
66      BOOL    1       DIG MIXR DAC R1 Switch
Off
67      BOOL    1       DIG MIXR DAC R2 Switch
Off
68      BOOL    1       SPK MIXL REC MIXL Switch
Off
69      BOOL    1       SPK MIXL INL Switch
Off
70      BOOL    1       SPK MIXL DAC L1 Switch
Off
71      BOOL    1       SPK MIXL DAC L2 Switch
Off
72      BOOL    1       SPK MIXL OUT MIXL Switch
Off
73      BOOL    1       SPK MIXR REC MIXR Switch
Off
74      BOOL    1       SPK MIXR INR Switch
Off
75      BOOL    1       SPK MIXR DAC R1 Switch
Off
76      BOOL    1       SPK MIXR DAC R2 Switch
Off
77      BOOL    1       SPK MIXR OUT MIXR Switch
Off
78      BOOL    1       SPOL MIX DAC R1 Switch
Off
79      BOOL    1       SPOL MIX DAC L1 Switch
Off
80      BOOL    1       SPOL MIX SPKVOL R Switch
Off
81      BOOL    1       SPOL MIX SPKVOL L Switch
Off
82      BOOL    1       SPOL MIX BST1 Switch
Off
83      BOOL    1       SPOR MIX DAC R1 Switch
Off
84      BOOL    1       SPOR MIX SPKVOL R Switch
Off
85      BOOL    1       SPOR MIX BST1 Switch
Off
86      BOOL    1       LOUT MIX DAC L1 Switch
Off
87      BOOL    1       LOUT MIX DAC R1 Switch
Off
88      BOOL    1       LOUT MIX OUTVOL L Switch
```

```
Off
89      BOOL    1       LOUT MIX OUTVOL R Switch
Off
90      BOOL    1       Speaker L Playback Switch
Off
91      BOOL    1       Speaker R Playback Switch
Off
92      BOOL    1       HP L Playback Switch
On
93      BOOL    1       HP R Playback Switch
On
94      ENUM    1       DAC L2 Mux
IF2
95      ENUM    1       DAC R2 Mux
IF2
96      BOOL    1       Stereo DAC MIXL DAC L1 Switch
Off
97      BOOL    1       Stereo DAC MIXL DAC L2 Switch
Off
98      BOOL    1       Stereo DAC MIXL ANC Switch
Off
99      BOOL    1       Stereo DAC MIXR DAC R1 Switch
Off
100     BOOL    1       Stereo DAC MIXR DAC R2 Switch
Off
101     BOOL    1       Stereo DAC MIXR ANC Switch
Off
102     BOOL    1       OUT MIXL SPK MIXL Switch
Off
103     BOOL    1       OUT MIXL BST1 Switch
Off
104     BOOL    1       OUT MIXL INL Switch
Off
105     BOOL    1       OUT MIXL REC MIXL Switch
Off
106     BOOL    1       OUT MIXL DAC R2 Switch
Off
107     BOOL    1       OUT MIXL DAC L2 Switch
Off
108     BOOL    1       OUT MIXL DAC L1 Switch
Off
109     BOOL    1       OUT MIXR SPK MIXR Switch
Off
110     BOOL    1       OUT MIXR BST2 Switch
Off
```

```
111     BOOL    1       OUT MIXR BST1 Switch
Off
112     BOOL    1       OUT MIXR INR Switch
Off
113     BOOL    1       OUT MIXR REC MIXR Switch
Off
114     BOOL    1       OUT MIXR DAC L2 Switch
Off
115     BOOL    1       OUT MIXR DAC R2 Switch
Off
116     BOOL    1       OUT MIXR DAC R1 Switch
Off
117     BOOL    1       HPO MIX DAC2 Switch
On
118     BOOL    1       HPO MIX DAC1 Switch
Off
119     BOOL    1       HPO MIX HPVOL Switch
Off
120     BOOL    1       Mono MIX DAC R2 Switch
Off
121     BOOL    1       Mono MIX DAC L2 Switch
Off
122     BOOL    1       Mono MIX OUTVOL R Switch
Off
123     BOOL    1       Mono MIX OUTVOL L Switch
Off
124     BOOL    1       Mono MIX BST1 Switch
Off
```

查看指定参数

```
rk3288:/ $ tinymix 33
RECMIXR INR Switch: Off
```

设置参数

```
# 当前位 Off 状态
rk3288:/ $ tinymix 33
RECMIXR INR Switch: Off

# 修改位 On 状态
rk3288:/ $ tinymix 33 1

rk3288:/ $ tinymix 33
RECMIXR INR Switch: On

# 修改回 Off 状态
rk3288:/ $ tinymix 33 0

rk3288:/ $ tinymix 33
RECMIXR INR Switch: Off
```

4.7. 麦克风阵列调试

USB-Audio - Yundea M1051

```
rk3568_r:/ # cat /proc/asound/cards
 0 [rockchiphdmi   ]: rockchip_hdmi - rockchip,hdmi
                      rockchip,hdmi
 1 [rockchiprk809co]: rockchip_rk809- - rockchip,rk809-codec
                      rockchip,rk809-codec
 2 [M1051          ]: USB-Audio - Yundea M1051
                      Yundea Technology Yundea M1051 at usb-
xhci-hcd.5.auto-1, full speed
```

```
rk3568_r:/ # cat /proc/asound/card2/usbmixer
USB Mixer: usb_id=0x4c4a3135, ctrlif=1, ctlerr=0
Card: Yundea Technology Yundea M1051 at usb-xhci-hcd.5.auto-1,
full speed
```

```
  Unit: 5
    Control: name="Auto Gain Control", index=0
    Info: id=5, control=7, cmask=0x0, channels=1,
type="BOOLEAN"
    Volume: min=0, max=1, dBmin=0, dBmax=0
  Unit: 5
    Control: name="Mic Capture Volume", index=0
    Info: id=5, control=2, cmask=0x0, channels=1, type="S16"
    Volume: min=-7264, max=-241, dBmin=-2837, dBmax=-94
  Unit: 5
    Control: name="Mic Capture Switch", index=0
    Info: id=5, control=1, cmask=0x0, channels=1,
type="INV_BOOLEAN"
    Volume: min=0, max=1, dBmin=0, dBmax=0
```

```
rk3568_r:/ # cat /proc/asound/card2/stream0
Yundea Technology Yundea M1051 at usb-xhci-hcd.5.auto-1, full
speed : USB Audio

Capture:
  Status: Stop
  Interface 2
    Altset 1
    Format: S16_LE
    Channels: 1
    Endpoint: 3 IN (ASYNC)
    Rates: 16000
```

```
rk3568_r:/ # cat /proc/asound/card2/pcm0c/info
card: 2
device: 0
subdevice: 0
stream: CAPTURE
id: USB Audio
name: USB Audio
subname: subdevice #0
```

```
class: 0
subclass: 0
subdevices_count: 1
subdevices_avail: 1
```

录音测试

尝试录音失败，参数设置不对

```
rk3568_r:/ # tinycap /sdcard/test.pcm -D 2 -d 0 -T 5
Unable to open PCM device (cannot set hw params: Invalid
argument)
Captured 0 frames

rk3568_r:/ # tinycap /sdcard/test.pcm -D 2 -d 0 -c 2 -T 5
Unable to open PCM device (cannot set hw params: Invalid
argument)
Captured 0 frames
```

查看麦克风参数

```
rk3568_r:/ # cat /proc/asound/card2/stream0
Yundea Technology Yundea M1051 at usb-xhci-hcd.5.auto-1, full
speed : USB Audio

Capture:
  Status: Stop
  Interface 2
    Altset 1
    Format: S16_LE
    Channels: 1
    Endpoint: 3 IN (ASYNC)
    Rates: 16000
```

这里可以看到 通道是 1，码率是 16000，调整录音参数之后，正常录音

```
rk3568_r:/ # tinycap /sdcard/test.pcm -D 2 -d 0 -c 1 -r 16000 -
T 5
Capturing sample: 1 ch, 16000 hz, 16 bit
Captured 81920 frames
```

下载录音文件

```
neo@MacBook-Pro-M2 ~ % cd tmp
neo@MacBook-Pro-M2 tmp % adb pull /sdcard/test.pcm
/sdcard/test.pcm: 1 file pulled, 0 skipped. 21.1 MB/s (163884
bytes in 0.007s)
```

第 4 章 Activity

1. 定义 UI

```
<?xml version="1.0" encoding="utf-8"?>
<manifest
xmlns:android="http://schemas.android.com/apk/res/android"
     <application android:label="Test">

                ...
                ...
        <activity android:name=".WriteActivity"></activity>

    </application>

</manifest>
```

```
setContentView(R.layout.view);
```

2. 隐藏虚拟键

```
        int uiOptions = View.SYSTEM_UI_FLAG_FULLSCREEN |
View.SYSTEM_UI_FLAG_HIDE_NAVIGATION |
View.SYSTEM_UI_FLAG_IMMERSIVE;

getWindow().getDecorView().setSystemUiVisibility(uiOptions);

        View decorView = getWindow().getDecorView();
        int uiOptions = View.SYSTEM_UI_FLAG_HIDE_NAVIGATION
                | View.SYSTEM_UI_FLAG_IMMERSIVE_STICKY
                | View.SYSTEM_UI_FLAG_LAYOUT_HIDE_NAVIGATION
                | View.SYSTEM_UI_FLAG_LAYOUT_FULLSCREEN
                | View.SYSTEM_UI_FLAG_FULLSCREEN
                | View.SYSTEM_UI_FLAG_IMMERSIVE_STICKY;
        decorView.setSystemUiVisibility(uiOptions);
```

3. 显式四种跳转方式

```
    Intent intent = new
Intent(MainActivity.this,HomeActivity.class);
    startActivity(intent);

    Intent intent = new Intent();
    intent.setClass(MainActivity.this,HomeActivity.class);
    startActivity(intent);

    Intent intent = new Intent();
    ComponentName componentName = new
ComponentName(MainActivity.this,HomeActivity.class);
    intent.setComponent(componentName);
    startActivity(intent);


        startActivity(new
Intent(MainActivity.this,HomeActivity.class));
```

4. 定时关闭

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

        Toast.makeText(getApplicationContext(), "5秒后关闭",
Toast.LENGTH_SHORT).show();
        final Timer timer = new Timer();
        timer.schedule(new TimerTask() {
            public void run() {
                //结束本界面并跳转到收派员列表的界面
                finish();
            }
        }, 5000);

    }
```

```
        new Handler().postDelayed(new Runnable() {
        @Override
        public void run() {
            view.close();
        }
    }, 10000);
```

5. 恢复触发

程序回到桌面，例如设置WI-FI，让步在回到程序，安卓会调用onResume()

```
@Override
public void onResume() {
    super.onResume();
    this.other();
}
```

6. 返回触发

```
    @Override
    public void onBackPressed() {
        // code here to show dialog
        super.onBackPressed();  // optional depending on your
needs
        ...
    }
```

7. startActivity()

```
                Button button = (Button)
findViewById(R.id.writeButton);

        button.setOnClickListener(new View.OnClickListener()
{
            public void onClick(View v) {
                setContentView(R.layout.activity_write);
                Intent intent = new
Intent(MainActivity.this,WriteActivity.class);
                startActivity(intent);
            }
        });
```

8. Activity 间数据传递

8.1. Intent 方式

设置数据

```
Intent intent= new Intent();
intent.putExtra("name","zhangsan");
```

取出数据

```
 Intent intent = getIntent();
 String name=intent.getStringExtra("name");
```

8.2. Bundle 方式

```
Intent it = new Intent(Activity.Main.this, Activity2.class);
Bundle bundle=new Bundle();
bundle.putString("name", "This is from MainActivity!");
it.putExtras(bundle);
startActivity(it);
```

获取数据

```
Bundle bundle=getIntent().getExtras();
String name=bundle.getString("name");
```

8.3. Flag 属性

Flag属性用来设定Activity的启动模式

```
Intent intent = new Intent();
intent.setFlags(Intent.FLAG_ACTIVITY_CLEAR_TOP);
```

与清单文件中的设置launchMode属性值相同

```
Intent.FLAG_ACTIVITY_CLEAR_TOP = singleTask
Intent.FLAG_ACTIVITY_SINGLE_TOP = singleTop
Intent.FLAG_ACTIVITY_NEW_TASK = singleInstance
```

8.4. 返回值

有返回值的跳转

```
Intent intent = new
Intent(MainActivity.this,HomeActivity.class);
intent.putExtra("nickname","netkiller");
// 第一个参数Intent对象, 第二个参数 RequestCode
startActivityForResult(intent,REQUSET_CODE);
```

第一个参数 是不是我要的返回结果 第二个参数 是谁返回给我的 第三个参数 返回的附加信息

```
        @Override
    protected void onActivityResult(int requestCode, int
resultCode, @Nullable Intent intent) {
        super.onActivityResult(requestCode, resultCode,
intent);

        if(requestCode == REQUSET_CODE && resultCode ==
HomeActivity.RESULT_CODE){
            String msg = data.getStringExtra("msg");

Toast.makeText(MainActivity.this,msg,Toast.LENGTH_SHORT).show
();
        }
    }
```

返回结果

```
Intent intent = new Intent();
Intent oldIntent = getIntent();
String nickname = oldIntent.getStringExtra("nickname");
if(TextUtils.isEmpty(nickname)){
    intent.putExtra("msg",nickname);
}else{
    intent.putExtra("msg","Neo");
}

setResult(RESULT_CODE,intent);
//关闭页面
finish();
```

9. intentActivityResultLauncher 跳转

```
// 定义跳转
ActivityResultLauncher<Intent> intentActivityResultLauncher =
        registerForActivityResult(new
ActivityResultContracts.StartActivityForResult(), result -> {
    Intent data = result.getData();
    if (result.getResultCode() == RESULT_OK && data != null)
{
       // 一些逻辑
    }

});

// 使用时
Intent intent = new Intent(this,跳转到的.class);

// 执行跳转
intentActivityResultLauncher.launch(intent);
```

10. startActivityForResult 替代方案

startActivityForResult 即将废弃

```
    private void dispatchTakePictureIntent() {
        Intent takePictureIntent = new
Intent(MediaStore.ACTION_IMAGE_CAPTURE);
        try {
            startActivityForResult(takePictureIntent, 1);
        } catch (ActivityNotFoundException e) {
            // display error state to the user
        }
    }
```

替代方案是

```
   //拍照
    private final ActivityResultLauncher<Void>
mLauncherCamera = registerForActivityResult(
    new ActivityResultContracts.TakePicturePreview(), result
-> {
        //result为拍摄照片Bitmap格式
    });

    //开启拍照，返回结果Bitmap
    private void launchCamera() {
        mLauncherCamera.launch(null);
    }
```

11. Fragment

11.1. 在 Fragment 中使用 findViewById

提示

使用 getView() 方法返回当前 fragment 的根视图。

```
Button btn = getView().findViewById(R.id.btn);
```

11.2. 在 Fragment 中使用 Intent 跳转

```
Intent intent = new Intent(getActivity(), MyService.class);
startActivity(intent);
```

11.3. Fragment 中调用 getPackageManager()

```
ResolveInfo resolveInfo =
getActivity().getPackageManager().resolveActivity(intent, 0);
```

11.4. 在 Fragment 中使用 runOnUiThread

```
    private void showResponse(final String response) {
        //在子线程中更新UI
        getActivity().runOnUiThread(new Runnable() {
            @Override
            public void run() {
                text_dashboard.setText(response);
            }
        });
    }
```

第 5 章 Palette 视觉设计

1. 样式布局

1.1. 动画

参数

- android:oneshot 代表播放次数 true 只展示一遍，设置为false会不停的循环播放动画
- android:duration 表示展示所用的该图片的时间长度

```
<?xml version="1.0" encoding="utf-8"?>
<animation-list
 xmlns:android="http://schemas.android.com/apk/res/android"
 android:oneshot="true"

  <item android:drawable="@drawable/icon1"
android:duration="150" </item
  <item android:drawable="@drawable/icon2"
android:duration="150" </item
  <item android:drawable="@drawable/icon3"
android:duration="150" </item
  <item android:drawable="@drawable/icon4"
android:duration="150" </item
  <item android:drawable="@drawable/icon5"
android:duration="150" </item
  <item android:drawable="@drawable/icon6"
android:duration="150" </item
</animation-list
```

```
        <ImageView
    android:id="@+id/load_image"
    android:layout_width="25dp"
    android:layout_height="25dp"
    android:layout_gravity="center_vertical"
    android:scaleType="centerCrop"
    android:src="@drawable/loading_anim_image" />
```

```
    ImageView mImageView = findViewById(R.id.load_image);
    animationDrawable = (AnimationDrawable)
mImageView.getDrawable();
    //直接就开始执行，性能不是最佳的。
    animationDrawable.start();
```

1.2. 声音波形图

```
package cn.aigcsst.album.view;

import android.content.Context;
import android.graphics.Canvas;
import android.graphics.Color;
import android.graphics.Paint;
import android.util.AttributeSet;
import android.view.View;

public class SoundWaveView extends View {
    private Paint paint;
    private float[] amplitudes;

    public SoundWaveView(Context context) {
        super(context);
        init();
    }
```

```
    public SoundWaveView(Context context, AttributeSet attrs)
{
        super(context, attrs);
        init();
    }

    private void init() {
        paint = new Paint();
        paint.setColor(Color.BLUE);
        paint.setStrokeWidth(2);
    }

    public void setAmplitudes(float[] amplitudes) {
        this.amplitudes = amplitudes;
        invalidate(); // 刷新视图
    }

    @Override
    protected void onDraw(Canvas canvas) {
        super.onDraw(canvas);

        if (amplitudes == null) {
            return;
        }

        int width = getWidth();
        int height = getHeight();
        int centerY = height / 2;

        for (int i = 0; i < amplitudes.length; i++) {
            float x = width * i / amplitudes.length;
            float y = centerY + amplitudes[i] * centerY;
            canvas.drawLine(x, centerY, x, y, paint);
        }
    }
}
```

```
public class MainActivity extends AppCompatActivity {
    private SoundWaveView soundWaveView;
```

```
    private AudioRecord audioRecord;
    private boolean isRecording = false;

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

        soundWaveView = findViewById(R.id.sound_wave_view);
        int bufferSize = AudioRecord.getMinBufferSize(44100,
AudioFormat.CHANNEL_IN_MONO, AudioFormat.ENCODING_PCM_16BIT);
        audioRecord = new
AudioRecord(MediaRecorder.AudioSource.MIC, 44100,
AudioFormat.CHANNEL_IN_MONO, AudioFormat.ENCODING_PCM_16BIT,
bufferSize);

        startRecording();
    }

    private void startRecording() {
        isRecording = true;
        audioRecord.startRecording();

        new Thread(new Runnable() {
            @Override
            public void run() {
                short[] buffer = new short[1024];
                while (isRecording) {
                    int read = audioRecord.read(buffer, 0,
buffer.length);
                    if (read > 0) {
                        float[] amplitudes = new float[read];
                        for (int i = 0; i < read; i++) {
                            amplitudes[i] = buffer[i] /
32768f; // 归一化为[-1, 1]
                        }

soundWaveView.setAmplitudes(amplitudes);
                    }
                }
                audioRecord.stop();
            }
        }).start();
    }
```

```
    @Override
    protected void onDestroy() {
        super.onDestroy();
        isRecording = false;
    }
}
```

```
<com.example.myapplication.SoundWaveView
    android:id="@+id/sound_wave_view"
    android:layout_width="match_parent"
    android:layout_height="match_parent" />
```

第 6 章 UI 界面

1. Toast

```
Toast.makeText(getApplicationContext(), "默认Toast样式",
Toast.LENGTH_SHORT).show();
```

自定义样式

```
toast = Toast.makeText(getApplicationContext(),"自定义位置
Toast", Toast.LENGTH_LONG);
   toast.setGravity(Gravity.CENTER, 0, 0);
   toast.show();
```

带有图片的样式

```
toast = Toast.makeText(getApplicationContext(),"带图片的Toast",
Toast.LENGTH_LONG);
   toast.setGravity(Gravity.CENTER, 0, 0);
   LinearLayout toastView = (LinearLayout) toast.getView();
   ImageView imageView = new
ImageView(getApplicationContext());
   imageView.setImageResource(R.drawable.icon);
   toastView.addView(imageView, 0);
   toast.show();
```

2. Button

```
public class MainActivity extends AppCompatActivity {

    //我们需要自己写一个常量作为requestCode，在请求result时传递进去
    public static final int REQUEST_CODE_NORMAL = 100;

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

        Button button = (Button) findViewById(R.id.Button);

        button.setOnClickListener(new View.OnClickListener()
{
            public void onClick(View view) {
                                startActivityForResult(new
Intent(this,SecondActivity.class),REQUEST_CODE_NORMAL);
            }
        });
    }

    @Override
    protected void onActivityResult(int requestCode, int
resultCode, Intent data) {
        super.onActivityResult(requestCode, resultCode,
data);
        if (requestCode == REQUEST_CODE_NORMAL) {
                //获得Result数据并处理
            ...
            ...
        }
    }
}
```

```
public class SecondActivity extends AppCompatActivity {

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

        Button button = (Button)
findViewById(R.id.SaveButton);

        button.setOnClickListener(new View.OnClickListener()
{
            public void onClick(View view) {
                        Intent intent = new
Intent(this,MainResultActivity.class);

intent.putExtra("content",etContent.getText().toString());
                        setResult(1,intent);
                        //发送Result数据给请求方，然后finish ()
                        finish();
            }
        });
    }
}
```

启用禁用

```
myButton.setEnabled(false);
```

实现 OnClickListener 接口

```
package cn.netkiller.video;
```

```
import android.content.Intent;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;

public class MainActivity extends AppCompatActivity
implements View.OnClickListener {

    private Button buttonVideoView;
    private Button buttonSurfaceView;

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

        buttonVideoView = (Button)
findViewById(R.id.buttonVideoView);
        buttonVideoView.setOnClickListener(this);

        buttonSurfaceView = (Button)
findViewById(R.id.buttonSurfaceView);
        buttonSurfaceView.setOnClickListener(this);
    }

    @Override
    public void onClick(View v) {
        Intent intent;
        switch (v.getId()) {
            case R.id.buttonVideoView:
                startActivity(new Intent(MainActivity.this,
VideoViewActivity.class));
                break;
            case R.id.buttonSurfaceView:

                break;
            default:
                break;
        }
    }
}
```

Fragment 中使用 Button

```
                Button buttonWifi =
root.findViewById(R.id.buttonWifi);
        buttonWifi.setOnClickListener(new
View.OnClickListener() {
            @Override
            public void onClick(View view) {
                Intent intent = new Intent();
                ComponentName componentName = new
ComponentName("com.android.settings",
"com.android.settings.wifi.WifiSettings");
                intent.setComponent(componentName);
                ResolveInfo resolveInfo =
getActivity().getPackageManager().resolveActivity(intent, 0);
                if (resolveInfo != null) {
                    startActivity(intent);
                }
            }
        });
```

3. ListView

Array

```
        String[] list = Arrays.asList("Apple", "Banana",
"Orange", "Watermelon",
                "Pear", "Grape", "Pineapple", "Strawberry",
"Cherry", "Mango";
        ArrayAdapter<String> adapter = new
ArrayAdapter<String>(MainActivity.this,
android.R.layout.simple_list_item_1,data);
        ListView listView = (ListView)
findViewById(R.id.history);
        listView.setAdapter(adapter);
```

```
    <ListView
        android:id="@+id/history"
        android:layout_width="368dp"
        android:layout_height="444dp"
        android:scrollbars="horizontal"
        tools:layout_editor_absoluteX="8dp"
        tools:layout_editor_absoluteY="59dp" />
```

List

```
                List<String> list = Arrays.asList("Apple",
"Banana", "Orange", "Watermelon","Pear", "Grape",
"Pineapple", "Strawberry", "Cherry", "Mango");
        ArrayAdapter<String> adapter = new
```

```
ArrayAdapter<String>(MainActivity.this,
android.R.layout.simple_list_item_1,list);
        ListView listView = (ListView)
findViewById(R.id.history);
        listView.setAdapter(adapter);
```

setOnItemClickListener()

```
        List<String> list = Arrays.asList("Text 文本", "URL 网
址", "电话号码", "短信","开启应用", "地址", "日历", "图片", "邮箱",
"GPS 坐标");
        ArrayAdapter<String> adapter = new
ArrayAdapter<String>(this,
android.R.layout.simple_list_item_1,list);
        final ListView listView = (ListView)
findViewById(R.id.schemaList);
        listView.setAdapter(adapter);


        listView.setOnItemClickListener(new
AdapterView.OnItemClickListener() {
            @Override
            public void onItemClick(AdapterView<?> parent,
View view, int position, long id) {

                String text =
listView.getItemAtPosition(position)+"";
                Log.e("WRITE","position="+position+",
text="+text);
            }
        });
```

用接口方法实现

```
public class MainActivity extends Activity implements
OnItemClickListener, OnScrollListener
```

```
                List<String> list = Arrays.asList("Text 文本",
"URL 网址", "电话号码", "短信","开启应用", "地址", "日历", "图片",
"邮箱", "GPS 坐标");
        ArrayAdapter<String> adapter = new
ArrayAdapter<String>(this,
android.R.layout.simple_list_item_1,list);
        final ListView listView = (ListView)
findViewById(R.id.schemaList);
        listView.setAdapter(adapter);

        listView.setOnItemClickListener(this);
        listView.setOnScrollListener(this);
```

```
        @Override
    public void onItemClick(AdapterView<?> parent, View view,
int position, long id) {
        String text =
listView.getItemAtPosition(position)+"";
        Log.e("WRITE","position="+position+", text="+text);
    }
```

```
        @Override
    public void onScrollStateChanged(AbsListView view, int
scrollState) {
```

```
        switch (scrollState) {
        case SCROLL_STATE_FLING:
            Log.i("tag", "用户手指离开屏幕后，因惯性继续滑动");
            Map<String,Object>map = new
HashMap<String,Object>();
            map.put("icon", R.drawable.ic_launcher);
            map.put("text", "新增加项目");
            dataList.add(map);
            adapter.notifyDataSetChanged();
            break;
        case SCROLL_STATE_IDLE:
            Log.i("tag","已经停止滑动");
            break;
        case SCROLL_STATE_TOUCH_SCROLL:
            Log.i("tag", "手指未离开屏幕，屏幕继续滑动");
            break;
        }
    }
```

4. Switch

```
    <Switch
        android:id="@+id/switchWrite"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginStart="8dp"
        android:layout_marginTop="8dp"
        android:layout_marginEnd="8dp"
        android:layout_marginBottom="8dp"
        android:text="NDEF Message write"
        android:textOff="NDEF Message write Off"
        android:textOn="NDEF Message write On"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toStartOf="@+id/writeButton"
        app:layout_constraintHorizontal_bias="0.076"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/ndefWrite"
        app:layout_constraintVertical_bias="0.087" />
```

```
                final Switch switchWrite = (Switch)
findViewById(R.id.switchWrite);

        switchWrite.setOnCheckedChangeListener(new
CompoundButton.OnCheckedChangeListener() {
            @Override
            public void onCheckedChanged(CompoundButton
buttonView, boolean isChecked) {

                if(isChecked) {

status.setText(switchWrite.getTextOn().toString());
                } else {

status.setText(switchWrite.getTextOff().toString());
                }
```

```
            }
        });
```

```
        if(switchWrite.isChecked()){

    }
```

5. ProgressBar

6. Dialog

7. Menu

2. Widgets

2.1. ImageView

全屏显示

```
android:scaleType="matrix"
```

水平居中显示 android:layout_gravity="center"

```
<ImageView
    android:id="@+id/imageView"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:layout_gravity="center"
    app:srcCompat="@drawable/niboer"
    tools:ignore="MissingConstraints" />
```

剧中效果

android:scaleType="fitCenter"

```
<ImageView
    android:id="@+id/imageView"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:scaleType="fitCenter"
    app:srcCompat="@drawable/fo2"
```

```
        tools:ignore="MissingConstraints" />
```

ImageView 显示 URL 图片

```
   private Drawable getImageDrawableFromUrl(String url) {
        try {
            InputStream inputStream = (InputStream) new
URL(url).getContent();
            Drawable drawable =
Drawable.createFromStream(inputStream, "image.jpg");
//            Drawable drawable =
Drawable.createFromStream(new URL(url).openStream(),
"image.jpg");
//            Drawable drawable =
Drawable.createFromStream(getContext().getContentResolver().o
penInputStream(Uri.parse(url)), null);

            return drawable;
        } catch (IOException e) {
            Log.d("VoicePopup", e.getMessage());

        } catch (Exception e) {
            Log.d("VoicePopup", e.getMessage());
        }
        return null;
    }

    Drawable drawable = getImageDrawableFromUrl(image);
    imageView = findViewById(R.id.imageView);
    imageView.setImageDrawable(drawable);
```

```
                          imageView =
findViewById(R.id.imageView);
```

```
            new Thread(new Runnable() {
                @Override
                public void run() {
                    try {
                        InputStream is = (InputStream) new
URL(image).getContent();
                        final Drawable d =
Drawable.createFromStream(is, null);
                        getActivity().runOnUiThread(new
Runnable() {
                            @Override
                            public void run() {

imageView.setImageDrawable(d);
                            }
                        });
                    } catch (Exception e) {
                    }
                }
            }).start();
```

唱片播放效果（旋转PNG图片）

旋转 PNG 动画效果，用于显示唱片播放效果

UI 布局

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
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="wrap_content"
    android:layout_height="wrap_content"
    tools:context="cn.aigcsst.album.MainActivity">

    <ImageView
        android:id="@+id/imageView"
```

```
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:adjustViewBounds="true"
        android:scaleType="fitCenter"
        app:srcCompat="@drawable/fo2"
        tools:ignore="MissingConstraints" />

    <cn.aigcsst.album.view.SoundWaveView
        android:id="@+id/sound_wave_view"
        android:layout_width="match_parent"
        android:layout_height="50dp"
        android:layout_alignBottom="@id/imageView"

app:layout_constraintBottom_toBottomOf="@+id/imageView" />

    <ImageView
        android:id="@+id/imageViewWan"
        android:layout_width="50dp"
        android:layout_height="50dp"
        android:layout_gravity="center"

app:layout_constraintBottom_toTopOf="@+id/sound_wave_view"

app:layout_constraintEnd_toEndOf="@+id/sound_wave_view"

app:layout_constraintStart_toStartOf="@+id/sound_wave_view"
        app:srcCompat="@drawable/wan"
        tools:ignore="MissingConstraints" />
</androidx.constraintlayout.widget.ConstraintLayout>
```

旋转动画效果文件

创建旋转动画效果 res/anim/rotate.xml

```
<?xml version="1.0" encoding="utf-8"?>
<rotate
xmlns:android="http://schemas.android.com/apk/res/android">
    <rotate
        android:duration="2000"
```

```
        android:fromDegrees="0"
        android:pivotX="50%"
        android:pivotY="50%"
        android:repeatCount="-1"
        android:toDegrees="359"></rotate>
</rotate>
```

启动旋转效果

```
        Animation rotateAnimation =
AnimationUtils.loadAnimation(this, R.anim.rotate);
        LinearInterpolator linearInterpolator = new
LinearInterpolator();
        rotateAnimation.setInterpolator(linearInterpolator);
        View imageViewWan = findViewById(R.id.imageViewWan);
        imageViewWan.startAnimation(rotateAnimation);
```

3. Legacy

3.1. GardView

3.2. GridView

第 7 章 Schedule 计划任务

1. Time 和 TimerTask 定时刷新

```
package cn.netkiller.okhttp;

import android.os.Handler;
import android.os.Message;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.widget.TextView;

import java.text.DateFormat;
import java.text.SimpleDateFormat;
import java.util.Date;
import java.util.Timer;
import java.util.TimerTask;

public class ScheduleActivity extends AppCompatActivity {

    private TextView clock;

    final Handler handler = new Handler() {
        public void handleMessage(Message msg) {
            super.handleMessage(msg);
            switch (msg.what) {
                case 1:
                    update(msg.obj.toString());
                    break;
            }
        }

        void update(String c) {

            clock.setText(c);
        }
    };
```

```
    Timer timer = new Timer();
    TimerTask task = new TimerTask() {
        DateFormat dateFormat = new
SimpleDateFormat("yyyy/MM/dd HH:mm:ss");

        public void run() {
            Message message = new Message();
            message.what = 1;
            message.obj = dateFormat.format(new Date());
            handler.sendMessage(message);
        }
    };


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

        clock = (TextView) findViewById(R.id.clock);
        clock.setText("Today is ...");
        timer.schedule(task, 1000 * 5, 1000); //启动timer

    }

    @Override
    protected void onDestroy() {
        super.onDestroy();
        if (timer != null) {
            timer.cancel();
            timer = null;
        }

    }

}
```

2. 使用 Runnable 和 Handler 实现定时执行

原理是使用 handler.postDelayed 延迟 Runnable 的运行时间

```
package cn.netkiller.okhttp;

import android.os.Handler;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.widget.TextView;

import java.text.DateFormat;
import java.text.SimpleDateFormat;
import java.util.Date;

public class RunnableActivity extends AppCompatActivity {

    private Handler handler = new Handler();
    private Runnable runnable = new Runnable() {
        public void run() {
            this.update();
            handler.postDelayed(this, 1000);// 1000 ms = 1s 间
隔1秒
        }

        void update() {
            DateFormat dateFormat = new
SimpleDateFormat("yyyy-MM-dd HH:mm:ss");
            time.setText(dateFormat.format(new Date()));
        }
    };
    private TextView time;

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

        time = (TextView) findViewById(R.id.time);
        time.setText("Start...");
```

```
        handler.postDelayed(runnable, 1000 * 5); // 5 秒后开始
    }

    @Override
    protected void onDestroy() {
        super.onDestroy();
        handler.removeCallbacks(runnable);
    }

}
```

3. TimerTask 更新 UI

```
        Timer timer = new Timer();

        private void Clock() {

        TextView textViewTime =
findViewById(R.id.textViewTime);
        TimerTask task = new TimerTask() {
            DateFormat dateFormat = new
SimpleDateFormat("yyyy/MM/dd HH:mm:ss");

            public void run() {
                String current = dateFormat.format(new
Date());
                textViewTime.post(new Runnable() {
                    @Override
                    public void run() {
                        textViewTime.setText(current);
                    }
                });
                Log.d(TAG, current);
            }
        };
        timer.schedule(task, 1000 * 5, 1000);
//        timer.cancel();
    }
```

第 8 章 Internationalization i18n with Android (国际化)

1. 创建国际化文件

进入 Android Studio 文件菜单 File -> New -> New Resource File

在左侧列表中找到 Locale 点击 “>>” 按钮

选择国家后，点击 OK 按钮即可。

资源文件夹中已经显示出国际化文件，上面并有对应的国旗。

查看项目文件夹

```
neo@MacBook-Pro ~/AndroidStudioProjects/locale % find
app/src/main/res | grep values
app/src/main/res/values-zh-rCN
app/src/main/res/values-zh-rCN/strings.xml
app/src/main/res/values
app/src/main/res/values/colors.xml
app/src/main/res/values/dimens.xml
app/src/main/res/values/styles.xml
app/src/main/res/values/strings.xml
```

2. strings.xml 文件

```
<resources>
    <string name="app_name">Netkiller</string>
    <string name="title_home">Home</string>
    <string name="title_dashboard">Dashboard</string>
    <string name="title_notifications">Notifications</string>
</resources>
```

3. 翻译语言

再 res/values/strings 目录上面单击鼠标右键，打开 Open Translations Editor 翻译编辑器。

单击地球图标，添加 zh-cn 语言

现在就可以对照翻译语言包文件了。

4. 引用国际化文件

```
String test = "Sign Up";

String test = getResources().getString(R.string.sign_up);
```

```
R.string.browserSentence = "You are using $1%s to browse the
Internet.";

String browser = getString(R.string.browserSentence,
browser.getBrowser());
```

```
TextView textView = new TextView(this);
TextView.setText(“Sign Up”);

TextView textView = new TextView(this);
textView.setText(R.string.sign_up);
```

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

<TextView
 android:layout_width="wrap_content"
 android:layout_height="wrap_content"
```

```
 android:text="@string/hello_world" />
```

5. 切换语言

```
        DisplayMetrics metrics =
getResources().getDisplayMetrics();
        Configuration configuration =
getResources().getConfiguration();
        configuration.setLocale(locale);
        getResources().updateConfiguration(configuration,
metrics);

        //重新启动 Activity
        Intent intent = new Intent(this, MainActivity.class);
        intent.setFlags(Intent.FLAG_ACTIVITY_NEW_TASK |
Intent.FLAG_ACTIVITY_CLEAR_TASK);
        startActivity(intent);
```

第 9 章 存储

1. SharedPreferences

SharedPreferences是Android中的数据存储技术，常用来存储一些轻量级的数据。

实际上SharedPreferences是 NoSQL 数据库，用于处理的key-value键值对存储，类似Windows 系统的注册表，Linux 系统里的 Berkeley DB (bdb) 以及衍生出的 dba,mdb 这类 hash 表数据库。

1.1. 操作模式

```
Context.MODE_PRIVATE：为默认操作模式,代表该文件是私有数据,只能被应用本身访问,在该模式下,写入的内容会覆盖原文件的内容
Context.MODE_APPEND：模式会检查文件是否存在,存在就往文件追加内容,否则就创建新文件.
Context.MODE_WORLD_READABLE和Context.MODE_WORLD_WRITEABLE用来控制其他应用是否有权限读写该文件.
MODE_WORLD_READABLE：表示当前文件可以被其他应用读取.
MODE_WORLD_WRITEABLE：表示当前文件可以被其他应用写入.
```

1.2. 保存数据

```
        Button buttonPut = (Button)
findViewById(R.id.buttonPut);

        buttonPut.setOnClickListener(new
View.OnClickListener() {
            public void onClick(View view) {
```

```
                //实例化SharedPreferences对象
                SharedPreferences sharedPreferences =
getSharedPreferences("test", Activity.MODE_PRIVATE);

                //实例化SharedPreferences.Editor对象
                SharedPreferences.Editor editor =
sharedPreferences.edit();

                //用putString的方法保存数据
                editor.putString("name", "Neo");
                editor.putString("nickname", "netkiller");
                editor.putBoolean("sex", true);
                editor.putInt("age", 30);
                editor.putFloat("tall", 180.23f);
                Set<String> books = new HashSet<String>();
                books.add("Netkiller Linux 手札");
                books.add("Netkiller Java 手札");
                books.add("Netkiller Android 手札");
                editor.putStringSet("books", books);

                //提交当前数据
                editor.commit();

            }
        });
```

1.3. 读取数据

```
        Button buttonGet = (Button)
findViewById(R.id.buttonGet);

        buttonGet.setOnClickListener(new
View.OnClickListener() {
            public void onClick(View view) {

                //实例化SharedPreferences对象
                SharedPreferences sharedPreferences =
```

```
getSharedPreferences("test", Activity.MODE_PRIVATE);

                //使用getString方法获得value,
                String name =
sharedPreferences.getString("name", "");
                String nickname =
sharedPreferences.getString("nickname", "");
                boolean sex =
sharedPreferences.getBoolean("sex", false);
                int age = sharedPreferences.getInt("age", 0);
                float tall =
sharedPreferences.getFloat("tall", 0f);
                Set<String> books =
sharedPreferences.getStringSet("books", null);

                Log.i("SharedPreferences",
String.format("%s,%s,%s,%s,%s,%s", name, nickname, sex, age,
tall, books.toString()));

            }
        });
```

1.4. 通过 key 查询数据是否存在

```
                SharedPreferences sharedPreferences =
getSharedPreferences("test", Activity.MODE_PRIVATE);
                if (sharedPreferences.contains("nickname")) {
            Log.i("SharedPreferences",
sharedPreferences.getString("nickname", ""));
        }else{
            Log.i("SharedPreferences", "key: nickname 不存
在");
        }
```

1.5. 删除数据

```
        SharedPreferences sharedPreferences =
getSharedPreferences("test", Activity.MODE_PRIVATE);
        SharedPreferences.Editor editor =
sharedPreferences.edit();

    editor.remove("nickname");
    editor.commit();

    Log.i("SharedPreferences",
sharedPreferences.getAll().toString());
```

1.6. 清空数据

```
        SharedPreferences sharedPreferences =
getSharedPreferences("test", Activity.MODE_PRIVATE);
        SharedPreferences.Editor editor =
sharedPreferences.edit();
    editor.clear();
    editor.commit();

    Log.i("SharedPreferences",
sharedPreferences.getAll().toString());
```

1.7. 对象存储

对象存储，需要将对象序列化，然后反序列化

1.8. SharedPreferences 读取物理存储文件

SharedPreferences 的数据存储再一个 xml 文件中，他的地址是:

```
//<package name>应替换成应用的包名, <name>
File xmlFile = new File("/data/data/<package
name>/shared_prefs/<name>.xml");
```

```
<?xml version='1.0' encoding='utf-8' standalone='yes' ?>
<map>
   <string name="name">陈景峰</string>
   <string name="nickname">netkiller</string>
   <int name="age" value="30" />
</map>
```

2. SD Card

2.1. SD Card 状态

```
        if
(!Environment.getExternalStorageState().equals(Environment.ME
DIA_MOUNTED)) {
                    TextView textView = (TextView)
findViewById(R.id.textView);
                    textView.setText("SD 卡不存在，请插入 SD
卡！");
                }
```

2.2. Android 11 申请 sdcard 权限

```
        private static final int REQUEST_CODE = 1024;

    private void requestPermission() {
        if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.R) {
            // 先判断有没有权限
            if (!Environment.isExternalStorageManager()) {
                Intent intent = new
Intent(Settings.ACTION_MANAGE_APP_ALL_FILES_ACCESS_PERMISSION
);
                intent.setData(Uri.parse("package:" +
getPackageName()));
                startActivityForResult(intent, REQUEST_CODE);
            }
        } else if (Build.VERSION.SDK_INT >=
Build.VERSION_CODES.M) {
            // 先判断有没有权限
            if (ActivityCompat.checkSelfPermission(this,
Manifest.permission.READ_EXTERNAL_STORAGE) ==
PackageManager.PERMISSION_GRANTED &&
```

```
                    ContextCompat.checkSelfPermission(this,
Manifest.permission.WRITE_EXTERNAL_STORAGE) ==
PackageManager.PERMISSION_GRANTED) {
//                  写入文件
            } else {
                ActivityCompat.requestPermissions(this, new
String[]{Manifest.permission.READ_EXTERNAL_STORAGE,
Manifest.permission.WRITE_EXTERNAL_STORAGE}, REQUEST_CODE);
            }
        }
    }

    @Override
    public void onRequestPermissionsResult(int requestCode,
@NonNull String[] permissions, @NonNull int[] grantResults) {
        super.onRequestPermissionsResult(requestCode,
permissions, grantResults);
        if (requestCode == REQUEST_CODE) {
            if (ActivityCompat.checkSelfPermission(this,
Manifest.permission.READ_EXTERNAL_STORAGE) ==
PackageManager.PERMISSION_GRANTED &&
                    ContextCompat.checkSelfPermission(this,
Manifest.permission.WRITE_EXTERNAL_STORAGE) ==
PackageManager.PERMISSION_GRANTED) {
//                  写入文件
            } else {
//                  ToastUtils.show("存储权限获取失败");
            }
        }
    }

    @Override
    protected void onActivityResult(int requestCode, int
resultCode, @Nullable Intent data) {
        super.onActivityResult(requestCode, resultCode,
data);
        if (requestCode == REQUEST_CODE &&
Build.VERSION.SDK_INT >= Build.VERSION_CODES.R) {
            if (Environment.isExternalStorageManager()) {
//                  写入文件
            } else {
//                  ToastUtils.show("存储权限获取失败");
            }
        }
    }
```

第 10 章 网络

1. Wifi 配置

方法一

```
        context.startActivity(new
Intent(Settings.ACTION_WIFI_SETTINGS));
```

方法二

```
        Button buttonWifi =
root.findViewById(R.id.buttonWifi);
        buttonWifi.setOnClickListener(new
View.OnClickListener() {
            @Override
            public void onClick(View view) {
                Intent intent = new Intent();
                ComponentName componentName = new
ComponentName("com.android.settings",
"com.android.settings.wifi.WifiSettings");
                intent.setComponent(componentName);
                ResolveInfo resolveInfo =
getActivity().getPackageManager().resolveActivity(intent, 0);
                if (resolveInfo != null) {
                    startActivity(intent);
                }
            }
        });
```

2. OkHttp - An HTTP & HTTP/2 client for Android and Java applications

http://square.github.io/okhttp/

2.1. Gradle

再 app/build.gradle 文件中增加依赖包

```
implementation "com.squareup.okhttp3:okhttp:4.10.0"
```

app/build.gradle

```
neo@MacBook-Pro ~/AndroidStudioProjects/okhttp % cat
app/build.gradle

apply plugin: 'com.android.application'

android {
    compileSdkVersion 28
    defaultConfig {
        applicationId "cn.netkiller.okhttp"
        minSdkVersion 28
        targetSdkVersion 28
        versionCode 1
        versionName "1.0"
        testInstrumentationRunner
"android.support.test.runner.AndroidJUnitRunner"
    }
    buildTypes {
        release {
            minifyEnabled false
            proguardFiles getDefaultProguardFile('proguard-
android.txt'), 'proguard-rules.pro'
```

```
        }
    }
}

dependencies {
    implementation fileTree(dir: 'libs', include: ['*.jar'])
    implementation 'com.android.support:appcompat-v7:28.0.0'
    implementation 'com.android.support:design:28.0.0'
    implementation 'com.android.support.constraint:constraint-
layout:1.1.3'
    testImplementation 'junit:junit:4.12'
    androidTestImplementation
'com.android.support.test:runner:1.0.2'
    androidTestImplementation
'com.android.support.test.espresso:espresso-core:3.0.2'
    implementation 'com.squareup.okhttp3:okhttp:3.11.0'
}
```

2.2. AndroidManifest.xml 开启网络访问权限

在 app/src/main/AndroidManifest.xml 文件中增加

```
<uses-permission android:name="android.permission.INTERNET" />
```

否则 okhttp 无法访问网络，添加后的效果如下。

```
neo@MacBook-Pro ~/AndroidStudioProjects/okhttp % cat
app/src/main/AndroidManifest.xml
<?xml version="1.0" encoding="utf-8"?>
<manifest
xmlns:android="http://schemas.android.com/apk/res/android"
    package="cn.netkiller.okhttp">
    <uses-permission android:name="android.permission.INTERNET"
/>
```

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

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

</manifest>
```

2.3. okhttp 默认是 HTTPS 开启 HTTP

如果你尝试使用 http 链接 OkHttp3 就抛出异常: CLEARTEXT communication to " + host + " not permitted by network security policy

开发过程中由于 https ssl 需要购买证书，费用较高，通常测试环境我们仍然使用 http 下面方法是开启 http 模式，

创建文件 res/xml/network_security_config.xml 内容如下

```
neo@MacBook-Pro ~/AndroidStudioProjects/okhttp % cat
app/src/main/res/xml/network_security_config.xml
<?xml version="1.0" encoding="utf-8"?>
<network-security-config>
    <domain-config cleartextTrafficPermitted="true">
        <domain includeSubdomains="true">localhost</domain>
```

```
    </domain-config>
</network-security-config>
```

再 app/src/main/AndroidManifest.xml 文件中增加 android:networkSecurityConfig="@xml/network_security_config"

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

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/AppTheme">
        <activity
            android:name=".MainActivity"
            android:label="@string/app_name"

android:networkSecurityConfig="@xml/network_security_config">
            <intent-filter>
                <action
android:name="android.intent.action.MAIN" />

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

    <uses-permission android:name="android.permission.INTERNET"
/>
</manifest>
```

2.4. GET

```
        OkHttpClient client = new OkHttpClient();

    Request request = new Request.Builder()
            .url("http://www.netkiller.cn")
            .build();

    Response response = client.newCall(request).execute();
    if (!response.isSuccessful()) {
        throw new IOException("服务器端错误: " + response);
    }

    Headers responseHeaders = response.headers();
    for (int i = 0; i < responseHeaders.size(); i++) {
        System.out.println(responseHeaders.name(i) + ": " +
responseHeaders.value(i));
    }

    System.out.println(response.body().string());
```

URL 组装

```
                HttpUrl.Builder urlBuilder =
HttpUrl.parse("https://www.netkiller.cn").newBuilder();

urlBuilder.addPathSegments("search/cache_vector_chatgpt");
                urlBuilder.addQueryParameter("question",
"Helloworld!!!");
                String url = urlBuilder.build().toString();

                Log.d("API", url);
                OkHttpClient client = new OkHttpClient();
                try {
                    Request request = new
Request.Builder().url(url).build();
```

```
                    Response response =
client.newCall(request).execute();
                    if (response.isSuccessful()) {
                        Log.d("API",
response.body().string());
                    }
                } catch (IOException e) {
                    throw new RuntimeException(e);
                }
```

URL 输出

```
https://www.netkiller.cn/search/cache_vector_chatgpt?
question=Helloworld%21%21%21
```

Get 异步调用

```
        OkHttpClient client = new OkHttpClient();

        Request request = new
Request.Builder().url(url).build();
        Call call = client.newCall(request);
        call.enqueue(new Callback() {
            @Override
            public void onFailure(@NotNull Call call,
@NotNull IOException e) {

            }

            @Override
            public void onResponse(@NotNull Call call,
@NotNull Response response) throws IOException {
                if (response.isSuccessful()) {
                    Log.i("TAG", "Async: " +
```

```
response.body().string());
                }
            }
        });
```

2.5. POST

POST Form Data

from 数据如下

```
key=value&other=data
```

```
                String url =
"https://api.netkiller.cn/oauth/token";

                OkHttpClient client = new OkHttpClient();

        RequestBody formBody = new FormBody.Builder()
                .add("grant_type", "password")
                .add("username", "netkiller")
                .add("password", "123456")
                .build();

        Request request = new Request.Builder()
                .url(url)
                .post(formBody)
                .build();

        Response response = client.newCall(request).execute();
        if (!response.isSuccessful()) {
            throw new IOException("服务器端错误: " + response);
        }
```

POST RAW JSON

POST RAW Json 数据，数据的形态这样的

```
{"key":"value"}
```

```
public static final MediaType JSON =
MediaType.parse("application/json; charset=utf-8");

OkHttpClient client = new OkHttpClient();

String post(String url, String json) throws IOException {
  RequestBody body = RequestBody.create(JSON, json);
  Request request = new Request.Builder()
      .url(url)
      .post(body)
      .build();
  Response response = client.newCall(request).execute();
  return response.body().string();
}
```

数据流提交

```
OkHttpClient client = new OkHttpClient();
final MediaType MEDIA_TYPE_TEXT =
MediaType.parse("text/plain");
final String postBody = "Hello World";
```

```
RequestBody requestBody = new RequestBody() {
    @Override
    public MediaType contentType() {
        return MEDIA_TYPE_TEXT;
    }
    @Override
    public void writeTo(BufferedSink sink) throws IOException
{
        sink.writeUtf8(postBody);
    }
    @Override
    public long contentLength() throws IOException {
        return postBody.length();
    }
};

Request request = new Request.Builder()
        .url("https://www.google.com")
        .post(requestBody)
        .build();
Response response = client.newCall(request).execute();
if (!response.isSuccessful()) {
    throw new IOException("服务器端错误: " + response);
}
System.out.println(response.body().string());
```

2.6. http header 相关设置

设置 HTTP 头

添加Http头

```
Request request = new Request.Builder()
        .url(url)
        .addHeader("CLIENT", "AD")
        .addHeader("USERID", "343")
        .build();
```

覆盖 HTTP 头

```
        Request request = new Request.Builder()
                .header("Authorization", "replace this text
with your token")
                .url(url)
                .build();
```

```
public class Headers {
   public static void main(String[] args) throws IOException
{
    OkHttpClient client = new OkHttpClient();

    Request request = new Request.Builder()
            .url("http://www.netkiller.cn")
            .header("User-Agent", "Apple Mac")
            .addHeader("Accept", "text/html")
            .build();

    Response response = client.newCall(request).execute();
    if (!response.isSuccessful()) {
        throw new IOException("服务器端错误: " + response);
    }

    System.out.println(response.header("Server"));
    System.out.println(response.headers("Set-Cookie"));
   }
}
```

Cookie 管理

```
OkHttpClient mHttpClient = new
OkHttpClient.Builder().cookieJar(new CookieJar() {
    private final HashMap<String, List<Cookie>> cookieStore =
new HashMap<>();
    @Override
    public void saveFromResponse(HttpUrl url, List<Cookie>
cookies) {
        cookieStore.put(url.host(), cookies);
    }
    @Override
    public List<Cookie> loadForRequest(HttpUrl url) {
        List<Cookie> cookies = cookieStore.get(url.host());
        return cookies != null ? cookies : new
ArrayList<Cookie>();
    }
}).build();
```

禁用缓存

```
Request request = new Request.Builder()
    .cacheControl(new
CacheControl.Builder().noCache().build())
    .url(url)
    .build();
```

设置缓存 max-age

```
// 等同于 nocache
Request request = new Request.Builder()
    .cacheControl(new CacheControl.Builder()
```

```
        .maxAge(0, TimeUnit.SECONDS)
        .build())
    .url("https://www.netkiller.cn")
    .build();

// 设置缓存 365 天
Request request = new Request.Builder()
    .cacheControl(new CacheControl.Builder()
        .maxStale(365, TimeUnit.DAYS)
        .build())
    .url("https://www.netkiller.cn")
    .build();
```

强制缓存

```
Request request = new Request.Builder()
    .cacheControl(new CacheControl.Builder()
        .onlyIfCached()
        .build())
    .url("https://www.netkiller.cn/helloworld.txt")
    .build();
Response forceCacheResponse =
client.newCall(request).execute();
if (forceCacheResponse.code() != 504) {
  // The resource was cached! Show it.
} else {
  // The resource was not cached.
}
```

2.7. HTTP Base Auth

```
        OkHttpClient client = new
OkHttpClient.Builder().authenticator(
```

```
                new Authenticator(){
                    public Request authenticate(Route route,
Response response) {
                        String credential =
Credentials.basic("api","secret");
                        return
response.request().newBuilder().header("Authorization",
credential).build();
                    }
                }
        ).build();
```

2.8. HttpUrl.Builder 组装 URL 地址参数

使用字符串拼接 URL地址特别容易出错

```
String url = "https://www.netkiller.cn/article?username="+
username + "&category="+ category;
```

较好的处理方式是使用 HttpUrl.Builder

```
                HttpUrl.Builder builder =
HttpUrl.parse("https://www.netkiller.cn/article").newBuilder(
);
        builder.addQueryParameter("username", "netkiller");
        builder.addQueryParameter("category", "android");
        String url = builder.build().toString();

        Log.d("okhttp", url);
```

输出结果

```
https://www.netkiller.cn/article?
username=netkiller&category=android
```

2.9. Android Activity Example

```
package cn.netkiller.okhttp;

import android.os.Handler;
import android.os.Looper;
import android.os.Message;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.util.Log;
import android.widget.TextView;

import java.io.IOException;

import okhttp3.Call;
import okhttp3.Callback;
import okhttp3.OkHttpClient;
import okhttp3.Request;
import okhttp3.Response;

public class HttpActivity extends AppCompatActivity {

    TextView textView;
    private Handler mHandler;

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

        textView = (TextView) findViewById(R.id.textView);

        mHandler = new Handler(Looper.getMainLooper()) {
            @Override
```

```
            public void handleMessage(Message msg) {
                textView.setText((String) msg.obj);
            }
        };

        OkHttpClient client = new OkHttpClient();
        Request request = new
Request.Builder().url("https://api.netkiller.cn/member/json")
.build();
        client.newCall(request).enqueue(new Callback() {
            @Override
            public void onFailure(Call call, IOException e) {
                Log.d("okhttp", "Connect timeout. " +
e.getMessage());
            }

            @Override
            public void onResponse(Call call, Response
response) throws IOException {
                String text = response.body().string();
                Log.d("okhttp", "HTTP Code " +
response.code() + " TEXT " + text);

                Message msg = new Message();
                msg.what = 0;
                msg.obj = text;
                mHandler.sendMessage(msg);

            }
        });

    }
}
```

2.10. Android Oauth2 + Jwt example

Oauth 返回数据，通过 Gson 的 fromJson 存储到下面类中。

```
package cn.netkiller.okhttp.pojo;
```

```
public class Oauth {
    private String access_token;
    private String token_type;
    private String refresh_token;
    private String expires_in;
    private String scope;
    private String jti;

    public String getAccess_token() {
        return access_token;
    }

    public void setAccess_token(String access_token) {
        this.access_token = access_token;
    }

    public String getToken_type() {
        return token_type;
    }

    public void setToken_type(String token_type) {
        this.token_type = token_type;
    }

    public String getRefresh_token() {
        return refresh_token;
    }

    public void setRefresh_token(String refresh_token) {
        this.refresh_token = refresh_token;
    }

    public String getExpires_in() {
        return expires_in;
    }

    public void setExpires_in(String expires_in) {
        this.expires_in = expires_in;
    }

    public String getScope() {
        return scope;
    }
```

```
    public void setScope(String scope) {
        this.scope = scope;
    }

    public String getJti() {
        return jti;
    }

    public void setJti(String jti) {
        this.jti = jti;
    }

    @Override
    public String toString() {
        return "Oauth{" +
                "access_token='" + access_token + '\'' +
                ", token_type='" + token_type + '\'' +
                ", refresh_token='" + refresh_token + '\'' +
                ", expires_in='" + expires_in + '\'' +
                ", scope='" + scope + '\'' +
                ", jti='" + jti + '\'' +
                '}';
    }
}
```

Activity 文件

```
package cn.netkiller.okhttp;

import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.util.Log;
import android.widget.TextView;

import com.google.gson.Gson;

import java.io.IOException;

import cn.netkiller.okhttp.pojo.Oauth;
```

```
import okhttp3.Authenticator;
import okhttp3.Call;
import okhttp3.Callback;
import okhttp3.Credentials;
import okhttp3.FormBody;
import okhttp3.OkHttpClient;
import okhttp3.Request;
import okhttp3.RequestBody;
import okhttp3.Response;
import okhttp3.Route;

public class Oauth2jwtActivity extends AppCompatActivity {

    private TextView token;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_oauth2jwt);

        token = (TextView) findViewById(R.id.token);

        try {
            get();
        } catch (IOException e) {
            e.printStackTrace();
        }

    }


    public static Oauth accessToken() throws IOException {


        OkHttpClient client = new
OkHttpClient.Builder().authenticator(
                new Authenticator() {
                    public Request authenticate(Route route,
Response response) {
                        String credential =
Credentials.basic("api", "secret");
                        return
response.request().newBuilder().header("Authorization",
credential).build();
                    }
```

```
                }
        ).build();

        String url = "https://api.netkiller.cn/oauth/token";

        RequestBody formBody = new FormBody.Builder()
                .add("grant_type", "password")
                .add("username", "blockchain")
                .add("password", "123456")
                .build();

        Request request = new Request.Builder()
                .url(url)
                .post(formBody)
                .build();

        Response response =
client.newCall(request).execute();
        if (!response.isSuccessful()) {
            throw new IOException("服务器端错误: " + response);
        }

        Gson gson = new Gson();
        Oauth oauth = gson.fromJson(response.body().string(),
Oauth.class);
        Log.i("oauth", oauth.toString());
        return oauth;
    }

    public void get() throws IOException {


        OkHttpClient client = new
OkHttpClient.Builder().authenticator(
                new Authenticator() {
                    public Request authenticate(Route route,
Response response) throws IOException {
                        return
response.request().newBuilder().header("Authorization",
"Bearer " + accessToken().getAccess_token()).build();
                    }
                }
        ).build();

        String url =
```

```
"https://api.netkiller.cn/misc/compatibility";

        Request request = new Request.Builder()
                .url(url)
                .build();

        client.newCall(request).enqueue(new Callback() {
            @Override
            public void onFailure(Call call, IOException e) {
                call.cancel();
            }

            @Override
            public void onResponse(Call call, Response
response) throws IOException {

                final String myResponse =
response.body().string();

                runOnUiThread(new Runnable() {
                    @Override
                    public void run() {
                        Log.i("oauth", myResponse);
                        token.setText(myResponse);
                    }
                });

            }
        });
    }

        public void post() throws IOException {


        OkHttpClient client = new
OkHttpClient.Builder().authenticator(
                new Authenticator() {
                    public Request authenticate(Route route,
Response response) throws IOException {
                        return
response.request().newBuilder().header("Authorization",
"Bearer " + accessToken().getAccess_token()).build();
                    }
                }
        ).build();
```

```
        String url =
"https://api.netkiller.cn/history/create";

        String json = "{\n" +
                "  \"assetsId\":
\"5bced71c432c001c6ea31924\",\n" +
                "  \"title\": \"添加信息\",\n" +
                "  \"message\": \"信息内容\",\n" +
                "  \"status\": \"录入\"\n" +
                "}";

        final MediaType JSON =
MediaType.parse("application/json; charset=utf-8");
        RequestBody body = RequestBody.create(JSON, json);

        Request request = new Request.Builder()
                .url(url)
                .post(body)
                .build();

        client.newCall(request).enqueue(new Callback() {
            @Override
            public void onFailure(Call call, IOException e) {
                call.cancel();
            }

            @Override
            public void onResponse(Call call, Response
response) throws IOException {

                final String myResponse =
response.body().string();

                runOnUiThread(new Runnable() {
                    @Override
                    public void run() {

//                        Gson gson = new Gson();
//                        Oauth oauth =
gson.fromJson(myResponse, Oauth.class);
//                        Log.i("oauth", oauth.toString());

                        token.setText(myResponse);
                    }
```

```
                    });

                }
            });
        }

    }
```

上面的例子演示了，怎样获取 access token 以及 HTTP 基本操作 GET 和 POST，再Restful接口中还可能会用到 PUT, DELETE 等等，原来相同，这里就不在演示。

2.11. HTTP/2

首先确认你的服务器是支持 HTTP2，HTTP2配置方法可以参考《Netkiller Web 手札》

下面是我的例子仅供参考，Nginx 开启 http2 代理后面的 Spring Cloud 接口。

```
server {
    listen       80;
    listen 443 ssl http2;
    server_name  api.netkiller.cn;

    ssl_certificate ssl/netkiller.cn.crt;
    ssl_certificate_key ssl/netkiller.cn.key;
    ssl_session_cache shared:SSL:20m;
    ssl_session_timeout 60m;

    charset utf-8;
    access_log  /var/log/nginx/api.netkiller.cn.access.log;

    error_page  497              https://$host$uri?$args;

    if ($scheme = http) {
        return 301 https://$server_name$request_uri;
```

```
    }

    location / {
                proxy_pass   http://127.0.0.1:8000;
        proxy_http_version 1.1;
                proxy_set_header    Host    $host;
                proxy_set_header    X-Forwarded-For
$proxy_add_x_forwarded_for;
    }

    #error_page  404              /404.html;

    # redirect server error pages to the static page /50x.html
    #
    error_page   500 502 503 504  /50x.html;
    location = /50x.html {
        root   /usr/share/nginx/html;
    }

    # deny access to .htaccess files, if Apache's document root
    # concurs with nginx's one
    #
    #location ~ /\.ht {
    #    deny  all;
    #}
}
```

安卓程序如下

```
                String url =
"https://api.netkiller.cn/member/json";

        OkHttpClient client = new OkHttpClient();
        Request request = new
Request.Builder().url(url).build();
        client.newCall(request).enqueue(new Callback() {
            @Override
            public void onFailure(Call call, IOException e) {
                Log.d("okhttp", "Connect timeout. " +
e.getMessage());
```

```
            }

            @Override
            public void onResponse(Call call, Response
response) throws IOException {
                String text = response.body().string();
                Log.d("okhttp", "HTTP Code " +
response.code() + " TEXT " + text);
                Log.d("okhttp", "Protocol: " +
response.protocol());
            }
        });
```

运行结果

```
D/okhttp: HTTP Code 200 TEXT
{"status":false,"reason":"","code":0,"data":
{"id":null,"username":"12322228888","mobile":"12322228888","pas
sword":"123456","wechat":"微信
ID","role":"Organization","captcha":null,"createDate":"2018-10-
25 09:25:23","updateDate":null}}
    Protocol: h2
```

输出 h2 表示当前接口与服务器连接是通过 http2 完成。

2.12. 异步更新 UI

```
                buttonSend.setOnClickListener(new
View.OnClickListener() {
            @Override
            public void onClick(View view) {

                HttpUrl.Builder urlBuilder =
HttpUrl.parse(apiUrl).newBuilder();
```

```
urlBuilder.addPathSegments("search/cache_vector_chatgpt");
                urlBuilder.addQueryParameter("question",
editTextMessage.getText().toString());
                String url = urlBuilder.build().toString();
                Log.d("API", url);
                OkHttpClient client = new OkHttpClient();

                Request request = new
Request.Builder().url(url).build();
                Call call = client.newCall(request);

                call.enqueue(new Callback() {
                    @Override
                    public void onFailure(@NotNull Call call,
@NotNull IOException e) {

                    }

                    @Override
                    public void onResponse(@NotNull Call
call, @NotNull Response response) throws IOException {
                        if (response.isSuccessful()) {
                            answer =
response.body().string();
                            updateResult(answer);
                            Log.i("TAG", "Async: " + answer);
                        }
                    }
                });
            }
        });
```

```
    private void updateResult(final String response) {
        //在子线程中更新UI
        getActivity().runOnUiThread(new Runnable() {
            @Override
            public void run() {
                // 在这里进行UI操作，将结果显示到界面上
                text_dashboard.setText(response);
```

```
            }
        });
    }
```

Lambda 表达式

```
        private void showResponse(final String response) {
        //在子线程中更新UI
        runOnUiThread(() -> {
                        // 在这里进行UI操作，将结果显示到界面上
                        responseText.setText(response);
        });
    }
```

在 Fragment 中需这样使用 getActivity().runOnUiThread()

```
        buttonSend.setOnClickListener(new
View.OnClickListener() {
            @Override
            public void onClick(View view) {

                HttpUrl.Builder urlBuilder =
HttpUrl.parse(apiUrl).newBuilder();

urlBuilder.addPathSegments("search/cache_vector_chatgpt");
                urlBuilder.addQueryParameter("question",
editTextMessage.getText().toString());
                String url = urlBuilder.build().toString();
                Log.d("API", url);
                OkHttpClient client = new OkHttpClient();

                Request request = new
Request.Builder().url(url).build();
                Call call = client.newCall(request);
```

```
                call.enqueue(new Callback() {
                    @Override
                    public void onFailure(@NotNull Call call,
@NotNull IOException e) {

                    }

                    @Override
                    public void onResponse(@NotNull Call
call, @NotNull Response response) throws IOException {
                        if (response.isSuccessful()) {
                            answer =
response.body().string();
                            getActivity().runOnUiThread(() ->
{

text_dashboard.setText(answer);
                            });
                            Log.i("TAG", "Async: " + answer);
                        }
                    }
                });
            }
        });
```

第 11 章 相机与相册

1. manifest 文件

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

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

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

        <provider

android:name="android.support.v4.content.FileProvider"

android:authorities="cn.netkiller.camera.provider"
            android:exported="false"
            android:grantUriPermissions="true">
            <meta-data

android:name="android.support.FILE_PROVIDER_PATHS"
```

```
                 android:resource="@xml/provider_paths" />
        </provider>
    </application>

</manifest>
```

provider_paths.xml 文件

```
<?xml version="1.0" encoding="utf-8"?>
<paths
xmlns:android="http://schemas.android.com/apk/res/android">
    <external-path name="external_files" path="."/>
</paths>
```

2. layout

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.constraint.ConstraintLayout
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"
    tools:context=".MainActivity">

    <TextView
        android:id="@+id/textView"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Hello World!"
        app:layout_constraintLeft_toLeftOf="parent"
        app:layout_constraintRight_toRightOf="parent"
        app:layout_constraintTop_toTopOf="parent" />


    <Button
        android:id="@+id/buttonOpenCamera"
        android:layout_width="160dp"
        android:layout_height="wrap_content"
        android:layout_marginStart="8dp"
        android:layout_marginTop="8dp"
        android:layout_marginEnd="8dp"
        android:layout_marginBottom="8dp"
        android:text="拍照立即显示"

app:layout_constraintBottom_toTopOf="@+id/buttonAlbum"

app:layout_constraintEnd_toStartOf="@+id/buttonSavePhoto"
        app:layout_constraintHorizontal_bias="0.283"
        app:layout_constraintStart_toStartOf="parent"

app:layout_constraintTop_toBottomOf="@+id/imageViewPicture"
        app:layout_constraintVertical_bias="0.0" />
```

```
    <Button
        android:id="@+id/buttonSavePhoto"
        android:layout_width="160dp"
        android:layout_height="wrap_content"
        android:layout_marginTop="8dp"
        android:layout_marginEnd="20dp"
        android:layout_marginBottom="8dp"
        android:text="拍照存储后显示"

app:layout_constraintBottom_toTopOf="@+id/buttonAlbum"
        app:layout_constraintEnd_toEndOf="parent"

app:layout_constraintTop_toBottomOf="@+id/imageViewPicture"
        app:layout_constraintVertical_bias="0.0" />

    <ImageView
        android:id="@+id/imageViewPicture"
        android:layout_width="338dp"
        android:layout_height="366dp"
        android:layout_centerHorizontal="true"
        android:layout_marginStart="8dp"
        android:layout_marginTop="8dp"
        android:layout_marginEnd="8dp"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/textView"
/>

    <Button
        android:id="@+id/buttonAlbum"
        android:layout_width="352dp"
        android:layout_height="wrap_content"
        android:layout_marginStart="8dp"
        android:layout_marginEnd="8dp"
        android:layout_marginBottom="8dp"
        android:text="Album"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent" />


</android.support.constraint.ConstraintLayout>
```

3. Activity

```
package cn.netkiller.camera;

import android.Manifest;
import android.content.ContentResolver;
import android.content.ContentValues;
import android.content.Intent;
import android.content.pm.PackageManager;
import android.database.Cursor;
import android.graphics.Bitmap;
import android.graphics.BitmapFactory;
import android.net.Uri;
import android.os.Environment;
import android.os.StrictMode;
import android.provider.MediaStore;
import android.support.v4.app.ActivityCompat;
import android.support.v4.content.FileProvider;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.util.Log;
import android.view.View;
import android.widget.Button;
import android.widget.ImageView;
import android.widget.TextView;

import java.io.File;
import java.io.FileInputStream;
import java.io.FileNotFoundException;
import java.io.IOException;
import java.io.InputStream;

public class MainActivity extends AppCompatActivity
implements View.OnClickListener {

    private ImageView imageViewPicture;
    private Button buttonOpenCamera;
    private Button buttonSavePhoto;
    private Button buttonAlbum;
```

```
    private static int REQUEST_CAMERA = 1;
    private static int REQUEST_PHOTO = 2;
    private static int REQUEST_ALBUM = 3;
    private File imageFile;

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

        imageViewPicture = (ImageView)
findViewById(R.id.imageViewPicture);
        buttonOpenCamera = (Button)
findViewById(R.id.buttonOpenCamera);
        buttonSavePhoto = (Button)
findViewById(R.id.buttonSavePhoto);
        buttonAlbum = (Button)
findViewById(R.id.buttonAlbum);

        buttonOpenCamera.setOnClickListener(this);
        buttonSavePhoto.setOnClickListener(this);
        buttonAlbum.setOnClickListener(this);

        StrictMode.VmPolicy.Builder newbuilder = new
StrictMode.VmPolicy.Builder();
        StrictMode.setVmPolicy(newbuilder.build());

    }

    @Override
    public void onClick(View view) {
        Intent intent;
        switch (view.getId()) {
            case R.id.buttonOpenCamera:
                // 拍照并显示图片
                intent = new
Intent(MediaStore.ACTION_IMAGE_CAPTURE);// 启动系统相机
                startActivityForResult(intent,
REQUEST_CAMERA);
                break;
            case R.id.buttonSavePhoto:

                if
(!Environment.getExternalStorageState().equals(Environment.ME
DIA_MOUNTED)) {
```

```
                    TextView textView = (TextView)
findViewById(R.id.textView);
                    textView.setText("SD 卡不存在，请插入 SD
卡！");
                }

                if (ActivityCompat.checkSelfPermission(this,
Manifest.permission.WRITE_EXTERNAL_STORAGE) !=
PackageManager.PERMISSION_GRANTED) {

ActivityCompat.requestPermissions(MainActivity.this, new
String[]{Manifest.permission.WRITE_EXTERNAL_STORAGE}, 1);
                    return;
                } else {

                    String dir =
Environment.getExternalStorageDirectory().getPath();
                    new File(dir).mkdirs();

                    imageFile = new File(dir, "tmp.png");

                    Uri imageUri =
FileProvider.getUriForFile(MainActivity.this,
"cn.netkiller.camera.provider", imageFile);
                    Log.d("album", imageFile.getPath());

                    intent = new
Intent(MediaStore.ACTION_IMAGE_CAPTURE);// 启动系统相机
                    intent.putExtra(MediaStore.EXTRA_OUTPUT,
imageUri);
                    startActivityForResult(intent,
REQUEST_PHOTO);


                }

                break;
            case R.id.buttonAlbum:

                intent = new
Intent(Intent.ACTION_GET_CONTENT);
                intent.setType("image/*");

startActivityForResult(Intent.createChooser(intent, "Select
Picture"), 3);
```

```
                break;
            default:
                break;
        }
    }

    @Override
    protected void onActivityResult(int requestCode, int
resultCode, Intent data) {
        super.onActivityResult(requestCode, resultCode,
data);
        if (resultCode == RESULT_OK) { // 如果返回数据
            if (requestCode == REQUEST_CAMERA) { // 判断请求码
是否为REQUEST_CAMERA,如果是代表是这个页面传过去的，需要进行获取
                Bundle bundle = data.getExtras(); // 从data中
取出传递回来缩略图的信息，图片质量差，适合传递小图片
                Bitmap bitmap = (Bitmap) bundle.get("data");
// 将data中的信息流解析为Bitmap类型
                imageViewPicture.setImageBitmap(bitmap);// 显
示图片
            } else if (requestCode == REQUEST_PHOTO) {
                Log.i("photo", imageFile.getPath());
                try {
//                    InputStream inputStream =
getContentResolver().openInputStream(imageUri);
                    String dir =
Environment.getExternalStorageDirectory().getPath();
                    FileInputStream fileInputStream = new
FileInputStream(imageFile);
                    Bitmap bitmap =
BitmapFactory.decodeStream(fileInputStream);
                    fileInputStream.close();

imageViewPicture.setImageBitmap(bitmap);// 显示图片
                } catch (Exception e) {
                    e.printStackTrace();
                }
            } else if (requestCode == REQUEST_ALBUM) {

                Bitmap bitmap = null;

                //外界的程序访问ContentProvider所提供数据 可以通过
ContentResolver接口
                ContentResolver resolver =
```

```
getContentResolver();
                Uri originalUri = data.getData();        //获
得图片的uri

                try {
                    bitmap =
MediaStore.Images.Media.getBitmap(resolver, originalUri);
//显得到bitmap图片
                } catch (IOException e) {
                    e.printStackTrace();
                }

                //获取图片的路径:
                Log.i("album", String.valueOf(originalUri));

                imageViewPicture.setImageBitmap(bitmap);
            }
        }
    }

}
```

4. LED flash 做手电筒

```
<uses-permission android:name="android.permission.FLASHLIGHT"
/>
```

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.constraint.ConstraintLayout
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"
    tools:context=".FlashLightActivity">

    <Button
        android:id="@+id/buttonLight"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginStart="8dp"
        android:layout_marginTop="8dp"
        android:layout_marginEnd="8dp"
        android:layout_marginBottom="8dp"
        android:text="On"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />
</android.support.constraint.ConstraintLayout>
```

```
package cn.netkiller.camera;
```

```
import android.app.AlertDialog;
import android.content.Context;
import android.content.DialogInterface;
import android.content.pm.PackageManager;
import android.hardware.Camera;
import android.hardware.camera2.CameraAccessException;
import android.hardware.camera2.CameraManager;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.Toast;

import java.security.Policy;

public class FlashLightActivity extends AppCompatActivity {

    private Button buttonLight;
    private CameraManager cameraManager;
    private String cameraId;

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

        buttonLight = (Button)
findViewById(R.id.buttonLight);

        Boolean isFlashAvailable =
getApplicationContext().getPackageManager().hasSystemFeature(
PackageManager.FEATURE_CAMERA_FLASH);

        if (!isFlashAvailable) {

            AlertDialog alert = new
AlertDialog.Builder(FlashLightActivity.this).create();
            alert.setTitle("Error");
            alert.setMessage("Your device doesn't support
flash light!");
            alert.setButton(DialogInterface.BUTTON_POSITIVE,
"OK", new DialogInterface.OnClickListener() {
                public void onClick(DialogInterface dialog,
int which) {
```

```
                    // closing the application
                    finish();
                    System.exit(0);
                }
            });
            alert.show();
            return;
        }

        cameraManager = (CameraManager)
getSystemService(Context.CAMERA_SERVICE);
        try {
            cameraId = cameraManager.getCameraIdList()[0];
        } catch (CameraAccessException e) {
            e.printStackTrace();
        }

        buttonLight.setOnClickListener(new
View.OnClickListener() {
            @Override
            public void onClick(View v) {

                try {
                    light();
                } catch (CameraAccessException e) {
                    e.printStackTrace();
                }

            }
        });
    }

    public void light() throws CameraAccessException {


        if (buttonLight.getText().equals("On")) {
            Toast.makeText(getApplicationContext(), "打开手电
筒", Toast.LENGTH_SHORT).show();
            cameraManager.setTorchMode(cameraId, true);
            buttonLight.setText("Off");
        } else {
            Toast.makeText(getApplicationContext(), "关闭手电
筒", Toast.LENGTH_SHORT).show();
            cameraManager.setTorchMode(cameraId, false);
            buttonLight.setText("On");
```

```
            }
        }
}
```

第 12 章 麦克风与录音

1. 开启麦克风和SD卡权限

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

    <uses-permission
android:name="android.permission.RECORD_AUDIO" />
    <uses-permission
android:name="android.permission.WRITE_EXTERNAL_STORAGE" />

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

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

</manifest>
```

2. layout

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.constraint.ConstraintLayout
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"
    tools:context=".MainActivity">

    <TextView
        android:id="@+id/status"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="http://www.netkiller.cn"
        app:layout_constraintLeft_toLeftOf="parent"
        app:layout_constraintRight_toRightOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

    <Button
        android:id="@+id/record"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginStart="8dp"
        android:layout_marginBottom="1dp"
        android:onClick="onClick"
        android:text="Record"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintStart_toStartOf="parent" />

    <Button
        android:id="@+id/play"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginEnd="8dp"
        android:layout_marginBottom="7dp"
        android:text="Play"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent" />
```

```
</android.support.constraint.ConstraintLayout>
```

3. Activity

```
package cn.netkiller.voice;

import android.Manifest;
import android.content.pm.PackageManager;
import android.media.MediaPlayer;
import android.media.MediaRecorder;
import android.os.Environment;
import android.support.v4.app.ActivityCompat;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.util.Log;
import android.view.View;
import android.widget.Button;
import android.widget.TextView;
import android.widget.Toast;

import java.io.IOException;
import java.text.SimpleDateFormat;
import java.util.Date;

public class MainActivity extends AppCompatActivity
implements View.OnClickListener {

    private static final int RECORD_AUDIO = 10;
    private Button record;
    private Button play;
    private MediaRecorder mediaRecorder;
    private TextView status;

    private MediaPlayer mediaPlayer;
    private String filename;

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

        status = (TextView) findViewById(R.id.status);
```

```
        record = (Button) findViewById(R.id.record);
        play = (Button) findViewById(R.id.play);

        record.setOnClickListener(this);
        play.setOnClickListener(this);


        if (ActivityCompat.checkSelfPermission(this,
Manifest.permission.WRITE_EXTERNAL_STORAGE) !=
PackageManager.PERMISSION_GRANTED ||
ActivityCompat.checkSelfPermission(this,
Manifest.permission.RECORD_AUDIO) !=
PackageManager.PERMISSION_GRANTED) {
            ActivityCompat.requestPermissions(this, new
String[]{Manifest.permission.WRITE_EXTERNAL_STORAGE,
Manifest.permission.RECORD_AUDIO}, RECORD_AUDIO);
        }

    }

    @Override
    public void onClick(View v) {

        switch (v.getId()) {
            case R.id.record:
                if (record.getText().equals("Record")) {
                    this.start();
                } else {
                    this.stop();
                }
                break;
            case R.id.play:
                play();
                status.setText("播放录音");
                break;
        }
    }

    private void start() {

        record.setText("Stop");
        status.setText("开启录音，请对准话筒讲话");

        mediaRecorder = new MediaRecorder();
```

```
mediaRecorder.setAudioSource(MediaRecorder.AudioSource.MIC);

mediaRecorder.setOutputFormat(MediaRecorder.OutputFormat.THRE
E_GPP);

mediaRecorder.setAudioEncoder(MediaRecorder.AudioEncoder.AMR_
NB);

        String dir =
Environment.getExternalStorageDirectory().getPath();
        String date = new SimpleDateFormat("yyyy-MM-
ddhhmmss").format(new Date());

        filename = String.format("%s/%s.3gp", dir, date);

        Log.e("Voice", "voice path " + filename);

        mediaRecorder.setOutputFile(filename);

        try {
            mediaRecorder.prepare();
        } catch (IOException e) {
            e.printStackTrace();
        }
        mediaRecorder.start();

    }

    private void stop() {
        record.setText("Record");
        status.setText("录音停止");

        if (mediaRecorder != null) {
            mediaRecorder.stop();
            mediaRecorder.release();
            mediaRecorder = null;
        }
    }

    private void play() {
        this.stop();
        if (filename == null) {
            Toast.makeText(getApplicationContext(), "请先录音",
Toast.LENGTH_SHORT).show();
            return;
```

```
        }
        try {

            if (mediaPlayer != null &&
mediaPlayer.isPlaying()) {
                mediaPlayer.reset();
            } else {
                mediaPlayer = new MediaPlayer();
                mediaPlayer.setDataSource(filename);
                mediaPlayer.prepare();
                mediaPlayer.start();
            }

        } catch (IOException e) {
            e.printStackTrace();
        }
    }

    @Override
    protected void onDestroy() {
        super.onDestroy();
        if (mediaPlayer != null) {
            mediaPlayer.stop();
            mediaPlayer.release();
        }
    }

}
```

第 13 章 多媒体开发

1. MediaPlayer

1.1. 播放Raw下的元数据

播放一段特效声音，例如铃音，可以在点击按钮德时候调用playSound()

```
        private void playSound(){

        MediaPlayer mediaPlayer =
MediaPlayer.create(FlashLightActivity.this, R.raw.alert);
        mediaPlayer.setOnCompletionListener(new
MediaPlayer.OnCompletionListener() {

            @Override
            public void onCompletion(MediaPlayer mp) {
                // TODO Auto-generated method stub
                mediaPlayer.release();
            }
        });
        mediaPlayer.start();
    }
```

播放 res 中的资源文件

```
        AssetFileDescriptor assetFileDescriptor =
context.getResources().openRawResourceFd(R.raw.music);

    public void play(AssetFileDescriptor assetFileDescriptor)
```

```
{
        if (mediaPlayer != null) {
            try {
                this.reset();

mediaPlayer.setDataSource(assetFileDescriptor);
                // 或使用这种方式
mediaPlayer.setDataSource(assetFileDescriptor.getFileDescript
or(), assetFileDescriptor.getStartOffset(),
assetFileDescriptor.getLength());
                mediaPlayer.prepare();
                //注意不能使用异步，异步方式是用来播放网络流音乐
                                //
mediaPlayer.prepareAsync();
                assetFileDescriptor.close();
            } catch (IOException e) {
                throw new RuntimeException(e);
            }

            mediaPlayer.start();
        }
    }
```

播放 assets 文件夹中的资源

```
//实例化播放器
MediaPlayer mediaPlayer = new MediaPlayer();
AssetManager am = getAssets();
try {
    AssetFileDescriptor afd = am.openFd("assets_video.mp4");
    mediaPlayer.setDataSource(afd.getFileDescriptor(),
afd.getStartOffset(), afd.getLength());
} catch (IOException e) {
    e.printStackTrace();
}
//设置准备就绪状态监听
mediaPlayer.setOnPreparedListener(new
MediaPlayer.OnPreparedListener() {
    @Override
```

```
    public void onPrepared(MediaPlayer mp) {
         // 开始播放
         mediaPlayer.start();
     }
});
//准备播放
mediaPlayer.prepareAsync();
```

其他方式获得 assets 文件夹中的资源

```
                String url = "file:///android_asset/" +
"video.mp3";
        AssetFileDescriptor assetFileDescriptor = null;
        try {
            assetFileDescriptor =
context.getResources().getAssets().openFd("nepal.mp3");
        } catch (IOException e) {
            throw new RuntimeException(e);
        }
```

1.2. 播放assets文件夹中的音乐

```
 //需将资源文件放在assets文件夹
 AssetFileDescriptor fd = getAssets().openFd("samsara.mp3");
 mMediaPlayer.setDataSource(fd)
 mMediaPlayer.prepare()

 AssetFileDescriptor fd = getAssets().openFd("samsara.mp3");
 mMediaPlayer.setDataSource(fd, fd.getStartOffset(),
fd.getLegth())
 mMediaPlayer.prepare();
```

1.3. 播放互联网音乐

```
package cn.aigcsst.demo.skill;

import android.media.MediaPlayer;
import android.util.Log;

import com.alibaba.fastjson.JSONArray;
import com.alibaba.fastjson.JSONObject;

import java.io.IOException;

public class Bible {
    private static final String TAG =
Bible.class.getSimpleName();

    public Bible(JSONArray array) {
        Log.i(TAG, array.toString());
        JSONObject item = (JSONObject) array.get(1);
        String url = item.getString("url");
        Log.i(TAG, "Audio url: " + url);
        MediaPlayer mediaPlayer = new MediaPlayer();
        try {
            mediaPlayer.setDataSource(url);//设置播放来源
        } catch (IOException e) {
            throw new RuntimeException(e);
        }
        mediaPlayer.prepareAsync();//异步准备
        mediaPlayer.setOnPreparedListener(new
MediaPlayer.OnPreparedListener() {
            //异步准备监听
            @Override
            public void onPrepared(MediaPlayer mediaPlayer) {

                Log.i(TAG, "Voice异步文件时长: " +
mediaPlayer.getDuration() / 1000 + "");
                mediaPlayer.start();//播放
            }
        });
        mediaPlayer.setScreenOnWhilePlaying(true);// 设置播放的
时候一直让屏幕变亮
```

```
        mediaPlayer.setOnBufferingUpdateListener(new
MediaPlayer.OnBufferingUpdateListener() {
            //文件缓冲监听
            @Override
            public void onBufferingUpdate(MediaPlayer
mediaPlayer, int i) {

                Log.i(TAG, "Voice进度: " + i + "% Voice文件长
度" + mediaPlayer.getDuration() / 1000 + "");
            }
        });
    }
}
```

1.4. 使用单例模式

```
package cn.aigcsst.demo.skill;

import android.media.MediaPlayer;
import android.util.Log;

import androidx.annotation.NonNull;

import com.alibaba.fastjson.JSONArray;
import com.alibaba.fastjson.JSONObject;

import java.io.IOException;

public class Bible {
    private static final String TAG =
Bible.class.getSimpleName();

    private static MediaPlayer mediaPlayer;

    public synchronized static MediaPlayer getInstance() {
        if (mediaPlayer == null)
            mediaPlayer = new MediaPlayer();
        return mediaPlayer;
```

```
    }

    public Bible(@NonNull JSONArray array) {
//        Log.i(TAG, array.toString());
        JSONObject item = (JSONObject) array.get(1);
        String url = item.getString("url");
//        Log.i(TAG, "Audio url: " + item.toString());
        MediaPlayer mediaPlayer = Bible.getInstance();
        try {
            mediaPlayer.reset();
            mediaPlayer.setDataSource(url);//设置播放来源
        } catch (IOException e) {
            throw new RuntimeException(e);
        }
        mediaPlayer.prepareAsync();//异步准备
        mediaPlayer.setOnPreparedListener(new
MediaPlayer.OnPreparedListener() {
            //异步准备监听
            @Override
            public void onPrepared(MediaPlayer mediaPlayer) {
                Log.i(TAG, "Voice 播放异步文件, 时长: " +
mediaPlayer.getDuration() / 1000 + " Audio: " +
item.toString());
                mediaPlayer.start();//播放
            }
        });
        mediaPlayer.setScreenOnWhilePlaying(true);// 设置播放的
时候一直让屏幕变亮
        mediaPlayer.setOnBufferingUpdateListener(new
MediaPlayer.OnBufferingUpdateListener() {
            //文件缓冲监听
            @Override
            public void onBufferingUpdate(MediaPlayer
mediaPlayer, int i) {
                Log.i(TAG, "Voice 缓冲区加载进度: " + i + "%");
            }
        });
    }
}
```

2. VideoView 开发

VideoView，用于播放一段视频媒体，它继承了SurfaceView，位于"android.widget.VideoView"，是一个视频控件。

```
VideoView也为开发人员提供了对应的方法，这里简单介绍一些常用的：

int getCurrentPosition()：获取当前播放的位置。
int getDuration()：获取当前播放视频的总长度。
isPlaying()：当前VideoView是否在播放视频。
void pause()：暂停
void seekTo(int msec)：从第几毫秒开始播放。
void resume()：重新播放。
void setVideoPath(String path)：以文件路径的方式设置VideoView播放的
视频源。
void setVideoURI(Uri uri)：以Uri的方式设置VideoView播放的视频源，可
以是网络Uri或本地Uri。
void start()：开始播放。
void stopPlayback()：停止播放。
setMediaController(MediaController controller)：设置
MediaController控制器。
setOnCompletionListener(MediaPlayer.onCompletionListener l)：监
听播放完成的事件。
setOnErrorListener(MediaPlayer.OnErrorListener l)：监听播放发生错
误时候的事件。
setOnPreparedListener(MediaPlayer.OnPreparedListener l)：：监听视
频装载完成的事件。
```

上面的一些方法通过方法名就可以了解用途。和MediaPlayer配合SurfaceView播放视频不同，VideoView播放之前无需编码装载视频，它会在start()开始播放的时候自动装载视频。并且VideoView在使用完之后，无需编码回收资源。

2.1. 播放网络视频

加入 android.permission.INTERNET 允许访问网络

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

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

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

</manifest>
```

最简洁的布局，只有一个 VideoView

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.constraint.ConstraintLayout
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"
    tools:context=".VideoViewActivity">

    <VideoView
        android:id="@+id/videoView"
        android:layout_width="match_parent"
        android:layout_height="236dp"
        android:layout_marginStart="8dp"
        android:layout_marginTop="8dp"
        android:layout_marginEnd="8dp"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

</android.support.constraint.ConstraintLayout>
```

播放的文件来自 IPFS 星际文件系统

```
package cn.netkiller.video;

import android.net.Uri;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.widget.MediaController;
import android.widget.VideoView;

public class VideoViewActivity extends AppCompatActivity {

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

        Uri uri =
Uri.parse("http://ipfs.netkiller.cn/ipfs/QmcA1Fsrt6jGTVqAUNZB
qaprMEdFaFkmkzA5s2M6mF85UC");
        VideoView videoView = (VideoView)
this.findViewById(R.id.videoView);
        videoView.setMediaController(new
```

```
MediaController(this));
        videoView.setVideoURI(uri);
        videoView.start();
        videoView.requestFocus();

    }
}
```

运行程序开始播放视频，点击视频会在屏幕下方弹出 MediaController 控制条

2.2. MediaController 添加翻页事件

```
package cn.netkiller.video;

import android.net.Uri;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.View;
import android.widget.MediaController;
import android.widget.Toast;
import android.widget.VideoView;

public class VideoViewActivity extends AppCompatActivity {

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

        final Uri uri =
Uri.parse("http://ipfs.netkiller.cn/ipfs/QmcA1Fsrt6jGTVqAUNZB
qaprMEdFaFkmkzA5s2M6mF85UC");
        final VideoView videoView = (VideoView)
this.findViewById(R.id.videoView);
        MediaController mediaController = new
MediaController(this);
        mediaController.setMediaPlayer(videoView);
```

```
        mediaController.setPrevNextListeners(
                new View.OnClickListener() {

                    @Override
                    public void onClick(View v) {

Toast.makeText(VideoViewActivity.this, "下一曲",
Toast.LENGTH_SHORT).show();

videoView.setVideoURI(Uri.parse("http://ipfs.netkiller.cn/ipf
s/QmUaDftnPB7zCTwTASnSAWLiXWd1L5vNGEeU585rddfVTh"));
                    }
                },
                new View.OnClickListener() {

                    @Override
                    public void onClick(View v) {

Toast.makeText(VideoViewActivity.this, "上一曲",
Toast.LENGTH_SHORT).show();

videoView.setVideoURI(Uri.parse("http://ipfs.netkiller.cn/ipf
s/QmbvKvj9X368WMtmkLYFuf59gSwLXYDLcdJuSiSHKPhTG4"));
                    }
                });

        videoView.setMediaController(mediaController);
        videoView.setVideoURI(uri);
        videoView.start();
        videoView.requestFocus();

    }
}
```

2.3. 静音播放视频

```
                videoView.setOnPreparedListener(new
MediaPlayer.OnPreparedListener() {
```

```
            @Override
            public void onPrepared(MediaPlayer mediaPlayer) {
                mediaPlayer.setVolume(0f, 0f);
            }
        });
```

2.4. 更新进度条

```
            new Thread() {
                @Override
                public void run() {
                    try {
                        while (videoView.isPlaying()) {
                            // 如果正在播放，没0.5.毫秒更新一次进度
条
                            int current =
videoView.getCurrentPosition();
                            seekBar.setProgress(current);
                            sleep(500);
                        }
                    } catch (Exception e) {
                        e.printStackTrace();
                    }
                }
            }.start();
```

2.5. 完整的例子

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.constraint.ConstraintLayout
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"
    tools:context=".VideoViewActivity">

    <VideoView
        android:id="@+id/videoView"
        android:layout_width="match_parent"
        android:layout_height="236dp"
        android:layout_marginStart="8dp"
        android:layout_marginTop="8dp"
        android:layout_marginEnd="8dp"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

    <LinearLayout
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_marginStart="8dp"
        android:layout_marginEnd="8dp"
        android:layout_marginBottom="8dp"
        android:orientation="vertical"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent">


        <LinearLayout
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:orientation="horizontal">

            <TextView
                android:id="@+id/textViewTime"
                android:layout_width="wrap_content"
                android:layout_height="match_parent"
                android:layout_weight="1"
                android:text="00:00" />

            <SeekBar
                android:id="@+id/seekBar"
                android:layout_width="270dp"
                android:layout_height="wrap_content" />
```

```
            <TextView
                android:id="@+id/textViewCurrentPosition"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:layout_weight="1"
                android:text="00:00" />
        </LinearLayout>

        <LinearLayout
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:orientation="horizontal">

            <Button
                android:id="@+id/buttonPlay"
                android:layout_width="183dp"
                android:layout_height="wrap_content"
                android:text="播放" />

            <Button
                android:id="@+id/buttonStop"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:layout_weight="1"
                android:text="停止" />

        </LinearLayout>

    </LinearLayout>

    <TextView
        android:id="@+id/textViewStatus"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginStart="8dp"
        android:layout_marginTop="8dp"
        android:layout_marginEnd="8dp"
        android:text="        "
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/videoView"
/>
</android.support.constraint.ConstraintLayout>
```

```
package cn.netkiller.video;

import android.media.MediaPlayer;
import android.net.Uri;
import android.os.Handler;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.SeekBar;
import android.widget.TextView;
import android.widget.Toast;
import android.widget.VideoView;

import java.text.SimpleDateFormat;
import java.util.Calendar;

public class VideoViewActivity extends AppCompatActivity
implements View.OnClickListener {

    private VideoView videoView;
    private SeekBar seekBar;
    private Button buttonPlay;
    private TextView textViewTime;
    private TextView textViewCurrentPosition;
    private TextView textViewStatus;

    private Handler handler = new Handler();
    private Runnable runnable = new Runnable() {
        public void run() {
            if (videoView.isPlaying()) {
                int current = videoView.getCurrentPosition();
                seekBar.setProgress(current);

textViewCurrentPosition.setText(time(videoView.getCurrentPosi
tion()));
            }
            handler.postDelayed(runnable, 500);
        }
```

```
    };

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

        final Uri uri =
Uri.parse("http://ipfs.netkiller.cn/ipfs/QmcA1Fsrt6jGTVqAUNZB
qaprMEdFaFkmkzA5s2M6mF85UC");
        videoView = (VideoView)
this.findViewById(R.id.videoView);
        videoView.setVideoURI(uri);
        videoView.requestFocus();

        videoView.setOnPreparedListener(new
MediaPlayer.OnPreparedListener() {

            @Override
            public void onPrepared(MediaPlayer mediaPlayer) {

textViewTime.setText(time(videoView.getDuration()));
                textViewStatus.setText("视频加载完毕");
                buttonPlay.setEnabled(true);

            }
        });

        // 在播放完毕被回调
        videoView.setOnCompletionListener(new
MediaPlayer.OnCompletionListener() {
            @Override
            public void onCompletion(MediaPlayer mp) {
                Toast.makeText(VideoViewActivity.this, "播放完
成", Toast.LENGTH_SHORT).show();
            }
        });

        videoView.setOnErrorListener(new
MediaPlayer.OnErrorListener() {

            @Override
            public boolean onError(MediaPlayer mp, int what,
int extra) {
                // 发生错误重新播放
```

```
                play();
                Toast.makeText(VideoViewActivity.this, "播放出
错", Toast.LENGTH_SHORT).show();
                return false;
            }
        });


        textViewStatus = (TextView)
findViewById(R.id.textViewStatus);
        textViewStatus.setText("玩命加载中");

        textViewTime = (TextView)
findViewById(R.id.textViewTime);

        seekBar = (SeekBar) findViewById(R.id.seekBar);
        // 为进度条添加进度更改事件

seekBar.setOnSeekBarChangeListener(onSeekBarChangeListener);

        textViewCurrentPosition = (TextView)
findViewById(R.id.textViewCurrentPosition);

        buttonPlay = (Button) findViewById(R.id.buttonPlay);
        buttonPlay.setEnabled(false);
        final Button buttonStop = (Button)
findViewById(R.id.buttonStop);

        buttonPlay.setOnClickListener(this);
        buttonStop.setOnClickListener(this);

    }

    @Override
    public void onClick(View v) {
        switch (v.getId()) {
            case R.id.buttonPlay:
                play();
                break;
            case R.id.buttonStop:
                stop();
                break;
            default:
                break;
        }
```

```
    }

    private SeekBar.OnSeekBarChangeListener
onSeekBarChangeListener = new
SeekBar.OnSeekBarChangeListener() {
        // 当进度条停止修改的时候触发
        @Override
        public void onStopTrackingTouch(SeekBar seekBar) {
            // 取得当前进度条的刻度
            int progress = seekBar.getProgress();
            if (videoView.isPlaying()) {
                // 设置当前播放的位置
                videoView.seekTo(progress);
            }
        }

        @Override
        public void onStartTrackingTouch(SeekBar seekBar) {

        }

        @Override
        public void onProgressChanged(SeekBar seekBar, int
progress,
                                      boolean fromUser) {

        }
    };

    protected void play() {

        if (buttonPlay.getText().equals("播放")) {
            buttonPlay.setText("暂停");
            textViewStatus.setText("请您欣赏");
            // 开始线程，更新进度条的刻度
            handler.postDelayed(runnable, 0);
            videoView.start();
            seekBar.setMax(videoView.getDuration());

        } else

        {
            buttonPlay.setText("播放");
            if (videoView.isPlaying()) {
                videoView.pause();
```

```
            }
        }

    }

    protected void stop() {
        if (videoView.isPlaying()) {
            videoView.stopPlayback();
        }
    }

    protected String time(long millionSeconds) {

        SimpleDateFormat simpleDateFormat = new
SimpleDateFormat("mm:ss");
        Calendar c = Calendar.getInstance();
        c.setTimeInMillis(millionSeconds);
        return simpleDateFormat.format(c.getTime());
    }

    @Override
    protected void onDestroy() {
        super.onDestroy();
        handler.removeCallbacks(runnable);
    }

}
```

2.6. 循环播放

```
        String uri = "android.resource://" + getPackageName()
+ "/" + R.raw.vfo1;
        VideoView videoView = (VideoView)
this.findViewById(R.id.videoView);
        videoView.setMediaController(new
MediaController(this));
        videoView.setVideoURI(Uri.parse(uri));
        videoView.start();
        videoView.requestFocus();
```

```
        videoView.setOnCompletionListener(new
MediaPlayer.OnCompletionListener() {
            @Override
            public void onCompletion(MediaPlayer mp) {
                Log.d(TAG, "我播完了");
                // 下一个视频
//                  videoView.setVideoURI(Uri.parse(uri));
                videoView.seekTo(10);
                videoView.start();
            }
        });
```

2.7. 静音播放

静音播放，只需设置音量位 0，使用这个方法 mediaPlayer.setVolume(0f, 0f);

```
        String uri = "android.resource://" + getPackageName()
+ "/" + R.raw.vfo1;
        VideoView videoView = (VideoView)
this.findViewById(R.id.videoView);
        videoView.setMediaController(new
MediaController(this));
        videoView.setVideoURI(Uri.parse(uri));
        videoView.start();
        videoView.requestFocus();
        videoView.setOnCompletionListener(new
MediaPlayer.OnCompletionListener() {
            @Override
            public void onCompletion(MediaPlayer mp) {
                videoView.seekTo(2);
                videoView.start();
            }
        });
        videoView.setOnPreparedListener(new
MediaPlayer.OnPreparedListener() {
            @Override
            public void onPrepared(MediaPlayer mediaPlayer) {
```

```
                mediaPlayer.setVolume(0f, 0f);
            }
        });
```

3. SoundPool

```
package cn.netkiller.sound;

import android.content.res.AssetManager;
import android.media.AudioManager;
import android.media.SoundPool;
import android.util.Log;

import cn.aigcsst.demo.ContextHolder;

import java.io.IOException;
import java.util.HashMap;


public class SoundPoolUtil {
    private static final String TAG =
SoundPoolUtil.class.getSimpleName();
    private static SoundPool soundPool;
    private static HashMap<String, Integer> soundPoolMap =
new HashMap();

    public static void create() {
        if (soundPool != null) {
            return;
        }

        int maxStreams = 3;
        soundPool = new
SoundPool.Builder().setMaxStreams(maxStreams).build();
        //加载音频到内存
        try {
            AssetManager am =
ContextHolder.getContext().getAssets();
            soundPoolMap.put("唤醒提示音",
soundPool.load(am.openFd("audio/wakeResponse.mp3"), 1));
            soundPoolMap.put("未联网提示音",
soundPool.load(am.openFd("audio/offline.mp3"), 1));
            soundPoolMap.put("在",
soundPool.load(am.openFd("audio/zai.wav"), 1));
```

```
        } catch (IOException e) {
            e.printStackTrace();
        }
        //资源加载完成回调
        soundPool.setOnLoadCompleteListener((soundPool,
sampleId, status) -> Log.i(TAG, "音频加载完毕, id=" +
sampleId));
    }

    public static void release() {
        if (soundPool != null) {
            soundPool.release();
            soundPool = null;
        }
    }

    public static void play(String audioName) {
        if (soundPool == null) {
            Log.e(TAG, "soundPool未初始化");
            return;
        }
        /**
         * 参数1：加载返回的声音Id
         * leftVolume：左声道音量, 0.0-1.0f
         * rightVolume：右声道音量, 0.0-1.0f
         * priority：优先级
         * loop：循环播放：  0(不循环)   -1(循环)
         * rate：播放速率  0.5--2.0f
         */
        soundPool.play(soundPoolMap.get(audioName), 1.0f,
1.0f, 1, 0, 1.0f);
    }

}
```

4. 音量控制

```
    private void volume(String control) {
        AudioManager audioManager = (AudioManager)
context.getSystemService(Context.AUDIO_SERVICE);
        int maxVolume =
audioManager.getStreamMaxVolume(AudioManager.STREAM_MUSIC);
        int minVolume = 10;
        int stepVolume = 5;
        int currentMusicVolume =
audioManager.getStreamVolume(AudioManager.STREAM_MUSIC);
        int currentTTSVolume =
audioManager.getStreamVolume(AudioManager.STREAM_ALARM);


        switch (control) {
            case "VOLUME_MINUS": //步进减小
                currentMusicVolume -= stepVolume;
                if (currentMusicVolume < minVolume) {
                    currentMusicVolume = minVolume;
                }
                currentTTSVolume -= stepVolume;
                if (currentTTSVolume < minVolume) {
                    currentTTSVolume = minVolume;
                }
                break;
            case "VOLUME_PLUS": //步进累加
                currentMusicVolume += stepVolume;
                if (currentMusicVolume >= maxVolume) {
                    currentMusicVolume = maxVolume;
                }
                currentTTSVolume += stepVolume;
                if (currentTTSVolume > maxVolume) {
                    currentTTSVolume = maxVolume;
                }
                break;

            case "VOLUME_MAX": // 最大
                currentMusicVolume = currentTTSVolume =
maxVolume;
```

```
                break;
            case "VOLUME_MIN": //最小
                currentMusicVolume = currentTTSVolume =
minVolume;

                break;
            case "MUTE": //静音
                currentMusicVolume = currentTTSVolume =
minVolume;
                break;

        }

audioManager.setStreamVolume(AudioManager.STREAM_MUSIC,
currentMusicVolume, AudioManager.FLAG_SHOW_UI);

audioManager.setStreamVolume(AudioManager.STREAM_ALARM,
currentTTSVolume, AudioManager.FLAG_PLAY_SOUND);
        Log.d(TAG, String.format("volume:
currentMusicVolume=%s, currentTTSVolume=%s, maxVolume=%s",
currentMusicVolume, currentTTSVolume, maxVolume));
    }

    private void volume(double percent) {
        if (percent < 0.3) {
            return;
        }
        AudioManager audioManager = (AudioManager)
context.getSystemService(Context.AUDIO_SERVICE);
        int maxVolume =
audioManager.getStreamMaxVolume(AudioManager.STREAM_MUSIC);
        int currentMusicVolume, currentTTSVolume;
        currentMusicVolume = currentTTSVolume = (int)
(maxVolume * percent);

audioManager.setStreamVolume(AudioManager.STREAM_MUSIC,
currentMusicVolume, AudioManager.FLAG_SHOW_UI);

audioManager.setStreamVolume(AudioManager.STREAM_ALARM,
currentTTSVolume, AudioManager.FLAG_PLAY_SOUND);
        Log.d(TAG, String.format("volume:
currentMusicVolume=%s, currentTTSVolume=%s, maxVolume=%s",
currentMusicVolume, currentTTSVolume, maxVolume));
    }
```

5. SurfaceView

6. Vitamio

第 14 章 定位

1. GPS + 网络 定位

1.1. manifest 权限配置

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

    <uses-permission
android:name="android.permission.ACCESS_COARSE_LOCATION" />
    <uses-permission
android:name="android.permission.ACCESS_FINE_LOCATION" />
    <uses-permission
android:name="android.permission.ACCESS_WIFI_STATE" />
    <uses-permission
android:name="android.permission.ACCESS_NETWORK_STATE" />
    <uses-permission
android:name="android.permission.CHANGE_WIFI_STATE" />
    <uses-permission
android:name="android.permission.INTERNET" />

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

```
            </intent-filter>
        </activity>
    </application>

</manifest>
```

1.2. layout

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.constraint.ConstraintLayout
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"
    tools:context=".MainActivity">

    <TableLayout
        android:id="@+id/tableLayout"
        android:layout_width="match_parent"
        android:layout_height="0dp"
        android:layout_marginStart="8dp"
        android:layout_marginTop="8dp"
        android:layout_marginEnd="8dp"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent">

        <TableRow
            android:layout_width="match_parent"
            android:layout_height="match_parent">

            <TextView
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:text="状态: " />

            <TextView
                android:id="@+id/status"
```

```
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:text="TextView" />
    </TableRow>

    <TableRow
        android:layout_width="match_parent"
        android:layout_height="match_parent">

        <TextView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="经度: " />

        <TextView
            android:id="@+id/textViewLatitude"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:text="TextView" />
    </TableRow>

    <TableRow
        android:layout_width="match_parent"
        android:layout_height="match_parent">

        <TextView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="纬度: " />

        <TextView
            android:id="@+id/textViewLongitude"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:text="TextView" />
    </TableRow>

    <TableRow
        android:layout_width="match_parent"
        android:layout_height="match_parent">

        <TextView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="海拔: " />
```

```
            <TextView
                android:id="@+id/textViewAltitude"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:text="TextView" />
        </TableRow>

        <TableRow
            android:layout_width="match_parent"
            android:layout_height="match_parent">

            <TextView
                android:id="@+id/textView4"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:text="速度" />

            <TextView
                android:id="@+id/textViewSpeed"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:text="TextView" />
        </TableRow>

        <TableRow
            android:layout_width="match_parent"
            android:layout_height="match_parent">

            <TextView
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:text="时间: " />

            <TextView
                android:id="@+id/textViewTime"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:text="TextView" />
        </TableRow>
    </TableLayout>

    <ListView
        android:id="@+id/listViewAddress"
        android:layout_width="368dp"
```

```
        android:layout_height="358dp"
        android:layout_marginStart="8dp"
        android:layout_marginTop="8dp"
        android:layout_marginEnd="8dp"
        android:layout_marginBottom="8dp"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"

app:layout_constraintTop_toBottomOf="@+id/tableLayout" />
</android.support.constraint.ConstraintLayout>
```

1.3. Activity

```
package cn.netkiller.location;

import android.Manifest;
import android.content.pm.PackageManager;
import android.location.Address;
import android.location.Criteria;
import android.location.Geocoder;
import android.location.Location;
import android.location.LocationListener;
import android.location.LocationManager;
import android.location.LocationProvider;
import android.support.v4.app.ActivityCompat;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.util.Log;
import android.view.View;
import android.widget.ArrayAdapter;
import android.widget.ListView;
import android.widget.TextView;
import android.widget.Toast;

import java.io.IOException;
import java.text.SimpleDateFormat;
import java.util.ArrayList;
import java.util.Date;
```

```
import java.util.List;
import java.util.Locale;

public class MainActivity extends AppCompatActivity {

    private static final int REQUEST_PERMISSION_CODE = 12;
    private TextView textViewLatitude;
    private TextView textViewLongitude;
    private TextView textViewAltitude;
    private TextView textViewSpeed;
    private TextView textViewTime;
    private TextView status;

    private static final SimpleDateFormat dateFormat = new
SimpleDateFormat("yyyy-MM-dd HH:mm:ss");
    private ListView listViewAddress;
    private ArrayAdapter<String> adapter;
    private ArrayList<String> list;

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

        textViewLatitude = (TextView)
findViewById(R.id.textViewLatitude);
        textViewLongitude = (TextView)
findViewById(R.id.textViewLongitude);
        textViewAltitude = (TextView)
findViewById(R.id.textViewAltitude);
        textViewSpeed = (TextView)
findViewById(R.id.textViewSpeed);
        textViewTime = (TextView)
findViewById(R.id.textViewTime);
        status = (TextView) findViewById(R.id.status);

        list = new ArrayList<String>();
        adapter = new ArrayAdapter<String>(MainActivity.this,
android.R.layout.simple_list_item_1, list);
        listViewAddress = (ListView)
findViewById(R.id.listViewAddress);
        listViewAddress.setAdapter(adapter);

        this.location();
    }
```

```
    private void loop() {

    }

    public void location() {

        //获取LocationManager对象
        LocationManager locationManager = (LocationManager)
getSystemService(LOCATION_SERVICE);

        boolean gpsStatus =
locationManager.isProviderEnabled(LocationManager.GPS_PROVIDE
R);
        Log.d("Location", "GPS Status: " + gpsStatus);

        boolean networkStatus =
locationManager.isProviderEnabled(LocationManager.NETWORK_PRO
VIDER);
        Log.d("Location", "Network Status: " +
networkStatus);

        //创建一个Criteria对象
        Criteria criteria = new Criteria();
        //设置粗略精确度
        criteria.setAccuracy(Criteria.ACCURACY_COARSE);
        //设置是否需要返回海拔信息
        criteria.setAltitudeRequired(true);
        //设置是否需要返回方位信息
        criteria.setBearingRequired(true);
        //设置是否允许付费服务
        criteria.setCostAllowed(false);
        //设置电量消耗等级
        criteria.setPowerRequirement(Criteria.POWER_HIGH);
        //设置是否需要返回速度信息
        criteria.setSpeedRequired(true);

        Log.d("Location", "Criteria: " +
criteria.toString());

        //获取最符合此标准的provider对象
//        String currentProvider =
locationManager.getProvider(LocationManager.GPS_PROVIDER).get
Name();
```

```
        //根据设置的Criteria对象，获取最符合此标准的provider对象
        String currentProvider =
locationManager.getBestProvider(criteria, true);

        Log.d("Location", "currentProvider: " +
currentProvider);
        status.setText(currentProvider);


        if (ActivityCompat.checkSelfPermission(this,
Manifest.permission.ACCESS_FINE_LOCATION) !=
PackageManager.PERMISSION_GRANTED &&
ActivityCompat.checkSelfPermission(this,
Manifest.permission.ACCESS_COARSE_LOCATION) !=
PackageManager.PERMISSION_GRANTED) {

ActivityCompat.requestPermissions(MainActivity.this, new
String[]{Manifest.permission.ACCESS_FINE_LOCATION,
Manifest.permission.ACCESS_COARSE_LOCATION},
REQUEST_PERMISSION_CODE);
            return;
        } else {
            status.setText("正在获取GPS坐标请稍候...");
        }


locationManager.requestLocationUpdates(currentProvider, 0, 0,
locationListener);
        //根据当前provider对象获取最后一次位置信息
        Location location =
locationManager.getLastKnownLocation(currentProvider);

        //如果位置信息不为null，则请求更新位置信息
        if (location != null) {

            textViewLatitude.setText(location.getLatitude() +
"");
            textViewLongitude.setText(location.getLongitude()
+ "");
            textViewAltitude.setText(location.getAltitude() +
"");
            textViewSpeed.setText(location.getSpeed() + "");
            textViewTime.setText(location.getTime() + "");
```

```
            Log.d("Location", "Latitude: " +
location.getLatitude());
            Log.d("Location", "location: " +
location.getLongitude());

        } else {

            Log.d("Location", "Latitude: " + 0);
            Log.d("Location", "location: " + 0);

        }

    }

    //创建位置监听器
    private LocationListener locationListener = new
LocationListener() {

        //位置发生改变时调用
        @Override
        public void onLocationChanged(Location location) {
            status.setText("onLocationChanged");

            //位置信息变化时触发
            Log.e("Location", "定位方式: " +
location.getProvider());
            Log.e("Location", "纬度: " +
location.getLatitude());
            Log.e("Location", "经度: " +
location.getLongitude());
            Log.e("Location", "海拔: " +
location.getAltitude());
            Log.e("Location", "时间: " + location.getTime());


            textViewLatitude.setText(location.getLatitude() +
"");
            textViewLongitude.setText(location.getLongitude()
+ "");
            textViewAltitude.setText(location.getAltitude() +
"");
            textViewSpeed.setText(location.getSpeed() + "");
            textViewTime.setText(dateFormat.format(new
Date(location.getTime())) + "");
```

```
            //解析地址
            Geocoder geoCoder = new
Geocoder(MainActivity.this, Locale.getDefault());

            double latitude = location.getLatitude();
            double longitude = location.getLongitude();

            List<Address> locationList = null;
            try {
                locationList =
geoCoder.getFromLocation(latitude, longitude, 5);
            } catch (IOException e) {
                e.printStackTrace();
            }

//            Address address = locationList.get(0);//得到
Address实例第一个地址
//            status.setText(address.toString());
//            String countryName =
address.getCountryName();//得到国家名称，比如：中国
//            String locality = address.getLocality();//得到城
市名称，比如：北京市

            list.clear();

            for (Address address : locationList) {

                for (int n = 0; address.getAddressLine(n) !=
null; n++) {
                    String addressLine =
address.getAddressLine(n);//得到周边信息，包括街道等，i=0，得到街道
名称
                    list.add(addressLine);
                    Log.i("Location", "addressLine = " +
addressLine);
                    Log.d("Location",
address.getCountryName() + address.getAdminArea() +
address.getFeatureName());
                }
            }

            adapter.notifyDataSetChanged();

        }
```

```
        //provider失效时调用
        @Override
        public void onProviderDisabled(String provider) {

            Log.d("Location", "onProviderDisabled");
            status.setText("onProviderDisabled");

        }

        //provider启用时调用
        @Override
        public void onProviderEnabled(String provider) {

            Log.d("Location", "onProviderEnabled");
            status.setText("onProviderEnabled");

        }

        //状态改变时调用
        @Override
        public void onStatusChanged(String provider, int
status, Bundle extras) {

            Log.d("Location", "onStatusChanged");
            //GPS状态变化时触发
            switch (status) {
                case LocationProvider.AVAILABLE:
                    Log.e("Location", "当前GPS状态为可见状态");
                    break;
                case LocationProvider.OUT_OF_SERVICE:
                    Log.e("Location", "当前GPS状态为服务区外状
态");
                    break;
                case
LocationProvider.TEMPORARILY_UNAVAILABLE:
                    Log.e("Location", "当前GPS状态为暂停服务状
态");
                    break;
            }

        }

    };
```

```
    @Override
    public void onRequestPermissionsResult(int requestCode,
String[] permissions, int[] grantResults) {
        super.onRequestPermissionsResult(requestCode,
permissions, grantResults);

        switch (requestCode) {
            case REQUEST_PERMISSION_CODE: {
                // If request is cancelled, the result arrays
are empty.
                if (grantResults.length > 0 &&
grantResults[0] == PackageManager.PERMISSION_GRANTED) {

                } else {
                    // permission denied, boo! Disable the
                    // functionality that depends on this
permission.
                }
                return;
            }

        }
    }


}
```

2. 只从 GPS 获取定位

默认安卓系统使用 GPS + 网络定位，网络定位速度非常快，GPS 需要一些搜星。但是网络定位没有海拔高度数据，所以有些场景需要 GPS 定位。

```
package cn.netkiller.ropeway;

import android.Manifest;
import android.annotation.SuppressLint;
import android.content.pm.PackageManager;
import android.location.Address;
import android.location.Criteria;
import android.location.Geocoder;
import android.location.Location;
import android.location.LocationListener;
import android.location.LocationManager;
import android.location.LocationProvider;
import android.os.Bundle;
import android.util.Log;
import android.widget.ArrayAdapter;
import android.widget.ListView;
import android.widget.TextView;

import
com.google.android.material.bottomnavigation.BottomNavigation
View;

import androidx.appcompat.app.AppCompatActivity;
import androidx.core.app.ActivityCompat;
import androidx.navigation.NavController;
import androidx.navigation.Navigation;
import androidx.navigation.ui.AppBarConfiguration;
import androidx.navigation.ui.NavigationUI;

import java.io.IOException;
import java.text.SimpleDateFormat;
import java.util.ArrayList;
import java.util.Date;
```

```
import java.util.List;
import java.util.Locale;

import cn.netkiller.ropeway.databinding.ActivityMainBinding;

public class MainActivity extends AppCompatActivity {

    private ActivityMainBinding binding;

    private static final int REQUEST_PERMISSION_CODE = 12;
    private TextView textViewLatitude;
    private TextView textViewLongitude;
    private TextView textViewAltitude;
    private TextView textViewSpeed;
    private TextView textViewTime;
    private TextView status;

    private final ArrayList<String> loglist = new
ArrayList<String>();
    private ArrayAdapter<String> loggerArrayAdapter;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);

        binding =
ActivityMainBinding.inflate(getLayoutInflater());
        setContentView(binding.getRoot());

        BottomNavigationView navView =
findViewById(R.id.nav_view);
        // Passing each menu ID as a set of Ids because each
        // menu should be considered as top level
destinations.
        AppBarConfiguration appBarConfiguration = new
AppBarConfiguration.Builder(R.id.navigation_home,
R.id.navigation_dashboard,
R.id.navigation_notifications).build();
        NavController navController =
Navigation.findNavController(this,
R.id.nav_host_fragment_activity_main);
        NavigationUI.setupActionBarWithNavController(this,
navController, appBarConfiguration);
        NavigationUI.setupWithNavController(binding.navView,
navController);
```

```
        textViewLatitude =
findViewById(R.id.textViewLatitude);
        textViewLongitude =
findViewById(R.id.textViewLongitude);
        textViewAltitude =
findViewById(R.id.textViewAltitude);
        textViewSpeed = findViewById(R.id.textViewSpeed);
        textViewTime = findViewById(R.id.textViewTime);
        status = findViewById(R.id.status);

        ListView listViewLogger =
findViewById(R.id.listViewLogger);
        loggerArrayAdapter = new ArrayAdapter<String>
(MainActivity.this, android.R.layout.simple_list_item_1,
loglist);
        listViewLogger.setAdapter(loggerArrayAdapter);


        this.location();
    }

    @SuppressLint("SetTextI18n")
    public void location() {

        if (ActivityCompat.checkSelfPermission(this,
Manifest.permission.ACCESS_FINE_LOCATION) !=
PackageManager.PERMISSION_GRANTED &&
ActivityCompat.checkSelfPermission(this,
Manifest.permission.ACCESS_COARSE_LOCATION) !=
PackageManager.PERMISSION_GRANTED) {

ActivityCompat.requestPermissions(MainActivity.this, new
String[]{Manifest.permission.ACCESS_FINE_LOCATION,
Manifest.permission.ACCESS_COARSE_LOCATION},
REQUEST_PERMISSION_CODE);
            return;
        } else {
            status.setText("正在获取GPS坐标请稍候...");
        }

        //获取LocationManager对象
        LocationManager locationManager = (LocationManager)
getSystemService(LOCATION_SERVICE);
```

```
        boolean gpsStatus =
locationManager.isProviderEnabled(LocationManager.GPS_PROVIDE
R);
//
locationManager.setTestProviderEnabled(LocationManager.NETWOR
K_PROVIDER,false);
        Log.d("Location", "GPS Status: " + gpsStatus);


        boolean networkStatus =
locationManager.isProviderEnabled(LocationManager.NETWORK_PRO
VIDER);
        Log.d("Location", "Network Status: " +
networkStatus);


locationManager.requestLocationUpdates(LocationManager.GPS_PR
OVIDER, 1000, 0, locationListener);
        //根据当前provider对象获取最后一次位置信息
        Location location =
locationManager.getLastKnownLocation(LocationManager.GPS_PROV
IDER);

        loglist.add(String.format("GPS Status: %s, Network
Status: %s, Criteria: %s", gpsStatus, networkStatus,
criteria));

        //如果位置信息不为null，则请求更新位置信息
        if (location != null) {

            textViewLatitude.setText(location.getLatitude() +
"");
            textViewLongitude.setText(location.getLongitude()
+ "");
            textViewAltitude.setText(location.getAltitude() +
"");
            textViewSpeed.setText(location.getSpeed() + "");
            textViewTime.setText(location.getTime() + "");

            Log.d("Location", "Latitude: " +
location.getLatitude());
            Log.d("Location", "Location: " +
location.getLongitude());
            Log.d("Location", "Altitude: " +
location.getAltitude());
```

```
            loglist.add(String.format("Provider: %s,
Latitude: %s, Location: %s, Altitude: %s",
location.getProvider(), location.getLatitude(),
location.getLongitude(), location.getAltitude()));
        } else {

            Log.d("Location", "Latitude: " + 0);
            Log.d("Location", "location: " + 0);

        }

    }

    //创建位置监听器
    private final LocationListener locationListener = new
LocationListener() {

        //位置发生改变时调用
        @SuppressLint("SetTextI18n")
        @Override
        public void onLocationChanged(Location location) {
            status.setText("onLocationChanged");

            //位置信息变化时触发
            Log.e("Location", "定位方式: " +
location.getProvider());
            Log.e("Location", "纬度: " +
location.getLatitude());
            Log.e("Location", "经度: " +
location.getLongitude());
            Log.e("Location", "海拔: " +
location.getAltitude());
            Log.e("Location", "时间: " + location.getTime());

            SimpleDateFormat simpleDateFormat = new
SimpleDateFormat("yyyy-MM-dd HH:mm:ss");

            textViewLatitude.setText(location.getLatitude() +
"");
            textViewLongitude.setText(location.getLongitude()
+ "");
            textViewAltitude.setText(location.getAltitude() +
"");
            textViewSpeed.setText(location.getSpeed() + "");
            textViewTime.setText(simpleDateFormat.format(new
```

```
Date(location.getTime())) + "");

            loglist.add(String.format("Provider: %s,
Latitude: %s, Location: %s, Altitude: %s",
location.getProvider(), location.getLatitude(),
location.getLongitude(), location.getAltitude()));
            loggerArrayAdapter.notifyDataSetChanged();

        }

        //provider失效时调用
        @Override
        public void onProviderDisabled(String provider) {

            Log.d("Location", "onProviderDisabled");
            status.setText("onProviderDisabled");

        }

        //provider启用时调用
        @Override
        public void onProviderEnabled(String provider) {

            Log.d("Location", "onProviderEnabled");
            status.setText("onProviderEnabled");

        }

        //状态改变时调用
        @Override
        public void onStatusChanged(String provider, int
status, Bundle extras) {

            Log.d("Location", "onStatusChanged");
            //GPS状态变化时触发
            switch (status) {
                case LocationProvider.AVAILABLE:
                    Log.e("Location", "当前GPS状态为可见状态");
                    break;
                case LocationProvider.OUT_OF_SERVICE:
                    Log.e("Location", "当前GPS状态为服务区外状
态");
                    break;
                case
LocationProvider.TEMPORARILY_UNAVAILABLE:
```

```
                    Log.e("Location", "当前GPS状态为暂停服务状
态");
                    break;
            }

        }

    };


    @Override
    public void onRequestPermissionsResult(int requestCode,
String[] permissions, int[] grantResults) {
        super.onRequestPermissionsResult(requestCode,
permissions, grantResults);

        switch (requestCode) {
            case REQUEST_PERMISSION_CODE: {
                // If request is cancelled, the result arrays
are empty.
                if (grantResults.length > 0 &&
grantResults[0] == PackageManager.PERMISSION_GRANTED) {

                } else {
                    // permission denied, boo! Disable the
                    // functionality that depends on this
permission.
                }
                return;
            }

        }
    }
}
```

第 15 章 电话

1. SIM 卡状态

```
                TelephonyManager telephonyManager =
(TelephonyManager)context.getSystemService(context.TELEPHONY_SE
RVICE);

        if(telephonyManager.getSimState() ==
TelephonyManager.SIM_STATE_READY){
            return true;
        }else{
            return false;
        }
```

2. 通信录与拨打电话

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

    <uses-permission
android:name="android.permission.READ_CONTACTS"/>
    <uses-permission
android:name="android.permission.CALL_PHONE"/>

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

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

</manifest>
```

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.constraint.ConstraintLayout
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"
    tools:context=".MainActivity">

    <TextView
        android:id="@+id/textView2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="通信录与拨打电话"
        app:layout_constraintLeft_toLeftOf="parent"
        app:layout_constraintRight_toRightOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

    <Button
        android:id="@+id/contact"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginStart="8dp"
        android:layout_marginTop="60dp"
        android:layout_marginEnd="8dp"
        android:text="联系人"
        app:layout_constraintEnd_toStartOf="@+id/call"
        app:layout_constraintHorizontal_bias="0.478"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/phone" />

    <EditText
        android:id="@+id/phone"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginStart="16dp"
        android:layout_marginTop="64dp"
        android:layout_marginEnd="8dp"
        android:ems="10"
        android:inputType="phone"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.475"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/textView2"
/>

    <Button
        android:id="@+id/call"
        android:layout_width="wrap_content"
```

```
        android:layout_height="wrap_content"
        android:layout_marginTop="60dp"
        android:layout_marginEnd="52dp"
        android:text="Call"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/phone" />

</android.support.constraint.ConstraintLayout>
```

```
package cn.netkiller.contacts;

import android.app.Activity;
import android.content.ContentResolver;
import android.content.Intent;
import android.content.pm.PackageManager;
import android.database.Cursor;
import android.net.Uri;
import android.provider.ContactsContract;
import android.support.v4.app.ActivityCompat;
import android.support.v4.content.ContextCompat;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.TextView;
import android.widget.Toast;

public class MainActivity extends AppCompatActivity
implements View.OnClickListener {

    private EditText phone;
    private Button contact;
    private Button call;

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

```
        contact = (Button) findViewById(R.id.contact);
        contact.setOnClickListener(this);

        call = (Button) findViewById(R.id.call);
        call.setOnClickListener(this);

        phone = (EditText) findViewById(R.id.phone);
        phone.setText("", TextView.BufferType.EDITABLE);


        if (ActivityCompat.checkSelfPermission(this,
android.Manifest.permission.READ_CONTACTS) !=
PackageManager.PERMISSION_GRANTED) {
            ActivityCompat.requestPermissions((Activity)
this, new String[]
{android.Manifest.permission.READ_CONTACTS}, 1);
        }
        if (ActivityCompat.checkSelfPermission(this,
android.Manifest.permission.CALL_PHONE) !=
PackageManager.PERMISSION_GRANTED) {
            ActivityCompat.requestPermissions((Activity)
this, new String[]{android.Manifest.permission.CALL_PHONE},
1);
        }


    }

    @Override
    public void onClick(View v) {
        switch (v.getId()) {
            case R.id.contact:
                Toast.makeText(MainActivity.this, "获取联系人手
机号码", Toast.LENGTH_SHORT).show();
                startActivityForResult(new
Intent(Intent.ACTION_PICK,
ContactsContract.Contacts.CONTENT_URI), 0);
                break;
            case R.id.call:
                Toast.makeText(MainActivity.this, "打电话",
Toast.LENGTH_SHORT).show();

                Intent intent = new Intent();
```

```
intent.setAction(Intent.ACTION_CALL);//ACTION_DIAL
                intent.setData(Uri.parse("tel:" +
phone.getText()));
                startActivity(intent);

                break;
        }

    }

    @Override
    protected void onActivityResult(int requestCode, int
resultCode, Intent data) {
        super.onActivityResult(requestCode, resultCode,
data);
        if (resultCode == Activity.RESULT_OK) {
            Uri contactData = data.getData();
            Cursor cursor =
getContentResolver().query(contactData, null, null, null,
null);
            cursor.moveToFirst();

            //条件为联系人ID
            String contactId =
cursor.getString(cursor.getColumnIndex(ContactsContract.Conta
cts._ID));
            //获得DATA表中的电话号码，条件为联系人ID,因为手机号码可能
会有多个
            Cursor contact =
getContentResolver().query(ContactsContract.CommonDataKinds.P
hone.CONTENT_URI, null,
ContactsContract.CommonDataKinds.Phone.CONTACT_ID + "=" +
contactId, null, null);
            while (contact.moveToNext()) {
                String contactNumber =
contact.getString(contact.getColumnIndex(ContactsContract.Com
monDataKinds.Phone.NUMBER));

                phone.setText(contactNumber);
            }
            cursor.close();

        }
    }
```

```
}
```

3. 发送短信

```
<uses-permission android:name="android.permission.SEND_SMS"
/>
```

```
        private void sendSMS(String phoneNumber,String
message){
        if(PhoneNumberUtils.isGlobalPhoneNumber(phoneNumber))
{
            Intent intent = new Intent(Intent.ACTION_SENDTO,
Uri.parse("smsto:"+phoneNumber));
            intent.putExtra("sms_body", message);
            startActivity(intent);
        }
    }
```

```
        public void sendSMS(String phoneNumber,String
message){
        android.telephony.SmsManager smsManager =
android.telephony.SmsManager.getDefault();
        //拆分短信内容，手机短信长度有限制
        List<String> divideContents =
smsManager.divideMessage(message);
        for (String text : divideContents) {
            smsManager.sendTextMessage(phoneNumber, null,
text, sentPI, deliverPI);
        }
    }
```

第 16 章 消息广播

安卓中有两种广播，一种是系统发出的广播信息，例如网络改变，电池的电量低等等，另一种是用户发出的广播信息。

Android 中的广播类型可以分为两种类型，标准广播和有序广播。

标准广播（Normal broadcasts）：标准广播是一种完全异步执行的广播，在广播发出之后，所有的广播接收器几乎会在同一时刻接收到这条广播消息。这种广播效率比较高，但同时也意味着它是无法被截断的。

有序广播（Ordered broadcasts）：有序广播则是一种同步执行的广播，在广播发出之后，同一时刻只会有一个广播接收器能够收到这条广播消息，当这个广播接收器中的逻辑执行完毕之后，广播才会继续传递。

```
android.intent.action.BATTERY_CHANGED    持久广播含充电状态，级别，以
及其他相关的电池信息。
android.intent.action.BATTERY_LOW                 显示设备的电池电量
低。
android.intent.action.BATTERY_OKAY                指示电池正在低点后
但没有问题。
android.intent.action.BOOT_COMPLETED     一次播出后，系统已完成启动。
android.intent.action.BUG_REPORT                  显示活动报告的错
误。
android.intent.action.CALL                                  执行呼叫
由数据指定某人。
android.intent.action.CALL_BUTTON                 用户按下“呼叫”按钮
进入拨号器或其他适当的用户界面发出呼叫。
android.intent.action.DATE_CHANGED                日期改变。
android.intent.action.REBOOT                      有设备重启。
```

1. 动态注册

```
package cn.netkiller.broadcast;

import android.content.BroadcastReceiver;
import android.content.Context;
import android.content.Intent;
import android.content.IntentFilter;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.widget.Toast;

public class MainActivity extends AppCompatActivity {

    private IntentFilter intentFilter;
    private MyBroadcastReceiver myBroadcastReceiver;

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

        intentFilter = new IntentFilter();
        //为过滤器添加处理规则

intentFilter.addAction("android.net.conn.CONNECTIVITY_CHANGE"
);
        myBroadcastReceiver = new MyBroadcastReceiver();
        //注册广播接收器
        registerReceiver(myBroadcastReceiver, intentFilter);
    }

    @Override
    protected void onDestroy() {
        super.onDestroy();
        //注销动态的广播接收器
        unregisterReceiver(myBroadcastReceiver);
    }

    //自定义内部类，继承 BroadcastReceiver
    public class MyBroadcastReceiver extends
```

```
BroadcastReceiver {

        @Override
        public void onReceive(Context context, Intent intent)
{
            Toast.makeText(context, "网络状态已改变",
Toast.LENGTH_SHORT).show();
        }
    }
}
```

现在尝试改变网络状态，例如开启或关闭飞行模式，程序会弹出"网络状态已改变"。

我的测试环境是 Android 9 Pie 没有加入下面的权限仍然能工作

```
<uses-permission
android:name="android.permission.ACCESS_NETWORK_STATE" />
```

优化程序

```
package cn.netkiller.broadcast;

import android.content.BroadcastReceiver;
import android.content.Context;
import android.content.Intent;
import android.content.IntentFilter;
import android.net.ConnectivityManager;
import android.net.NetworkInfo;
import android.os.BatteryManager;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
```

```
import android.widget.Toast;

public class MainActivity extends AppCompatActivity {

    private IntentFilter intentFilter;
    private MyBroadcastReceiver myBroadcastReceiver;

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

        intentFilter = new IntentFilter();
        //为过滤器添加处理规则

intentFilter.addAction("android.net.conn.CONNECTIVITY_CHANGE"
);

intentFilter.addAction(ConnectivityManager.CONNECTIVITY_ACTIO
N);

intentFilter.addAction(Intent.ACTION_BATTERY_CHANGED);
        intentFilter.addAction(Intent.ACTION_BATTERY_LOW);

        myBroadcastReceiver = new MyBroadcastReceiver();
        //注册广播接收器
        registerReceiver(myBroadcastReceiver, intentFilter);
    }

    @Override
    protected void onDestroy() {
        super.onDestroy();
        //注销动态的广播接收器
        unregisterReceiver(myBroadcastReceiver);
    }

    //自定义内部类，继承 BroadcastReceiver
    public class MyBroadcastReceiver extends
BroadcastReceiver {

        @Override
        public void onReceive(Context context, Intent intent)
{
            ConnectivityManager connectivityManager =
(ConnectivityManager)
```

```
getSystemService(Context.CONNECTIVITY_SERVICE);
            NetworkInfo networkInfo =
connectivityManager.getActiveNetworkInfo();
            //判断是否联网
            if (networkInfo != null &&
networkInfo.isConnected()) {
                Toast.makeText(context, "网络已连接",
Toast.LENGTH_SHORT).show();
            } else {
                Toast.makeText(context, "网络不可用",
Toast.LENGTH_SHORT).show();
            }

            int status =
intent.getIntExtra(BatteryManager.EXTRA_STATUS, -1);
            boolean isCharging = status ==
BatteryManager.BATTERY_STATUS_CHARGING ||
                    status ==
BatteryManager.BATTERY_STATUS_FULL;

            if (isCharging) {
                Toast.makeText(context, "正在充电",
Toast.LENGTH_SHORT).show();
            } else {
                Toast.makeText(context, "电池已经充满",
Toast.LENGTH_SHORT).show();
            }

            int chargePlug =
intent.getIntExtra(BatteryManager.EXTRA_PLUGGED, -1);
            boolean usbCharge = chargePlug ==
BatteryManager.BATTERY_PLUGGED_USB;
            boolean acCharge = chargePlug ==
BatteryManager.BATTERY_PLUGGED_AC;
            if (usbCharge) {
                Toast.makeText(context, "USB 充电",
Toast.LENGTH_SHORT).show();
            }

        }
    }
}
```

2. 静态注册

Android Studio 选择 File - New - Other - Broadcast Receiver

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

    <uses-permission
android:name="android.permission.ACCESS_NETWORK_STATE" />
    <uses-permission
android:name="android.permission.RECEIVE_BOOT_COMPLETED" />

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

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

        <receiver
            android:name=".MyReceiver"
            android:enabled="true"
            android:exported="true">

            <intent-filter>
                <action
android:name="android.intent.action.BOOT_COMPLETED" />
            </intent-filter>
```

```
        </receiver>
    </application>

</manifest>
```

MyReceiver 集成 BroadcastReceiver 在 onReceive 中写入你的业务逻辑。

```
package cn.netkiller.broadcast;

import android.content.BroadcastReceiver;
import android.content.Context;
import android.content.Intent;
import android.widget.Toast;

public class MyReceiver extends BroadcastReceiver {

    @Override
    public void onReceive(Context context, Intent intent) {
        Toast.makeText(context, "程序已启动，接收到系统启动广播",
Toast.LENGTH_SHORT).show();
    }
}
```

现在重启 Android 模拟器，启动后虽然 App 并没有进入，但是屏幕底部会看到 "程序已启动，接收到系统启动广播"

3. 自定义用户消息广播

```
package cn.netkiller.broadcast;

import android.content.BroadcastReceiver;
import android.content.Context;
import android.content.Intent;
import android.content.IntentFilter;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.TextView;
import android.widget.Toast;

public class MainActivity extends AppCompatActivity {

    private IntentFilter intentFilter;
    private TextView textView;
    private MyBroadcastReceiver myBroadcastReceiver;

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

        intentFilter = new IntentFilter();

intentFilter.addAction("cn.netkiller.broadcast.MESSAGE");
        myBroadcastReceiver = new MyBroadcastReceiver();
        //注册广播接收器
        registerReceiver(myBroadcastReceiver, intentFilter);

        textView = (TextView) findViewById(R.id.textView);

        Button button = (Button) findViewById(R.id.button);
        button.setOnClickListener(new View.OnClickListener()
{
            @Override
            public void onClick(View v) {
```

```
                //把要发送的广播值传入Intent对象
                Intent intent = new
Intent("cn.netkiller.broadcast.MESSAGE");
                intent.putExtra("msg", "Helloworld");
                //调用Context的 sendBroadcast()方法发送广播
                sendBroadcast(intent);
                textView.setText("Send");

            }
        });
    }

    @Override
    protected void onDestroy() {
        super.onDestroy();
        //注销动态的广播接收器
        unregisterReceiver(myBroadcastReceiver);
    }

    //自定义内部类，继承 BroadcastReceiver
    public class MyBroadcastReceiver extends
BroadcastReceiver {

        @Override
        public void onReceive(Context context, Intent intent)
{

            String data = intent.getStringExtra("msg");
            Toast.makeText(context, data,
Toast.LENGTH_SHORT).show();

        }
    }
}
```

4. 本地广播

注意：LocalBroadcastManager 已经被废弃

上面讲的系统广播是全局的，任何APP都能接收到你的广播，这样就很容易引起APP的安全性问题。很多时候我们只想接收来自本应用程序发出的广播。

```
package cn.netkiller.broadcast;

import android.content.BroadcastReceiver;
import android.content.Context;
import android.content.Intent;
import android.content.IntentFilter;
import android.support.v4.content.LocalBroadcastManager;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.TextView;
import android.widget.Toast;

public class MainActivity extends AppCompatActivity {

    private TextView textView;
    private IntentFilter intentFilter;
    private LocalBroadcastManager localBroadcastManager;
    private MyBroadcastReceiver myBroadcastReceiver;

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

        //获取LocalBroadcastManger 单li实例
        localBroadcastManager =
localBroadcastManager.getInstance(this);

        intentFilter = new IntentFilter();
```

```
intentFilter.addAction("cn.netkiller.broadcast.MESSAGE");
        myBroadcastReceiver = new MyBroadcastReceiver();
        //注册本地广播接收器

localBroadcastManager.registerReceiver(myBroadcastReceiver,
intentFilter);

        textView = (TextView) findViewById(R.id.textView);

        Button button = (Button) findViewById(R.id.button);
        button.setOnClickListener(new View.OnClickListener()
{
            @Override
            public void onClick(View v) {

                textView.setText("Send");

                Intent intent = new Intent();

intent.setAction("cn.netkiller.broadcast.MESSAGE");
                intent.putExtra("msg",
"http://www.netkiller.cn");
                //发送本地广播
                localBroadcastManager.sendBroadcast(intent);
            }
        });
    }

    @Override
    protected void onDestroy() {
        super.onDestroy();
        //注销本地广播接收器

localBroadcastManager.unregisterReceiver(myBroadcastReceiver)
;
    }

    //自定义内部类，继承 BroadcastReceiver
    public class MyBroadcastReceiver extends
BroadcastReceiver {

        @Override
        public void onReceive(Context context, Intent intent)
{
```

```
            String data = intent.getStringExtra("msg");
            Toast.makeText(context, data,
Toast.LENGTH_SHORT).show();

        }
    }
}
```

第 17 章 Service 服务

手动调用方法

```
手动调用方法                作用
startService()  启动服务
stopService()   关闭服务
bindService()   绑定服务
unbindService() 解绑服务
```

自动调用的方法

```
自动调用方法                        作用
onCreate()                          创建服务
onStartCommand()            开始服务
onDestroy()                         销毁服务
onBind()                            绑定服务
onUnbind()                          解绑服务
```

生命周期调用

```
1. 启动Service服务

单次: startService() -> onCreate() -> onStartCommand()
多次: startService() -> onCreate() -> onStartCommand() ->
onStartCommand()

2. 停止Service服务
```

```
stopService() -> onDestroy()

3. 绑定Service服务

bindService() -> onCreate() -> onBind()

4. 解绑Service服务

unbindService() -> onUnbind() -> onDestroy()

5. 启动绑定Service服务

startService() -> onCreate() -> onStartCommand() ->
bindService() -> onBind()

6. 解绑停止Service服务

unbindService() -> onUnbind() -> stopService() -> onDestroy()

7. 解绑绑定Service服务

unbindService() -> onUnbind(ture) -> bindService() ->
onRebind()
```

1. Service的基本用法

1.1. manifest 文件

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

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
```

```
        android:label="@string/app_name"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/AppTheme">
        <activity android:name=".MainActivity">
            <intent-filter>
                <action
android:name="android.intent.action.MAIN" />

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

        <service
            android:name=".MyService"
            android:enabled="true"
            android:exported="true"></service>
    </application>

</manifest>
```

这段代码不是手工加入的，只需在 Android Studio 中选择 File - New - Service - Service 创建 Service 会自动加入下面代码

```
<service
            android:name=".MyService"
            android:enabled="true"
            android:exported="true"></service>
```

1.2. 创建 Service

在 Android Studio 中选择 File - New - Service - Service 创建 Service

MyService继承自Service，并重写父类的onCreate()、onStartCommand()和onDestroy()方法

```
package cn.netkiller.service;

import android.app.Service;
import android.content.Intent;
import android.os.IBinder;
import android.util.Log;

public class MyService extends Service {
    public MyService() {
    }

    @Override
    public void onCreate() {
        super.onCreate();
        Log.d("Service", "onCreate() executed");
        Log.d("Service", "MyService thread id is " +
Thread.currentThread().getId());
    }

    @Override
    public int onStartCommand(Intent intent, int flags, int
startId) {
        Log.d("Service", "onStartCommand() executed");
        return super.onStartCommand(intent, flags, startId);
    }

    @Override
    public void onDestroy() {
        super.onDestroy();
        Log.d("Service", "onDestroy() executed");
    }

    @Override
    public IBinder onBind(Intent intent) {
        // TODO: Return the communication channel to the
service.
        throw new UnsupportedOperationException("Not yet
implemented");
    }
```

```
}
```

onCreate() Service 创建的时候执行，已经创建的Service不会再执行

onStartCommand() 任何时候，只要执行 startService(intent); 便会执行

onDestroy() 停止的时候执行

1.3. Layout 代码

在布局文件中加入了两个按钮，一个用于启动Service，一个用于停止Service

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.constraint.ConstraintLayout
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"
    tools:context=".MainActivity">

    <LinearLayout
        android:layout_width="368dp"
        android:layout_height="229dp"
        android:layout_marginStart="8dp"
        android:layout_marginTop="8dp"
        android:layout_marginEnd="8dp"
        android:layout_marginBottom="8dp"
        android:orientation="vertical"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.0"
        app:layout_constraintStart_toStartOf="parent"
```

```
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.0">

        <Button
            android:id="@+id/startService"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:text="Start service" />

        <Button
            android:id="@+id/stopService"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:text="Stop service" />
    </LinearLayout>

</android.support.constraint.ConstraintLayout>
```

1.4. Activity 代码

```
package cn.netkiller.service;

import android.content.Intent;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.util.Log;
import android.view.View;
import android.widget.Button;

public class MainActivity extends AppCompatActivity
implements View.OnClickListener {

    private Button startService;
    private Button stopService;

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

```
        startService = (Button)
findViewById(R.id.startService);
        stopService = (Button)
findViewById(R.id.stopService);
        startService.setOnClickListener(this);
        stopService.setOnClickListener(this);

        Log.d("Service", "MainActivity thread id is " +
Thread.currentThread().getId());

    }

    @Override
    public void onClick(View v) {
        Intent intent;
        switch (v.getId()) {
            case R.id.startService:
                intent = new Intent(this, MyService.class);
                startService(intent);
                break;
            case R.id.stopService:
                intent = new Intent(this, MyService.class);
                stopService(intent);
                break;
            default:
                break;
        }
    }
}
```

2. Service 中启动线程

```
    @Override
    public int onStartCommand(Intent intent, int flags, int
startId) {

        Log.d("Service", "onStartCommand() begin");
        new Thread(new Runnable() {
            @Override
            public void run() {
                // 开始执行后台任务
                Log.d("Service", "onStartCommand()
executed");
            }
        }).start();

        Log.d("Service", "onStartCommand() end");

        return super.onStartCommand(intent, flags, startId);
    }
```

3. Service 和 Activity 通信

3.1. Layout

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.constraint.ConstraintLayout
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"
    tools:context=".MainActivity">

    <LinearLayout
        android:layout_width="368dp"
        android:layout_height="229dp"
        android:layout_marginStart="8dp"
        android:layout_marginTop="8dp"
        android:layout_marginEnd="8dp"
        android:layout_marginBottom="8dp"
        android:orientation="vertical"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.0"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.0">

        <Button
            android:id="@+id/startService"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:text="Start service" />

        <Button
            android:id="@+id/stopService"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:text="Stop service" />
```

```
        <Button
            android:id="@+id/bindService"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:text="Bind Service" />

        <Button
            android:id="@+id/unbindService"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:text="Unbind Service" />

    </LinearLayout>

</android.support.constraint.ConstraintLayout>
```

3.2. Service

```
package cn.netkiller.service;

import android.app.Service;
import android.content.Intent;
import android.os.Binder;
import android.os.IBinder;
import android.util.Log;

public class MyService extends Service {

    private MyBinder myBinder = new MyBinder();

    public MyService() {
    }

    @Override
    public void onCreate() {
        super.onCreate();
        Log.d("Service", "onCreate() executed");
        Log.d("Service", "MyService thread id is " +
```

```
Thread.currentThread().getId());
    }

    @Override
    public int onStartCommand(Intent intent, int flags, int
startId) {

        Log.d("Service", "onStartCommand() begin");
        new Thread(new Runnable() {
            @Override
            public void run() {
                // 开始执行后台任务
                Log.d("Service", "onStartCommand()
executed");
            }
        }).start();

        Log.d("Service", "onStartCommand() end");

        return super.onStartCommand(intent, flags, startId);
    }

    @Override
    public void onDestroy() {
        super.onDestroy();
        Log.d("Service", "onDestroy() executed");
    }

    @Override
    public IBinder onBind(Intent intent) {
        return myBinder;
    }

    class MyBinder extends Binder {

        public void startTask() {
            new Thread(new Runnable() {
                @Override
                public void run() {
                    // 执行具体的任务
                    Log.d("Service", "startTask()");
                }
            }).start();
        }
```

```
    }
}
```

3.3. Activity

```
package cn.netkiller.service;

import android.content.ComponentName;
import android.content.Intent;
import android.content.ServiceConnection;
import android.os.IBinder;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.util.Log;
import android.view.View;
import android.widget.Button;

public class MainActivity extends AppCompatActivity
implements View.OnClickListener {

    private Button startService;
    private Button stopService;
    private Button bindService;
    private Button unbindService;

    private MyService.MyBinder myBinder;

    private ServiceConnection connection = new
ServiceConnection() {

        @Override
        public void onServiceDisconnected(ComponentName name)
{
        }

        @Override
        public void onServiceConnected(ComponentName name,
IBinder service) {
            myBinder = (MyService.MyBinder) service;
```

```
            myBinder.startTask();
        }
    };

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

        startService = (Button)
findViewById(R.id.startService);
        stopService = (Button)
findViewById(R.id.stopService);
        startService.setOnClickListener(this);
        stopService.setOnClickListener(this);

        bindService = (Button)
findViewById(R.id.bindService);
        unbindService = (Button)
findViewById(R.id.unbindService);
        bindService.setOnClickListener(this);
        unbindService.setOnClickListener(this);

        Log.d("Service", "MainActivity thread id is " +
Thread.currentThread().getId());

    }

    @Override
    public void onClick(View v) {
        Intent intent;
        switch (v.getId()) {
            case R.id.startService:
                intent = new Intent(this, MyService.class);
                startService(intent);
                break;
            case R.id.stopService:
                intent = new Intent(this, MyService.class);
                stopService(intent);
                break;
            case R.id.bindService:
                Intent bindIntent = new Intent(this,
MyService.class);
                bindService(bindIntent, connection,
BIND_AUTO_CREATE);
```

```
                break;
            case R.id.unbindService:
                unbindService(connection);
                break;
            default:
                break;
        }
    }
}
```

4. Service 和 Toast

```
Handler handler=new Handler(Looper.getMainLooper());
handler.post(new Runnable(){
    public void run(){
        Toast.makeText(getApplicationContext() ,"显示Toast在屏
幕上！",Toast.LENGTH_LONG).show();
    }
});
```

```
Handler handler = new Handler(Looper.getMainLooper());
handler.post(() -> {
    Toast.makeText(getApplicationContext(), "显示Toast在屏幕
上！", Toast.LENGTH_LONG).show();
    notify1();
});
```

5. Service 中启动 Activity

```
        Intent intent = new Intent(getBaseContext(),
FullscreenActivity.class);
    intent.addFlags(Intent.FLAG_ACTIVITY_NEW_TASK);
    getApplication().startActivity(intent);
```

6. Service 中更新 UI

```
// 在 Service 中定义一个 Handler
private Handler mHandler = new
Handler(Looper.getMainLooper());

// 在 Service 中定义一个 Runnable 对象
private Runnable mRunnable = new Runnable() {
    @Override
    public void run() {
        // 在这里执行与 UI 相关的操作
    }
};

// 在 Service 中使用 Handler 将 Runnable 对象发送到 UI 线程的消息队
列中
mHandler.post(mRunnable);
```

```
private Handler handler = new
Handler(Looper.getMainLooper());
handler.post(() -> {
    musicSkillComponent.stop();
});
```

第 18 章 Notification 通知中心

1. 文本通知

```
    private int createNotification(String title, String text)
{
        String channelId = "channelId";
        String channelName = "channelName";
        String description = "description";
        String group = "group";
        int notificationId = new Random().nextInt(101);
        NotificationManagerCompat notificationManagerCompat =
NotificationManagerCompat.from(this);
        NotificationChannel channel = new
NotificationChannel(channelId, channelName,
NotificationManager.IMPORTANCE_HIGH);
        channel.setDescription(description);

notificationManagerCompat.createNotificationChannel(channel);

        NotificationCompat.Builder notification = new
NotificationCompat.Builder(this, channelId)
                .setContentTitle(title).setContentText(text)
                .setSmallIcon(R.mipmap.ic_launcher)

.setPriority(NotificationCompat.PRIORITY_HIGH)
                .setAutoCancel(true).setGroup(group);

        notificationManagerCompat.notify(notificationId,
notification.build());
        return notificationId;
    }
```

```
        CharSequence name = "test";
        String description = "test";
        NotificationChannel channel = new
NotificationChannel("test", name,
NotificationManager.IMPORTANCE_DEFAULT);
        channel.setDescription(description);


        NotificationManager notificationManager =
(NotificationManager)
getSystemService(Context.NOTIFICATION_SERVICE);

notificationManager.createNotificationChannel(channel);
        Notification.Builder builder = new
Notification.Builder(this, "test")
                .setContentTitle("XXX 门票打折")          //标
题
                .setContentText("参与 XXX 领取 XXX")      //内
容
                .setSubText("打折信息")                   //内
容下面的一小段文字
                .setWhen(System.currentTimeMillis())     //设
置通知时间
                .setSmallIcon(R.mipmap.ic_launcher)      //设
置小图标
                .setAutoCancel(false);                   //设
置点击后取消Notification

        notificationManager.notify(1, builder.build());
```

2. 添加点击操作

```
    private int createNotification(String title, String text)
{
        String channelId = "channelId";
        String channelName = "channelName";
        String description = "description";
        String group = "group";
        int notificationId = new Random().nextInt(101);

        Intent intent = new Intent(this,
FullscreenActivity.class);
        intent.setFlags(Intent.FLAG_ACTIVITY_NEW_TASK |
Intent.FLAG_ACTIVITY_CLEAR_TASK);
        PendingIntent pendingIntent =
PendingIntent.getActivity(this, 0, intent,
PendingIntent.FLAG_IMMUTABLE);

        NotificationManagerCompat notificationManagerCompat =
NotificationManagerCompat.from(this);
        NotificationChannel channel = new
NotificationChannel(channelId, channelName,
NotificationManager.IMPORTANCE_HIGH);
        channel.setDescription(description);

notificationManagerCompat.createNotificationChannel(channel);

        NotificationCompat.Builder notification = new
NotificationCompat.Builder(this, channelId)
                .setContentTitle(title).setContentText(text)
                .setSmallIcon(R.mipmap.ic_launcher)

.setPriority(NotificationCompat.PRIORITY_HIGH)
                .setContentIntent(pendingIntent)
                .setAutoCancel(true);

        notificationManagerCompat.notify(notificationId,
notification.build());
        return notificationId;
    }
```

第 19 章 NFC (Near field communication)

1. AndroidManifest.xml 文件配置

```
<?xml version="1.0" encoding="utf-8"?>
<manifest
xmlns:android="http://schemas.android.com/apk/res/android"
    package="cn.netkiller.nfc">
    <!--<uses-sdk android:minSdkVersion="14"/>-->
    <uses-permission android:name="android.permission.NFC" />
    <!-- 要求当前设备必须要有NFC芯片 -->
    <uses-feature
        android:name="android.hardware.nfc"
        android:required="true" />

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="NFC 初始化工具"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/AppTheme">
        <activity
            android:name=".MainActivity"
            android:launchMode="singleTop">

            <intent-filter>
                <action
android:name="android.intent.action.MAIN" />

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

```
            <intent-filter>
                <action
android:name="android.nfc.action.NDEF_DISCOVERED" />
                <category
android:name="android.intent.category.DEFAULT" />
                <data android:mimeType="text/plain" />

                <!--<data android:mimeType="text/plain" />-->
                <!--<data android:mimeType="*/*" />-->
            </intent-filter>
            <intent-filter>
                <action
android:name="android.nfc.action.TECH_DISCOVERED" />
            </intent-filter>

            <intent-filter>
                <action
android:name="android.nfc.action.TAG_DISCOVERED" />
                <category
android:name="android.intent.category.DEFAULT" />
            </intent-filter>

        </activity>

    </application>

</manifest>
```

2. Loyout 文件

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.constraint.ConstraintLayout
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"
    tools:context=".MainActivity">


    <TextView
        android:id="@+id/ndefMessage"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_marginStart="8dp"
        android:layout_marginTop="20dp"
        android:layout_marginEnd="8dp"
        android:text="message"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="1.0"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/textView4"
/>

    <TextView
        android:id="@+id/textView4"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginStart="8dp"
        android:layout_marginTop="16dp"
        android:layout_marginEnd="8dp"
        android:text="NDEF Message : "
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.03"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

    <TableLayout
```

```
        android:id="@+id/tableLayout"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_marginStart="10dp"
        android:layout_marginTop="24dp"
        android:layout_marginEnd="8dp"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.0"
        app:layout_constraintStart_toStartOf="parent"

app:layout_constraintTop_toBottomOf="@+id/ndefMessage">

        <TableRow
            android:layout_width="match_parent"
            android:layout_height="match_parent">

            <TextView
                android:id="@+id/textView"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:text="NFC UID" />

            <TextView
                android:id="@+id/uid"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:text="UID"
                tools:layout_editor_absoluteX="146dp"
                tools:layout_editor_absoluteY="71dp" />
        </TableRow>

        <TableRow
            android:layout_width="match_parent"
            android:layout_height="match_parent">

            <TextView
                android:id="@+id/textView2"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:text="NFC Tag" />

            <TextView
                android:id="@+id/nfcTag"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
```

```
            android:text="tag"
            tools:layout_editor_absoluteX="179dp"
            tools:layout_editor_absoluteY="83dp" />
        </TableRow>

        <TableRow
            android:layout_width="match_parent"
            android:layout_height="match_parent">

            <TextView
                android:id="@+id/textView3"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:text="NFC Size" />

            <TextView
                android:id="@+id/size"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:text="size"
                tools:layout_editor_absoluteX="179dp"
                tools:layout_editor_absoluteY="150dp" />
        </TableRow>

        <TableRow
            android:layout_width="match_parent"
            android:layout_height="match_parent">

            <TextView
                android:id="@+id/textView5"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:text="NDEF Type" />

            <TextView
                android:id="@+id/schema"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:text="schema"
                tools:layout_editor_absoluteX="168dp"
                tools:layout_editor_absoluteY="257dp" />
        </TableRow>

        <TableRow
            android:layout_width="match_parent"
```

```
            android:layout_height="match_parent">

        </TableRow>


        <TableRow
            android:layout_width="match_parent"
            android:layout_height="match_parent">

            <TextView
                android:id="@+id/textView7"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:text="NDEF charset" />

            <TextView
                android:id="@+id/charset"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:text="charset"
                tools:layout_editor_absoluteX="163dp"
                tools:layout_editor_absoluteY="304dp" />


        </TableRow>


        <TableRow
            android:layout_width="match_parent"
            android:layout_height="match_parent">

            <TextView
                android:id="@+id/textView8"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:text="NDEF Lang" />

            <TextView
                android:id="@+id/language"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:text="lang"
                tools:layout_editor_absoluteX="163dp"
                tools:layout_editor_absoluteY="331dp" />
        </TableRow>
```

```
        <TableRow
            android:layout_width="match_parent"
            android:layout_height="match_parent">

            <TextView
                android:id="@+id/textView6"
                android:layout_width="79dp"
                android:layout_height="wrap_content"
                android:text="Status" />

            <TextView
                android:id="@+id/status"
                android:layout_width="289dp"
                android:layout_height="wrap_content"
                android:text="status"
                tools:layout_editor_absoluteX="90dp"
                tools:layout_editor_absoluteY="404dp" />


        </TableRow>
    </TableLayout>


    <TextView
        android:id="@+id/textView9"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginStart="8dp"
        android:layout_marginTop="48dp"
        android:layout_marginEnd="8dp"
        android:text="NDEF Message write :"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.0"
        app:layout_constraintStart_toStartOf="parent"

app:layout_constraintTop_toBottomOf="@+id/tableLayout" />

    <TextView
        android:id="@+id/ndefWrite"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_marginStart="8dp"
        android:layout_marginTop="8dp"
```

```
        android:layout_marginEnd="8dp"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="1.0"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/textView9"
/>

    <Switch
        android:id="@+id/switchWrite"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginStart="8dp"
        android:layout_marginTop="28dp"
        android:layout_marginEnd="8dp"
        android:text="NDEF Message write"
        android:textOff="NDEF Message write Off"
        android:textOn="NDEF Message write On"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.497"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/ndefWrite"
/>

</android.support.constraint.ConstraintLayout>
```

3. Activity 文件

```
package cn.netkiller.nfc;


import android.nfc.FormatException;
import android.nfc.NdefRecord;
import android.support.annotation.NonNull;
import android.support.design.widget.BottomNavigationView;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;

import android.app.PendingIntent;
import android.content.Intent;
import android.nfc.NdefMessage;
import android.nfc.NfcAdapter;

import android.nfc.Tag;
import android.nfc.tech.Ndef;
import android.os.Parcelable;
import android.view.MenuItem;
import android.view.View;
import android.widget.Button;
import android.widget.CompoundButton;
import android.widget.Switch;
import android.widget.TextView;

import java.io.IOException;
import java.io.UnsupportedEncodingException;
import java.math.BigInteger;
import java.util.UUID;

public class MainActivity extends AppCompatActivity {

    private NfcAdapter nfcAdapter;
    private PendingIntent pendingIntent;

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

```
        setContentView(R.layout.activity_main);

        final TextView status = (TextView)
findViewById(R.id.status);

        nfcAdapter = NfcAdapter.getDefaultAdapter(this);

        if (nfcAdapter == null) {
            System.out.println("**** NFC ERROR ****");
            status.setText("NFC is not available.");
            return;
        } else if (!nfcAdapter.isEnabled()) {
            status.setText("请开启系统NFC功能");
        }

        status.setText("Start...");

        pendingIntent = PendingIntent.getActivity(this, 0,
new Intent(this,
getClass()).addFlags(Intent.FLAG_ACTIVITY_SINGLE_TOP), 0);


        final Switch switchWrite = (Switch)
findViewById(R.id.switchWrite);

        switchWrite.setOnCheckedChangeListener(new
CompoundButton.OnCheckedChangeListener() {
            @Override
            public void onCheckedChanged(CompoundButton
buttonView, boolean isChecked) {

                if (isChecked) {

status.setText(switchWrite.getTextOn().toString());
                } else {

status.setText(switchWrite.getTextOff().toString());
                }
            }
        });
    }

    @Override
    protected void onResume() {
        super.onResume();
```

```
        nfcAdapter.enableForegroundDispatch(this,
pendingIntent, null, null);
    }

    @Override
    protected void onPause() {
        super.onPause();
        nfcAdapter.disableForegroundDispatch(this);
    }


    //当窗口的创建模式是singleTop或singleTask时调用，用于取代
onCreate方法
    //当NFC标签靠近手机，建立连接后调用
    @Override
    public void onNewIntent(Intent intent) {
        super.onNewIntent(intent);

        TextView status = (TextView)
findViewById(R.id.status);
        TextView type = (TextView) findViewById(R.id.nfcTag);
        TextView size = (TextView) findViewById(R.id.size);
        TextView uidTextView = (TextView)
findViewById(R.id.uid);
        TextView ndefMessage = (TextView)
findViewById(R.id.ndefMessage);
        TextView schema = (TextView)
findViewById(R.id.schema);
        TextView charset = (TextView)
findViewById(R.id.charset);
        TextView language = (TextView)
findViewById(R.id.language);
        TextView ndefWrite = (TextView)
findViewById(R.id.ndefWrite);
        Switch switchWrite = (Switch)
findViewById(R.id.switchWrite);


        Tag tag =
intent.getParcelableExtra(NfcAdapter.EXTRA_TAG);

        if (switchWrite.isChecked()) {

            UUID uuid = UUID.randomUUID();
            try {
```

```
                write(uuid.toString(), tag);
                ndefWrite.setText(uuid.toString());
            } catch (IOException e) {
                e.printStackTrace();
            } catch (FormatException e) {
                e.printStackTrace();
            }

        }

        if
(NfcAdapter.ACTION_NDEF_DISCOVERED.equals(intent.getAction())
) {
            status.setText("Read NDEF Message...");

            byte[] uid = tag.getId();
            BigInteger n = new BigInteger(uid);
            String hex = n.toString(16);
            uidTextView.setText(hex);

            Ndef ndef = Ndef.get(tag);
            String log = ndef.getType() + "\n最大数据容量: " +
ndef.getMaxSize() + " bytes\n\n";
            System.out.println(log);
            type.setText(ndef.getType());
            size.setText(ndef.getMaxSize() + " bytes");

            Parcelable[] rawMessages =
intent.getParcelableArrayExtra(NfcAdapter.EXTRA_NDEF_MESSAGES
);
            if (rawMessages != null) {
                NdefMessage[] messages = new
NdefMessage[rawMessages.length];
                for (int i = 0; i < rawMessages.length; i++)
{
                    messages[i] = (NdefMessage)
rawMessages[i];
                }
                byte[] payload = messages[0].getRecords()
[0].getPayload();

                try {

                    String tagId = new
String(messages[0].getRecords()[0].getType());
```

```
                    schema.setText(tagId);

                    String encoding = ((payload[0] & 128) ==
0) ? "UTF-8" : "UTF-16";
                    charset.setText(encoding);

                    int languageCodeLength = payload[0] &
0x3f;
                    String languageCode = new String(payload,
1, languageCodeLength, "US-ASCII");


//                    String lang = new String(payload, 1,
payload[0] & 0063, "US-ASCII");
                    language.setText(languageCode);

//                    String text = new
String(messages[0].getRecords()[0].getPayload());
                    String text = new String(payload,
languageCodeLength + 1, payload.length - languageCodeLength -
1, encoding);
                    ndefMessage.setText(text);

                } catch (UnsupportedEncodingException e) {
                    e.printStackTrace();
                }

            }

        } else {
            status.setText("NOT NDEF Messages tag");
        }
    }


    private void write(String text, Tag tag) throws
IOException, FormatException {
        NdefRecord[] records = {createRecord(text)};
        NdefMessage message = new NdefMessage(records);
        // Get an instance of Ndef for the tag.
        Ndef ndef = Ndef.get(tag);
        // Enable I/O
        ndef.connect();
        // Write the message
        ndef.writeNdefMessage(message);
```

```
        // Close the connection
        ndef.close();
    }

    private NdefRecord createRecord(String text) throws
UnsupportedEncodingException {
        String lang = "en";
        byte[] textBytes = text.getBytes();
        byte[] langBytes = lang.getBytes("US-ASCII");
        int langLength = langBytes.length;
        int textLength = textBytes.length;
        byte[] payload = new byte[1 + langLength +
textLength];

        // set status byte (see NDEF spec for actual bits)
        payload[0] = (byte) langLength;

        // copy langbytes and textbytes into payload
        System.arraycopy(langBytes, 0, payload, 1,
langLength);
        System.arraycopy(textBytes, 0, payload, 1 +
langLength, textLength);

        NdefRecord recordNFC = new
NdefRecord(NdefRecord.TNF_WELL_KNOWN, NdefRecord.RTD_TEXT,
new byte[0], payload);

        return recordNFC;
    }

}
```

第 20 章 EventBus

http://greenrobot.org/eventbus

在EventBus中主要有以下三个成员：

```
Event：事件，可以自定义为任意对象，类似Message类的作用；
Publisher：事件发布者，可以在任意线程、任意位置发布Event，已发布的Evnet
则由EventBus进行分发；
Subscriber：事件订阅者，接收并处理事件，需要通过register(this)进行注
册，而在类销毁时要使用unregister(this)方法解注册。每个Subscriber可以定
义一个或多个事件处理方法，其方法名可以自定义，但需要添加@Subscribe的注解，
并指明ThreadMode（不写默认为Posting）。
```

1. 添加 EventBus 依赖到项目Gradle文件

Gradle:

```
implementation 'org.greenrobot:eventbus:3.1.1'
```

完整的例子

```
apply plugin: 'com.android.application'

android {
    compileSdkVersion 28
    defaultConfig {
        applicationId "cn.netkiller.eventbus"
        minSdkVersion 26
```

```
        targetSdkVersion 28
        versionCode 1
        versionName "1.0"
        testInstrumentationRunner
"android.support.test.runner.AndroidJUnitRunner"
    }
    buildTypes {
        release {
            minifyEnabled false
            proguardFiles getDefaultProguardFile('proguard-
android.txt'), 'proguard-rules.pro'
        }
    }
}

dependencies {
    implementation fileTree(dir: 'libs', include: ['*.jar'])
    implementation 'com.android.support:appcompat-v7:28.0.0'
    implementation 'com.android.support.constraint:constraint-
layout:1.1.3'
    testImplementation 'junit:junit:4.12'
    androidTestImplementation
'com.android.support.test:runner:1.0.2'
    androidTestImplementation
'com.android.support.test.espresso:espresso-core:3.0.2'
    implementation 'org.greenrobot:eventbus:3.1.1'
}
```

2. 快速开始一个演示例子

操作 EventBus 只需四个步骤

```
1. 注册事件

EventBus.getDefault().register( this );

2. 取消注册

EventBus.getDefault().unregister( this );

3. 订阅事件

        @Subscribe(threadMode = ThreadMode.MAIN)
    public void onMessageEvent(MessageEvent event) {
        Toast.makeText(this, event.message,
Toast.LENGTH_SHORT).show();
    }

4. 发送数据

EventBus.getDefault().post(new MessageEvent("Helloworld"));
```

2.1. 创建 MessageEvent 类

```
package cn.netkiller.eventbus.pojo;

public class MessageEvent {
    public final String message;

    public MessageEvent(String message) {
        this.message = message;
    }
```

```
}
```

2.2. Layout

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.constraint.ConstraintLayout
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"
    tools:context=".MainActivity">

    <TextView
        android:id="@+id/textView"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Hello World!"
        app:layout_constraintLeft_toLeftOf="parent"
        app:layout_constraintRight_toRightOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

    <Button
        android:id="@+id/button"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginStart="8dp"
        android:layout_marginTop="8dp"
        android:layout_marginEnd="8dp"
        android:text="Button"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/textView"
/>

</android.support.constraint.ConstraintLayout>
```

2.3. Activity

```
package cn.netkiller.eventbus;

import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.View;
import android.widget.Toast;

import org.greenrobot.eventbus.EventBus;
import org.greenrobot.eventbus.Subscribe;
import org.greenrobot.eventbus.ThreadMode;

import cn.netkiller.eventbus.pojo.MessageEvent;

public class MainActivity extends AppCompatActivity {

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

        EventBus.getDefault().register(this);

        findViewById(R.id.button).setOnClickListener(new
View.OnClickListener() {
            @Override
            public void onClick(View v) {
                EventBus.getDefault().post(new
MessageEvent("Hello everyone!"));
            }
        });

    }

    @Override
    protected void onDestroy() {
        super.onDestroy();

        //取消注册 , 防止Activity内存泄漏
        EventBus.getDefault().unregister(this);
```

```
    }

    @Subscribe(threadMode = ThreadMode.MAIN)
    public void onMessageEvent(MessageEvent event) {
        Toast.makeText(this, event.message,
Toast.LENGTH_SHORT).show();
    }
}
```

3. Sticky Events

Sticky Events 粘性事件可以理解为Message做了持久化，直到Message被消费为止。无需注册即可发送Message。

下面的例子：在MainActivity发送事件，在StickyActivity里注册并且接收事件

```
A. MainActivity 发送事件:

EventBus.getDefault().postSticky(new
MessageEvent("http://www.netkiller.cn"));

B. StickyActivity 接收事件

1. 注册

EventBus.getDefault().register( this );

2. 事件接收

        @Subscribe(threadMode = ThreadMode.MAIN, sticky = true)
    public void onMessageEvent(MessageEvent event) {
        Toast.makeText(this, event.message,
Toast.LENGTH_SHORT).show();
    }

3. 取消注册

EventBus.getDefault().unregister( this ) ;
```

3.1. MainActivity

Layout

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.constraint.ConstraintLayout
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"
    tools:context=".MainActivity">

    <TextView
        android:id="@+id/textView"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Hello World!"
        app:layout_constraintLeft_toLeftOf="parent"
        app:layout_constraintRight_toRightOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

    <Button
        android:id="@+id/button"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginStart="8dp"
        android:layout_marginTop="8dp"
        android:layout_marginEnd="8dp"
        android:text="Button"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/textView"
/>

</android.support.constraint.ConstraintLayout>
```

MainActivity

```
package cn.netkiller.eventbus;
```

```
import android.content.Intent;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.View;
import android.widget.Toast;

import org.greenrobot.eventbus.EventBus;
import org.greenrobot.eventbus.Subscribe;
import org.greenrobot.eventbus.ThreadMode;

import cn.netkiller.eventbus.pojo.MessageEvent;

public class MainActivity extends AppCompatActivity {

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

        findViewById(R.id.button).setOnClickListener(new
View.OnClickListener() {
            @Override
            public void onClick(View v) {
                EventBus.getDefault().postSticky(new
MessageEvent("Hello everyone!"));
                startActivity(new Intent(MainActivity.this,
StickyActivity.class));
            }
        });

    }

}
```

3.2. StickyActivity

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.constraint.ConstraintLayout
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"
    tools:context=".StickyActivity">

</android.support.constraint.ConstraintLayout>
```

StickyActivity

```
package cn.netkiller.eventbus;

import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.widget.Toast;

import org.greenrobot.eventbus.EventBus;
import org.greenrobot.eventbus.Subscribe;
import org.greenrobot.eventbus.ThreadMode;

import cn.netkiller.eventbus.pojo.MessageEvent;

public class StickyActivity extends AppCompatActivity {

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

        EventBus.getDefault().register(this);
    }

    @Override
    protected void onDestroy() {
        super.onDestroy();
        EventBus.getDefault().unregister(this);
    }

    @Subscribe(threadMode = ThreadMode.MAIN, sticky = true)
```

```
    public void onMessageEvent(MessageEvent event) {
        Toast.makeText(this, event.message,
Toast.LENGTH_SHORT).show();
    }

}
```

3.3. MessageEvent

```
package cn.netkiller.eventbus.pojo;

public class MessageEvent {
    public final String message;

    public MessageEvent(String message) {
        this.message = message;
    }
}
```

3.4. 删除粘性事件

```
MessageEvent stickyEvent =
EventBus.getDefault().getStickyEvent(MessageEvent.class);

// Better check that an event was actually posted before
if(stickyEvent != null) {
        // "Consume" the sticky event
        EventBus.getDefault().removeStickyEvent(stickyEvent);
        // Now do something with it
}
```

4. 线程模型

EventBus 有五种线程模型（ThreadMode）

```
Posting：直接在事件发布者所在线程执行事件处理方法；
Main：直接在主线程中执行事件处理方法（即UI线程），如果发布事件的线程也是主线程，那么事件处理方法会直接被调用，并且未避免ANR，该方法应避免进行耗时操作；
MainOrdered：也是直接在主线程中执行事件处理方法，但与Main方式不同的是，不论发布者所在线程是不是主线程，发布的事件都会进入队列按事件串行顺序依次执行；
BACKGROUND：事件处理方法将在后台线程中被调用。如果发布事件的线程不是主线程，那么事件处理方法将直接在该线程中被调用。如果发布事件的线程是主线程，那么将使用一个单独的后台线程，该线程将按顺序发送所有的事件。
Async：不管发布者的线程是不是主线程，都会开启一个新的线程来执行事件处理方法。如果事件处理方法的执行需要一些时间，例如网络访问，那么就应该使用该模式。为避免触发大量的长时间运行的事件处理方法，EventBus使用了一个线程池来有效地重用已经完成调用订阅者方法的线程以限制并发线程的数量。    后面会通过代码展示五种ThreadMode的工作方式。
```

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.constraint.ConstraintLayout
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"
    tools:context=".ThreadModeActivity">

    <Button
        android:id="@+id/buttonSend"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginStart="8dp"
        android:layout_marginTop="8dp"
        android:layout_marginEnd="8dp"
        android:text="Send"
```

```
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

    <Button
        android:id="@+id/buttonThread"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginStart="8dp"
        android:layout_marginTop="8dp"
        android:layout_marginEnd="8dp"
        android:text="Send Thread"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/buttonSend"
/>
</android.support.constraint.ConstraintLayout>
```

```
package cn.netkiller.eventbus;

import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.util.Log;
import android.view.View;

import org.greenrobot.eventbus.EventBus;
import org.greenrobot.eventbus.Subscribe;
import org.greenrobot.eventbus.ThreadMode;

public class ThreadModeActivity extends AppCompatActivity {

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

        EventBus.getDefault().register(this);

        findViewById(R.id.buttonSend).setOnClickListener(new
View.OnClickListener() {
```

```
            @Override
            public void onClick(View v) {
                Log.d("EventBus Thread : ",
Thread.currentThread().getName());

EventBus.getDefault().post("http://www.netkiller.cn");
            }
        });


findViewById(R.id.buttonThread).setOnClickListener(new
View.OnClickListener() {
            @Override
            public void onClick(View v) {
                new Thread(new Runnable() {
                    @Override
                    public void run() {
                        Log.d("EventBus Thread : ",
Thread.currentThread().getName());

EventBus.getDefault().post("http://www.netkiller.cn");

                    }
                }).start();

            }
        });

    }

    @Subscribe(threadMode = ThreadMode.POSTING)
    public void onMessageEventPostThread(String event) {
        Log.d("EventBus PostThread", "Message:  " + event + "
thread: " + Thread.currentThread().getName());
    }

    @Subscribe(threadMode = ThreadMode.MAIN)
    public void onMessageEventMainThread(String event) {
        Log.d("EventBus MainThread", "Message:  " + event + "
thread: " + Thread.currentThread().getName());
    }

    @Subscribe(threadMode = ThreadMode.MAIN_ORDERED)
    public void onEventMainOrdered(String event) {
        Log.d("EventBus MainOrdered", "Message: " + event + "
```

```
thread:" + Thread.currentThread().getName());
    }

    @Subscribe(threadMode = ThreadMode.BACKGROUND)
    public void onMessageEventBackgroundThread(String event)
{
        Log.d("EventBus BackgroundThread", "Message:  " +
event + "  thread: " + Thread.currentThread().getName());
    }

    @Subscribe(threadMode = ThreadMode.ASYNC)
    public void onMessageEventAsync(String event) {
        Log.d("EventBus Async", "Message:  " + event + "
thread: " + Thread.currentThread().getName());
    }


    @Override
    protected void onDestroy() {
        super.onDestroy();
        EventBus.getDefault().unregister(this);
    }
}
```

在 main 线程中发布消息

```
D/EventBus Thread :: main
D/EventBus MainThread: Message:  http://www.netkiller.cn
thread: main
D/EventBus PostThread: Message:  http://www.netkiller.cn
thread: main
D/EventBus Async: Message:  http://www.netkiller.cn  thread:
pool-1-thread-1
D/EventBus BackgroundThread: Message:  http://www.netkiller.cn
thread: pool-1-thread-2
D/EventBus MainOrdered: Message: http://www.netkiller.cn
thread:main
```

在线程中发布消息

```
D/EventBus Thread :: Thread-2
D/EventBus BackgroundThread: Message:  http://www.netkiller.cn
thread: Thread-2
D/EventBus PostThread: Message:  http://www.netkiller.cn
thread: Thread-2
D/EventBus Async: Message:  http://www.netkiller.cn  thread:
pool-1-thread-2
D/EventBus MainOrdered: Message: http://www.netkiller.cn
thread:main
D/EventBus MainThread: Message:  http://www.netkiller.cn
thread: main
```

5. 配置 EventBus

上面章节中的例子EventBus实例中采用默认方式

```
EventBus.getDefault().register(this);
```

这种方式的获取到的EventBus的都是默认属性，有时候并不能满足我们的要求，这时候我们可以通过EventBusBuilder来个性化配置EventBus的属性。

```
// 创建默认的EventBus对象，相当于EventBus.getDefault()。

EventBus installDefaultEventBus()；
// 添加由EventBus“注释预处理器生成的索引
EventBuilder addIndex(SubscriberInfoIndex index)；
// 默认情况下，EventBus认为事件类有层次结构（订户超类将被通知）
EventBuilder eventInheritance(boolean eventInheritance)；
// 定义一个线程池用于处理后台线程和异步线程分发事件
EventBuilder
executorService(java.util.concurrent.ExecutorService
executorService)；
// 设置忽略订阅索引，即使事件已被设置索引，默认为false
EventBuilder ignoreGeneratedIndex(boolean
ignoreGeneratedIndex)；
// 打印没有订阅消息，默认为true
EventBuilder logNoSubscriberMessages(boolean
logNoSubscriberMessages)；
// 打印订阅异常，默认true
EventBuilder logSubscriberExceptions(boolean
logSubscriberExceptions)；
// 设置发送的的事件在没有订阅者的情况时，EventBus是否保持静默，默认true
EventBuilder sendNoSubscriberEvent(boolean
sendNoSubscriberEvent)；
// 发送分发事件的异常，默认true
```

```
EventBuilder sendSubscriberExceptionEvent(boolean
sendSubscriberExceptionEvent)：
// 在3.0以前，接收处理事件的方法名以onEvent开头，方法名称验证避免不是以
此开头，启用严格的方法验证（默认：false）
EventBuilder strictMethodVerification(java.lang.Class<?>
clazz)
// 如果onEvent***方法出现异常，是否将此异常分发给订阅者（默认：false）
EventBuilder throwSubscriberException(boolean
throwSubscriberException)
```

我的实例参考

```
EventBus eventBus = EventBus.builder().eventInheritance(true)
    .ignoreGeneratedIndex(false)
    .logNoSubscriberMessages(true)
    .logSubscriberExceptions(false)
    .sendNoSubscriberEvent(true)
    .sendSubscriberExceptionEvent(true)
    .throwSubscriberException(false)
    .strictMethodVerification(true)
    .build();
eventBus.register(this);
```

6. 事件优先级

priority 数值越大优先级又高

```
        // MainActivity
        @Subscribe(priority = 2)
    public void onMessageEvent(MessageEvent event) {
        Toast.makeText(this, event.message,
Toast.LENGTH_SHORT).show();
    }

        // SecondActivity
        @Subscribe(priority = 1)
    public void onMessageSecondEvent(MessageEvent event) {
        Toast.makeText(this, event.message,
Toast.LENGTH_SHORT).show();
    }
```

时间拦截，MainActivity 收到信息后调用 EventBus.getDefault().cancelEventDelivery(event); 之后所有订阅将收不到信息。

```
        // MainActivity
        @Subscribe(priority = 2)
    public void onMessageEvent(MessageEvent event) {
        Toast.makeText(this, event.message,
Toast.LENGTH_SHORT).show();
        EventBus.getDefault().cancelEventDelivery(event);
    }

        // SecondActivity
        @Subscribe(priority = 1)
    public void onMessageSecondEvent(MessageEvent event) {
        Toast.makeText(this, event.message,
```

```
Toast.LENGTH_SHORT).show();
    }
```

7. 捕获异常事件

在 init() 中加入你的业务逻辑，根据需要，在特定的情况下使用 throw new Exception("异常信息"); 抛出异常。异常会被 hrowableFailureEvent(ThrowableFailureEvent event) 捕获到。

```
package cn.netkiller.eventbus;

import android.content.Intent;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.util.Log;
import android.view.View;
import android.widget.Toast;

import org.greenrobot.eventbus.EventBus;
import org.greenrobot.eventbus.Subscribe;
import org.greenrobot.eventbus.ThreadMode;
import org.greenrobot.eventbus.util.AsyncExecutor;
import org.greenrobot.eventbus.util.ThrowableFailureEvent;

import cn.netkiller.eventbus.pojo.MessageEvent;

public class MainActivity extends AppCompatActivity {

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

        EventBus.getDefault().register(this);

        findViewById(R.id.button).setOnClickListener(new
View.OnClickListener() {
            @Override
            public void onClick(View v) {

                AsyncExecutor.create().execute(
                        new AsyncExecutor.RunnableEx() {
```

```
                                @Override
                                public void run() throws
Exception {
                                    init();

EventBus.getDefault().post(new MessageEvent("Hello
everyone!"));
                                }
                            }
                    );
                }
            });
    }

    @Override
    protected void onDestroy() {
        super.onDestroy();
        EventBus.getDefault().unregister(this);
    }

    @Subscribe(threadMode = ThreadMode.MAIN)
    public void onMessageEvent(MessageEvent event) {
        Toast.makeText(this, event.message,
Toast.LENGTH_SHORT).show();
    }

    public void init() throws Exception {
        // ...
        throw new Exception("实际发送异常");
    }

    @Subscribe(threadMode = ThreadMode.MAIN)
    public void hrowableFailureEvent(ThrowableFailureEvent
event) {
        Log.d("EventBus", "hrowableFailureEvent: " +
event.getThrowable().getMessage());
        Toast.makeText(this,
event.getThrowable().getMessage(),
Toast.LENGTH_SHORT).show();
    }

}
```

第 21 章 图形开发

1. Paint

```
paint = new Paint();
paint.setColor(Color.GREEN);
paint.setColor(0xFF00F8C1);
paint.setStrokeWidth(1);
```

2. AnimationDrawable

```
    private int getTotalDuration(AnimationDrawable
animationDrawable) {
        int totalDuration = 0;
        for (int i = 0; i <
animationDrawable.getNumberOfFrames(); i++) {
            totalDuration +=
animationDrawable.getDuration(i);
        }
        return totalDuration;
    }
```

第 22 章 Android Things

1. GPIO

配置权限

```
<uses-permission
android:name="com.google.android.things.permission.USE_PERIPHER
AL_IO" />
<uses-permission
android:name="com.google.android.things.permission.MANAGE_INPUT
_DRIVERS" />
```

第 23 章 Android MQTT

1. build.gradle 添加依赖包

```
implementation group: 'org.eclipse.paho', name:
'org.eclipse.paho.mqttv5.client', version: '1.2.5'
implementation group: 'org.eclipse.paho', name:
'org.eclipse.paho.android.service', version: '1.1.1', ext:
'pom'
```

提示

2. AndroidManifest.xml

```
 <uses-permission android:name="android.permission.WAKE_LOCK"
/>
 <uses-permission
android:name="android.permission.READ_EXTERNAL_STORAGE" />
 <uses-permission
android:name="android.permission.WRITE_EXTERNAL_STORAGE" />
 <uses-permission
android:name="android.permission.ACCESS_NETWORK_STATE" />
 <uses-permission android:name="android.permission.INTERNET" />
```

3. Android Mqtt v5 例子

```
package cn.netkiller.ropeway.service;

import android.app.Service;
import android.content.Intent;
import android.os.IBinder;
import android.util.Log;

import org.eclipse.paho.mqttv5.client.IMqttDeliveryToken;
import org.eclipse.paho.mqttv5.client.IMqttToken;
import org.eclipse.paho.mqttv5.client.MqttAsyncClient;
import org.eclipse.paho.mqttv5.client.MqttCallback;
import org.eclipse.paho.mqttv5.client.MqttConnectionOptions;
import org.eclipse.paho.mqttv5.client.MqttDisconnectResponse;
import
org.eclipse.paho.mqttv5.client.persist.MemoryPersistence;
import org.eclipse.paho.mqttv5.common.MqttException;
import org.eclipse.paho.mqttv5.common.MqttMessage;
import org.eclipse.paho.mqttv5.common.packet.MqttProperties;

public class MyService extends Service {
    MqttAsyncClient mqttAsyncClient;
    IMqttToken token;

    String topic = "/netkiller/test";
    String content = "Helloworld!!!";
    int qos = 2;
    String broker = "tcp://broker.emqx.io:1883";
    String clientId = "JavaSample" +
System.currentTimeMillis();

    public MyService() {
        try {
            MemoryPersistence persistence = new
MemoryPersistence();
            mqttAsyncClient = new MqttAsyncClient(broker,
clientId, persistence);
        } catch (MqttException me) {
            System.out.println("reason " +
```

```
me.getReasonCode());
            System.out.println("msg " + me.getMessage());
            System.out.println("loc " +
me.getLocalizedMessage());
            System.out.println("cause " + me.getCause());
            System.out.println("excep " + me);
        }
    }

    @Override
    public IBinder onBind(Intent intent) {
        // TODO: Return the communication channel to the
service.
        throw new UnsupportedOperationException("Not yet
implemented");
    }

    @Override
    public void onCreate() {
        super.onCreate();
        Log.d("Service", "onCreate() executed");
        try {
            MqttConnectionOptions mqttConnectionOptions = new
MqttConnectionOptions();
            mqttConnectionOptions.setCleanStart(false);

mqttConnectionOptions.setAutomaticReconnect(true);

            Log.d("Service", "Connecting to broker: " +
broker);

            token =
mqttAsyncClient.connect(mqttConnectionOptions);
            token.waitForCompletion();
            if (token.isComplete()) {
                Log.d("Service", "Connected");

mqttAsyncClient.subscribe("/netkiller/message", qos);
            }

        } catch (MqttException e) {
            throw new RuntimeException(e);
        }

        mqttAsyncClient.setCallback(new MqttCallback() {
```

```
            @Override
            public void disconnected(MqttDisconnectResponse
disconnectResponse) {

            }

            @Override
            public void mqttErrorOccurred(MqttException
exception) {

            }

            @Override
            public void messageArrived(String topic,
MqttMessage message) throws Exception {
                String msg = new
String(message.getPayload());
                Log.d("Service", String.format("接收消息 Id:%s,
Topic: %s, QoS: %s, Message: %s, ", message.getId(), topic,
message.getQos(), message.toString()));
            }

            @Override
            public void deliveryComplete(IMqttToken token) {

            }

            @Override
            public void connectComplete(boolean reconnect,
String serverURI) {
//                if (reconnect) {
//                    try {
//
mqttAsyncClient.subscribe("/netkiller/message", qos);
//                    } catch (MqttException e) {
//                        throw new RuntimeException(e);
//                    }
//                }
            }

            @Override
            public void authPacketArrived(int reasonCode,
MqttProperties properties) {
```

```
            }


            public void deliveryComplete(IMqttDeliveryToken
arg0) {
                try {
                    System.out.println(arg0.getMessage());
                } catch (MqttException e1) {
                    e1.printStackTrace();
                }

            }


            public void connectionLost(Throwable err) {
                System.out.println("连接丢失");
                System.out.println(err.getMessage());

            }
        });


    }

    @Override
    public int onStartCommand(Intent intent, int flags, int
startId) {
        Log.d("Service", "onStartCommand() executed");

        try {
            Log.d("Service", "Publishing message: " +
content);
            MqttMessage message = new
MqttMessage(content.getBytes());
            message.setQos(qos);
            token = mqttAsyncClient.publish(topic, message);
            token.waitForCompletion();
        } catch (MqttException e) {
            throw new RuntimeException(e);
        }

        return super.onStartCommand(intent, flags, startId);
    }

    @Override
```

```
    public void onDestroy() {
        super.onDestroy();
        try {
            if (mqttAsyncClient.isConnected()) {
                mqttAsyncClient.close();
                Log.d("Service", "Close client.");
            }
        } catch (MqttException e) {
            Log.d("Service", "Disconnected");
        }
        Log.d("Service", "onDestroy() executed");
    }
}
```

第 24 章 安卓开发版

1. rk3568

1.1. 声卡

```
rk3568_r:/storage/emulated/0 # cat /proc/asound/cards
 0 [rockchiphdmi   ]: rockchip_hdmi - rockchip,hdmi
                      rockchip,hdmi
 1 [rockchiprk809co]: rockchip_rk809- - rockchip,rk809-codec
                      rockchip,rk809-codec
 2 [UR22C          ]: USB-Audio - Steinberg UR22C
                      Yamaha Corporation Steinberg UR22C at
usb-xhci-hcd.5.auto-1, high speed
```

```
rk3568_r:/storage/emulated/0 # cat /proc/asound/devices
  2: [ 0- 0]: digital audio playback
  3: [ 0]   : control
  4: [ 1- 0]: digital audio playback
  5: [ 1- 0]: digital audio capture
  6: [ 1]   : control
  7: [ 2- 0]: digital audio playback
  8: [ 2- 0]: digital audio capture
  9: [ 2]   : control
 10: [ 2- 0]: raw midi
 33:        : timer
```

```
rk3568_r:/storage/emulated/0 # cat /proc/asound/pcm
```

```
00-00: fe400000.i2s-i2s-hifi i2s-hifi-0 : fe400000.i2s-i2s-hifi
i2s-hifi-0 : playback 1
01-00: fe410000.i2s-rk817-hifi rk817-hifi-0 : fe410000.i2s-
rk817-hifi rk817-hifi-0 : playback 1 : capture 1
02-00: USB Audio : USB Audio : playback 1 : capture 1
```

第 25 章 杂项

1. Sleep

```
try {
    Thread.sleep(10000);
} catch (InterruptedException e) {
    Log.e("Location", e.getMessage());
}
```

2. Caused by: java.net.UnknownServiceException: CLEARTEXT communication to 47.100.253.187 not permitted by network security policy

在AndroidManifest.xml的文件的application节点中增加 android:usesCleartextTraffic="true"

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

    <uses-permission
android:name="android.permission.ACCESS_COARSE_LOCATION" />
    <uses-permission
android:name="android.permission.ACCESS_FINE_LOCATION" />
    <uses-permission
android:name="android.permission.ACCESS_WIFI_STATE" />
    <uses-permission
android:name="android.permission.ACCESS_NETWORK_STATE" />
    <uses-permission
android:name="android.permission.CHANGE_WIFI_STATE" />
    <uses-permission android:name="android.permission.INTERNET"
/>

    <application
        android:allowBackup="true"

android:dataExtractionRules="@xml/data_extraction_rules"
        android:fullBackupContent="@xml/backup_rules"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/Theme.Ropeway"
        android:usesCleartextTraffic="true"
```

```
        tools:targetApi="31">
        <service
            android:name=".service.RopewayService"
            android:enabled="true"
            android:exported="true"></service>
        <service
            android:name=".service.BroadcastService"
            android:enabled="true"
            android:exported="true" />

        <activity
            android:name=".MainActivity"
            android:exported="true"
            android:label="@string/app_name">
            <intent-filter>
                <action
android:name="android.intent.action.MAIN" />

                <category
android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>
    <meta-data
        android:name="apiUrl"
        android:value="http://47.100.25.18:8000" />
</manifest>
```

3. 设计模式

3.1. 单例模式

```
package cn.netkiller.voice;

import android.media.MediaRecorder;
import android.os.Environment;
import android.util.Log;

import java.io.File;
import java.io.IOException;
import java.text.SimpleDateFormat;
import java.util.Date;

public class Audio {

    private boolean isRecord = false;

    private MediaRecorder mediaRecorder;
    private String filename;

    private Audio() {

    }

    private static Audio instance;

    public synchronized static Audio getInstance() {
        if (instance == null)
            instance = new Audio();
        return instance;
    }

    public String getFilename() {
        return filename;
    }

    public void start() {
```

```
        if (mediaRecorder == null) {

            String path =
Environment.getExternalStorageDirectory().getPath();
            String folder = new SimpleDateFormat("yyyy-MM-
dd").format(new Date());
            String name = new
SimpleDateFormat("hhmmss").format(new Date());
            new File(path, folder).mkdirs();

            filename = String.format("%s/%s/%s.3gp", path,
folder, name);
            Log.e("Voice", "voice path " + filename);

            try {

                mediaRecorder = new MediaRecorder();

mediaRecorder.setAudioSource(MediaRecorder.AudioSource.MIC);

mediaRecorder.setOutputFormat(MediaRecorder.OutputFormat.DEFA
ULT);

mediaRecorder.setAudioEncoder(MediaRecorder.AudioEncoder.DEFA
ULT);
                mediaRecorder.setOutputFile(filename);
                mediaRecorder.prepare();
                mediaRecorder.start();

                isRecord = true;

            } catch (IOException ex) {
                ex.printStackTrace();

            }
        }

    }

    public void stop() {
        if (mediaRecorder != null && isRecord) {
            System.out.println("stopRecord");
            isRecord = false;
            mediaRecorder.stop();
```

```
            mediaRecorder.release();
            mediaRecorder = null;
        }
    }

}
```

4. Android OS 包

4.1. 进程ID

```
android.os.Process.myTid()
```

4.2. handler

```
        handler.postDelayed(() -> {
        }, 1000);
```

5. fastjson android

```
implementation 'com.alibaba:fastjson:2.0.20.android'
```

5.1. 对象转字符串

```
String json = JSON.toJSONString(user); //序列化
```

5.2. JsonObject 转对象

对象转JsonObject

```
User user =JSON.parseObject(json,User.class); //反序列化
JSONObject jsonObject=(JSONObject)JSON.toJSON(user);
jsonObject.getIntValue("id");
```

jsonObject 转 Java Object

```
User user=JSON.toJavaObject(jsonObject, User.class);
```

5.3. 字符串 与 json 互转

json 转 字符串

```
String jsonString=JSON.toJSONString(jsonObject);
```

字符串 转 json

```
JSONObject jsonObject=JSON.parseObject(jsonString);
jsonObject.getString("name");
```

5.4. json 转 数组

```
JSONArray
jArray=JSON.parseArray(JSON.toJSONString(userList));
```

5.5. JSON数组转List

```
List<Map> listMaps =
JSONArray.parseArray(JSON.toJSONString(data),Map.class);
List<Map> mapsList =
JSONObject.parseArray(JSON.toJSONString(data), Map.class);
```

5.6. Map 与 Json 互转

Json 转 map

```
Map<String,Object> maps =
JSONObject.parseObject(json2,Map.class);
```

Map转JSON

```
JSONObject jsonObject =
JSONObject.parseObject(JSON.toJSONString(maps));
```

6. Butter Knife

http://jakewharton.github.io/butterknife/

第 26 章 FAQ

1. java.net.UnknownServiceException: CLEARTEXT communication to 192.168.0.185 not permitted by network security policy

okhttp 默认使用 https 链接服务器，如果使用 http 会抛出现上面的异常

```
if (!Platform.get().isCleartextTrafficPermitted(host)) {
      throw new RouteException(new UnknownServiceException(
          "CLEARTEXT communication to " + host + " not
permitted by network security policy"));
}
```

创建文件 res/xml/network_security_config.xml 内容如下

```
neo@MacBook-Pro ~/AndroidStudioProjects/okhttp % cat
app/src/main/res/xml/network_security_config.xml
<?xml version="1.0" encoding="utf-8"?>
<network-security-config>
    <base-config cleartextTrafficPermitted="true" />
</network-security-config>
```

再 app/src/main/AndroidManifest.xml 文件中增加 android:networkSecurityConfig="@xml/network_security_config"

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

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/AppTheme">
        <activity
            android:name=".MainActivity"
            android:label="@string/app_name"

android:networkSecurityConfig="@xml/network_security_config">
            <intent-filter>
                <action
android:name="android.intent.action.MAIN" />

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

    <uses-permission android:name="android.permission.INTERNET"
/>
</manifest>
```

2. Caused by: android.os.NetworkOnMainThreadException

主线程不能访问网络，在访问网络的代码前面添加如下代码即可：

```
StrictMode.ThreadPolicy policy= new
StrictMode.ThreadPolicy.Builder().permitAll().build();
StrictMode.setThreadPolicy(policy);
```

或者写在 setContentView(R.layout.activity_main); 后面

另一种方式是在线程中执行

```
       new Thread(new Runnable() {
            @Override
            public void run() {

                try {
                    String json =
get("http://192.168.0.185:8080/member/json");
                } catch (IOException e) {
                    e.printStackTrace();
                }

            }
        }).start();
```

3. java.lang.IllegalStateException: Player is accessed on the wrong thread.

```
// 在 Service 中定义一个 Handler
private Handler mHandler = new
Handler(Looper.getMainLooper());

// 在 Service 中定义一个 Runnable 对象
private Runnable mRunnable = new Runnable() {
    @Override
    public void run() {
        // 在这里执行与 UI 相关的操作
    }
};

// 在 Service 中使用 Handler 将 Runnable 对象发送到 UI 线程的消息队
列中
mHandler.post(mRunnable);
```

第 27 章 讯飞云

1. AIUI

```
// 写入文本
//                byte[] content= "你好".getBytes();
//                String params = "data_type=text";
//                AIUIMessage msg = new
AIUIMessage(AIUIConstant.CMD_WRITE, 0, 0, "tag=write_data_1",
content);
//                mAIUIAgent.sendMessage(msg);

                AIUIMessage aiuiMessage = new AIUIMessage(0,
0, 0, "", null);
                aiuiMessage.msgType = AIUIConstant.CMD_WRITE;
                aiuiMessage.arg1 = 0;
                aiuiMessage.arg2 = 0;
                // 在输入参数中设置tag，则对应结果中也将携带该tag，可
用于关联输入输出
                aiuiMessage.params =
"data_type=text,tag=text-tag";
                aiuiMessage.data = "天
气".getBytes(StandardCharsets.UTF_8);
                mAIUIAgent.sendMessage(aiuiMessage);
```

1.1. AIUIPlayer

```
package cn.netkiller.aiui;

import android.content.Context;
import android.util.Log;
```

```
import androidx.annotation.NonNull;

import com.iflytek.aiui.player.common.data.MetaItem;
import com.iflytek.aiui.player.core.AIUIPlayer;
import com.iflytek.aiui.player.core.PlayState;
import com.iflytek.aiui.player.core.PlayerListener;
import com.iflytek.aiui.player.players.KuwoMusicRemote;

import org.json.JSONArray;
import org.json.JSONException;
import org.json.JSONObject;

import kotlin.Unit;
import kotlin.jvm.functions.Function0;
import kotlin.jvm.functions.Function2;

public class MusicSkillComponent {
    private static final String TAG =
MusicSkillComponent.class.getSimpleName();
    private static MusicSkillComponent musicSkillComponent =
null;
    private static String kuwoParams = null;
    private static int playListIndex = 0; //当前播放歌曲序号
    private static JSONArray mPlayList = null; //播放列表
    private static AIUIPlayer mAIUIPlayer = null;
    private static KuwoMusicRemote kuwoRemote = null;
    private final Context context;
    private final PlayerListener mPlayListener = new
PlayerListener() {
        @Override
        public void onMediaUrl(@NonNull JSONObject
jsonObject) {
            Log.d(TAG, "onMediaUrl: " + jsonObject);
        }

        @Override
        public void onPlayerReady() {
            Log.i(TAG, "onPlayerReady()");
        }

        @Override
        public void onStateChange(@NonNull PlayState state) {
            Log.i(TAG, "onStateChange: PlayState is" +
state);
            switch (state) {
```

```
                case READY:
                    Log.d(TAG, "AIUIPlayer 播放准备就绪");
                    break;
                case PLAYING:
                    Log.d(TAG, "播放");
                    break;
                case PAUSED:
                    Log.d(TAG, "停止");
                    break;
                case LOADING:
                    Log.d(TAG, "音乐加载中……");
                    break;
                case COMPLETE:
                    Log.d(TAG, "继续");
                    startPlayMusic();
                    break;
                case IDLE:
                    long currentPosition =
mAIUIPlayer.getCurrentPosition();
                    Log.d(TAG,
String.format("CurrentPosition: %l", currentPosition));
                    break;
                case ERROR:
                    Log.d(TAG, "播放出错");
                    break;
                default:
                    break;
            }
        }

        @Override
        public void onMediaChange(@NonNull MetaItem metaItem)
{
            String songName = metaItem.getTitle();      //歌名
            String author = metaItem.getAuthor();      //作者
            playListIndex = getPlayIndex(songName);
            Log.d(TAG, "onMediaChange: " + metaItem);
        }

        @Override
        public void onError(int code, @NonNull String info) {
            Log.e(TAG, "onError 播放出错: " + code + " , 错误信息
为: " + info);
            // 真实错误码需要从 info 中解析   "track link failed
code: 40006 description:"
```

```
            if (info.isEmpty()) {
                if (code == 200001) {
                    Log.d(TAG, "歌曲\"" + "" + "\"播放出错: " +
code + "\nINFO: 产品未通过酷我验收,仅支持获取奇数id资源\n");
                    if (!mAIUIPlayer.next()) {
                    }
                }
            }
//            else {
//                String[] split = info.split(" ");
//                String errCode = split[4];
//                switch (errCode) {
//                    case "40000":
//                        Log.d(TAG, "播放出错: " + errCode +
"\n设备音乐装机量超限");
//                        break;
//                    case "40001":
//                        Log.d(TAG, "播放出错: " + errCode +
"\n设备激活限制");
//                        break;
//                    case "40004":
//                        Log.d(TAG, "播放出错: " + errCode +
"\n设备音乐未激活");
//                        break;
//                    case "40006":
//                        Log.d(TAG, "播放出错: " + errCode +
"\n酷我用户账号未登陆绑定");
//                        break;
//                    case "40007":
//                        Log.d(TAG, "播放出错: " + errCode +
"\n厂商ID与设备当前绑定用户ID不一致");
//                        break;
//                    default:
//                        Log.d(TAG, "播放出错: " + errCode + "
错误码查询: https://www.yuque.com/iflyaiui/zzoolv/rft07m");
//                        break;
//                }
//            }
        }

        @Override
        public void onPlayerRelease() {

        }
    };
```

```
    public MusicSkillComponent(Context context) {
        this.context = context;
        initSDK();
    }

    public synchronized static MusicSkillComponent
getInstance(Context context) {
        if (musicSkillComponent == null) {
            musicSkillComponent = new
MusicSkillComponent(context);
        }
        return musicSkillComponent;
    }

    public boolean previous() {
        if (mAIUIPlayer != null) {
            return mAIUIPlayer.previous();
        }
        return false;
    }

    public boolean next() {
        if (mAIUIPlayer != null) {
            return mAIUIPlayer.next();
        }
        return false;
    }

    public void pause() {
        if (mAIUIPlayer == null) {
            return;
        }
        if (mAIUIPlayer.getCurrentState() ==
PlayState.PLAYING) {
            mAIUIPlayer.pause();
        }
    }

    public void resume() {
        if (mAIUIPlayer == null) {
            return;
        }
        if (mAIUIPlayer.getCurrentState() ==
PlayState.PAUSED) {
```

```
            mAIUIPlayer.resume();
        }
    }

    public boolean play(@NonNull JSONObject object) {

//        Log.d(TAG, object.toString());

        try {

            JSONArray musics = object.getJSONArray("result");
            if (null != musics) {
                mPlayList = musics;
                playListIndex = 0;
            }
            return startPlayMusic();

        } catch (JSONException e) {
            e.printStackTrace();
        }

        return false;
    }

    public void initSDK() {
        //TODO 开发者需要实现生成sn的代码，参考：
https://www.yuque.com/iflyaiui/zzoolv/tgftb5
        //注意事项1：sn每台设备需要唯一！！！！WakeupEngine的sn和
AIUI的sn要一致
        //注意事项2：获取的值要保持稳定，否则会重复授权，浪费授权量
        String serialNumber = "test";
        String appId = "c84e1ddb";
        String appKey = "7a1583c3190b83fe4d62573ee9cfbfc1";

        //TODO 设置酷我音乐SDK设置相关参数，appid和appkey请使用自己
的进行开发，并且与aiui.cfg一致
        kuwoParams = "appId=" + appId + ",appKey=" + appKey +
"," + "serialNumber=" + serialNumber + ",deviceModel=" +
serialNumber + ",userId=" + serialNumber;


        if (null == kuwoRemote) {
            kuwoRemote = new KuwoMusicRemote(kuwoParams);
            // 酷我SDK日志开关 ：true 打开，false 关闭
            kuwoRemote.setDebug(false);
```

```
            if (null != kuwoRemote) {
                Log.i(TAG, "KuwoRemote 初始化成功");
            }

        }

        if (null == mAIUIPlayer) {
            mAIUIPlayer = new AIUIPlayer(context,
kuwoParams);
            // 播放前焦点占用设置
            mAIUIPlayer.setParameter("customAudioFocus",
"true");
            // 回调信息设置
            mAIUIPlayer.addListener(mPlayListener);
            // AIUIPlayer SDK调试日志设置 : true 打开, false 关闭
            mAIUIPlayer.setDebug(false);
            // 初始化播放器
            mAIUIPlayer.initialize();
            Log.i(TAG, "AIUIPlayer 初始化成功");

        }
    }

    public void release() {
        mPlayList = null;
        if (null != mAIUIPlayer) {
            mAIUIPlayer.release();
            mAIUIPlayer = null;
        }
        if (null != kuwoRemote) {
            kuwoRemote.destroy();
            kuwoRemote = null;
        }
    }

    private void activate() {
//        if (isActivated) {
////            showToast("当前设备已激活酷我音乐");
//            return;
//        }
        kuwoRemote.active(() -> {
            Log.i(TAG, "激活成功");
            return null;
        }, (errCode, errInfo) -> {
```

```
            Log.i(TAG, "激活失败，错误码为：" + errCode + " ,信息为： " + errInfo);
            return null;
        });
    }

    private void login() {
        //酷我不强制用户登陆，开发者自己实现登陆代码，可参考下方酷狗音乐代码，改一下接口
//        Intent intent = new Intent(KuwoDemo.this, LoginActivity.class);
//        startActivityForResult(intent, 3);
    }

    private void logout() {
        kuwoRemote.logout(new Function0<Unit>() {
            @Override
            public Unit invoke() {
                Log.i(TAG, "酷我账号退出成功");
//                LoginBtn.setText("手机登陆");
                return null;
            }
        }, new Function2<Integer, String, Unit>() {
            @Override
            public Unit invoke(Integer errCode, String errInfo) {
                Log.i(TAG, "酷我账号退出失败，错误码为：" + errCode + " ,信息为： " + errInfo);
                return null;
            }
        });
    }

    public void stop() {
        if (mAIUIPlayer != null) {
            mAIUIPlayer.stop();
        }
    }

    private boolean startPlayMusic() {
        if (null == mPlayList || mPlayList.length() == 0) {
            Log.w(TAG, "播放列表为空");
            return false;
        }
        Log.i(TAG, "mPlayList is: " + mPlayList.toString());
```

```
        mAIUIPlayer.reset();
        return mAIUIPlayer.play(mPlayList, "musicX", "",
false, playListIndex);
    }

    //构建虚假播放信息测试AIUIPlayer SDK播放是否可以正常调用
//    private void mockPlayMusic() {
//        try {
//            JSONArray musiclist = new JSONArray();
//            JSONObject music = new JSONObject();
//            music.put("source", "kuwo");
//            music.put("songname", "天地龙鳞");
//            music.put("itemid", "353833243");
//            musiclist.put(0, music);
//            mPlayList = musiclist;
//        } catch (JSONException e) {
//            e.printStackTrace();
//        }
//
//        if (null == mPlayList || mPlayList.length() == 0) {
////            showToast("播放列表为空");
//            return;
//        }
//        Log.i(TAG, "mPlayList is:\n" +
mPlayList.toString());
//        mAIUIPlayer.reset();
//        mAIUIPlayer.play(mPlayList, "musicX", "", false,
playListIndex);
//    }

    // 获取当前播放下标
    private int getPlayIndex(String songName) {
        if (mPlayList != null) {
            try {
                String playData = null;
                for (int i = 0; i < mPlayList.length(); i++)
{
                    playData =
mPlayList.getJSONObject(i).toString();
                    if (playData.contains(songName)) {
                        return i;
                    }
                }
            } catch (JSONException e) {
                e.printStackTrace();
```

```
                return 0;
            }
        }
        return 0;
    }
}
```

1.2. 酷我音乐

获取音乐URL

通过 itemId 获取 audio URL

```
    kuwoRemote.getAudioUrl("26383685", "128kmp3", new
Function1<AudioUrl, Unit>() {
        @Override
        public Unit invoke(AudioUrl audioUrl) {
            Log.d(TAG, audioUrl.toString());
            return null;
        }
    }, new Function2<Integer, String, Unit>() {
        @Override
        public Unit invoke(Integer code, String msg) {
            Log.d(TAG, "Code: " + code + ", Msg: " + msg);
            return null;
        }
    });
```

日志输出结果

```
AudioUrl(expiretime=, itemid=26383685, source=kuwo,
audiopath=http://other.player.ri01.sycdn.kuwo.cn/6dcbdb125c72df
ff3978fe29ab50fdd4/64eeef6f/resource/n2/27/48/2060696053.mp3)
```