系统app

关键字：Android系统 ROM，系统app，系统应用，分布

android系统app定制

adb pull /data/system/packages.xml

Rom精简 <http://club.huawei.com/thread-4643339-1-1.html>

http://tcgz.leanote.com/post/Android%E7%B3%BB%E7%BB%9F%E8%87%AA%E5%B8%A6app%E5%8A%9F%E8%83%BD%E5%8F%8A%E5%88%A0%E5%87%8F%E8%AF%B4%E6%98%8E

# Packages/apps

# Img大包

## /system/priv-app

### 目录

原生：

DefaultContainerService：应用服务，负责实现APK等相关资源文件在内部或外部存储器上的存储工作。

**DownloadProvider**：下载管理器:通知ui

ExternalStorageProvider

FusedLocation

InputDevices:只有一个广播接收器，没有任何逻辑处理，有键盘映射

ManagedProvisioning

MediaProvider: 媒体数据库的

MmsService: 发送彩信时，终端将建立短连接, 容易看出MmsService是运行在Phone进程中的

MusicFX: 音频控制面板

OneTimeInitializer: 首次启动时用于安装谷歌应用的，删除之后双清会卡在开机向导,无ui

ProxyHandler

**Settings**

SettingsProvider

SharedStorageBackup: 共享备份存储

Shell

StressTest

**SystemUI**

TeleService: http://blog.csdn.net/yihongyuelan/article/details/19930861

Telecom

TelephonyProvider

VpnDialogs

**WallpaperCropper**

预装：

GoogleInput

Google\_pdf\_viewer

## /system/app

BasicDreams

Browser

CaptivePortalLogin

CertInstaller

**Crystal\_Sky\_Launcher**

DJIService

DocumentsUI

DownloadProviderUi

Exchange2：邮件

GPSRecoder：自定义的，cpp代码

Galaxy4：壁纸？

Gallery2

HTMLViewer

HoloSpiralWallpaper：很丑的动态壁纸

KeyChain：http://blog.csdn.net/innost/article/details/44199503

LatinIME

LiveWallpapers

LiveWallpapersPicker

MediaFloat：？

NoiseField？

OpenWnn:另外的输入法

PacProcessor: 链接电脑的时候用的～但是告诉你中国连接电脑用不到PacProcessor.apk！

PackageInstaller

PhaseBeam：动态壁纸：光束 PhaseBeam

PhotoTable：照片屏幕保护程序

PicoTts: 这个是Android原生就有的语音引擎，中文语音似乎google暂时还没有,我们有必要加么

Provision:开机向导

Rk4kVideoPlayer

RkExplorer

**SystemSetup**

**SystemUpgrade**

UserDictionaryProvider

VisualizationWallpapers

WinStartService？

Webview?

/system/vendor

## /system/framework

## 定制部分

**DownloadProvider**

**Settings**

**SystemUI**

**WallpaperCropper**

**文本查看器**

**LatinIME：输入法太丑了。。**

Tts？

# 详解

## DefaultContainerService

源码路径：frameworks/base/packages/

一个类无UI

|  |
| --- |
| //创建DefaultContainerService类的ComponentName对象  static final ComponentName DEFAULT\_CONTAINER\_COMPONENT = new ComponentName(DEFAULT\_CONTAINER\_PACKAGE,  "com.android.defcontainer.DefaultContainerService");  private boolean connectToService() {  //设置Intent  Intent service = new Intent().setComponent(DEFAULT\_CONTAINER\_COMPONENT);  Process.setThreadPriority(Process.THREAD\_PRIORITY\_DEFAULT);  //通过bindServiceAsUser来启动DefaultContainerService服务  if (mContext.bindServiceAsUser(service, mDefContainerConn,Context.BIND\_AUTO\_CREATE, UserHandle.OWNER)) {  mBound = true;  return true;  }  Process.setThreadPriority(Process.THREAD\_PRIORITY\_BACKGROUND);  return false;  } |

我们安装一个应用的时候还需要另外一个apk来提供服务，这个apk就是DefaultContainerService.apk

## DownloadProvider

Download的源码编译分为两个部分，一个是DownloadProvider.apk, 一个是DownloadProviderUi.apk.

这两个apk的源码分别位于

packages/providers/DownloadProvider/ui/src

packages/providers/DownloadProvider/src

<http://blog.csdn.net/chaoy1116/article/details/22384841>

## ExternalStorageProvider

frameworks/base/packages/ExternalStorageProvider

root@zs600b:/ # logcat | grep externalstorage

I/ActivityManagerService( 486): Killing 21621:com.android.externalstorage/u0a5 (adj 15): empty for 16356s

无UI，就是Provider而已

<http://blog.csdn.net/snail201211/article/details/77979988>

## FusedLocation

frameworks/base/packages/FusedLocation

融合位置，

<https://github.com/kesenhoo/android-training-course-in-chinese/blob/master/location/retrieve-current.md>

## ManagedProvisioning

配置管理器？用来干嘛的？

<http://blog.csdn.net/Liu1314you/article/details/52028823、>

工作手机的安全模式？

## MediaProvider

媒体数据库的封装类，代码量比较大（四千多行），功能比较复杂，但总的来说就是创建数据库，对外提供URI以实现对数据库的增删改查功能；

<http://gqdy365.iteye.com/blog/2150883>

## **MmsService**

Android中彩信相关的应用为MmsService，我们看看它AndroidManifest.xml中的部分片段：

<application android:label="MmsService"

android:process="com.android.phone"

android:usesCleartextTraffic="true">

<service android:name=".MmsService"

android:enabled="true"

android:exported="true"/>

</application>

容易看出MmsService是运行在Phone进程中的。在这篇博客中，我们不深入研究彩信服务的启动和收发彩信的过程，主要看看彩信如何建立和释放短连接。

## wallpapercropper

frameworks/base/packages/

## DocumentsUI

frameworks/base/packages/

小米继承了迅雷的下载引擎

<http://www.cnblogs.com/zqlxtt/p/4451949.html>

Android中Download由三个部分组成：

1.DocumentsUI -----> /[frameworks](http://androidxref.com/5.1.0_r1/xref/frameworks/)/[base](http://androidxref.com/5.1.0_r1/xref/frameworks/base/)/[packages](http://androidxref.com/5.1.0_r1/xref/frameworks/base/packages/)/[DocumentsUI](http://androidxref.com/5.1.0_r1/xref/frameworks/base/packages/DocumentsUI/)/

2.DownloadManager ---->/[frameworks](http://androidxref.com/5.1.0_r1/xref/frameworks/)/[base](http://androidxref.com/5.1.0_r1/xref/frameworks/base/)/[core](http://androidxref.com/5.1.0_r1/xref/frameworks/base/core/)/[java](http://androidxref.com/5.1.0_r1/xref/frameworks/base/core/java/)/[android](http://androidxref.com/5.1.0_r1/xref/frameworks/base/core/java/android/)/[app](http://androidxref.com/5.1.0_r1/xref/frameworks/base/core/java/android/app/)/

3.DownloadProvider ---->/[packages](http://androidxref.com/5.1.0_r1/xref/packages/)/[providers](http://androidxref.com/5.1.0_r1/xref/packages/providers/)/[DownloadProvider](http://androidxref.com/5.1.0_r1/xref/packages/providers/DownloadProvider/)/

下图中用MVC的分层将这三部分做了划分：



其中DocumentsUI作为视图层（V）负责展示Download信息

DownloadManager和DownloadProvder的一部分作为控制层（C）负责下载的逻辑控制

DownloadProvder的另一部分则作为数据层（M）负责数据的存储

## HTMLViewer

shell@kpad:/ $ dumpsys activity | grep mFo

mFocusedActivity: ActivityRecord{2ec50016 u0 com.android.htmlviewer/.HTMLViewerActivity t1}

# 详解app

## Browser

<http://blog.csdn.net/weirdo_c/article/details/50716848>

## CaptivePortalLogin

## CertInstaller

/packages/apps

在Android中安装数字签名，被调用

## KeyChain

<http://blog.csdn.net/innost/article/details/44199503>

数字签名

## LatinIME

/packages/inputmethods/LatinIME/

com.android.settings/.inputmethod.InputMethodAndSubtypeEnablerActivity

com.android.inputmethod.latin/.settings.SettingsActivity

<http://blog.csdn.net/asasasdasfaf/article/details/53744675>

## PicoTts

http://www.manew.com/blog-166890-7593.html