IPC

开启多进程方式：android:process，通过JNI在native层fork一个进程

android:process——默认进程名是包名，

android:process=":xyz"——为包名加+“:xyz”，其为私有进程，其他应用的组件不可以跑在同一进程中

android:process="com.abc"——com.abc，全局进程，其他应用可以通过shareUID方式跑在同一进程

多进程：同一进程的组件属于同一个虚拟机和同一个application，内存空间

1、静态成员变量失效，2、线程同步机制失效，3、SharePreference可靠性下降，4、application多次创建

Parcelable

describeContents方法：通常返回0，效值是CONTENTS\_FILE\_DESCRIPTOR(0x01)，指明这个Parcel的内容包含文件描述符，文件描述符就是操作文件的数据结构，获取到文件描述符可以完成所有文件相关的操作

Binder

客户端和服务端在同一进程，返回服务端的Stub本身，否则是Stub.proxy对象

最大线程数：15个

死亡代理：linkToDeath，当Binder死亡时，会收到通知

调用服务端/客户端的方法在当前线程中，然后挂起线程，收到返回数据唤醒

被调用的方法在Binder线程池中

文件共享

sharePreferences本身是文件共享，有一定缓存策略，内存中会保存一份缓存，在多进程模式下不可靠

Message

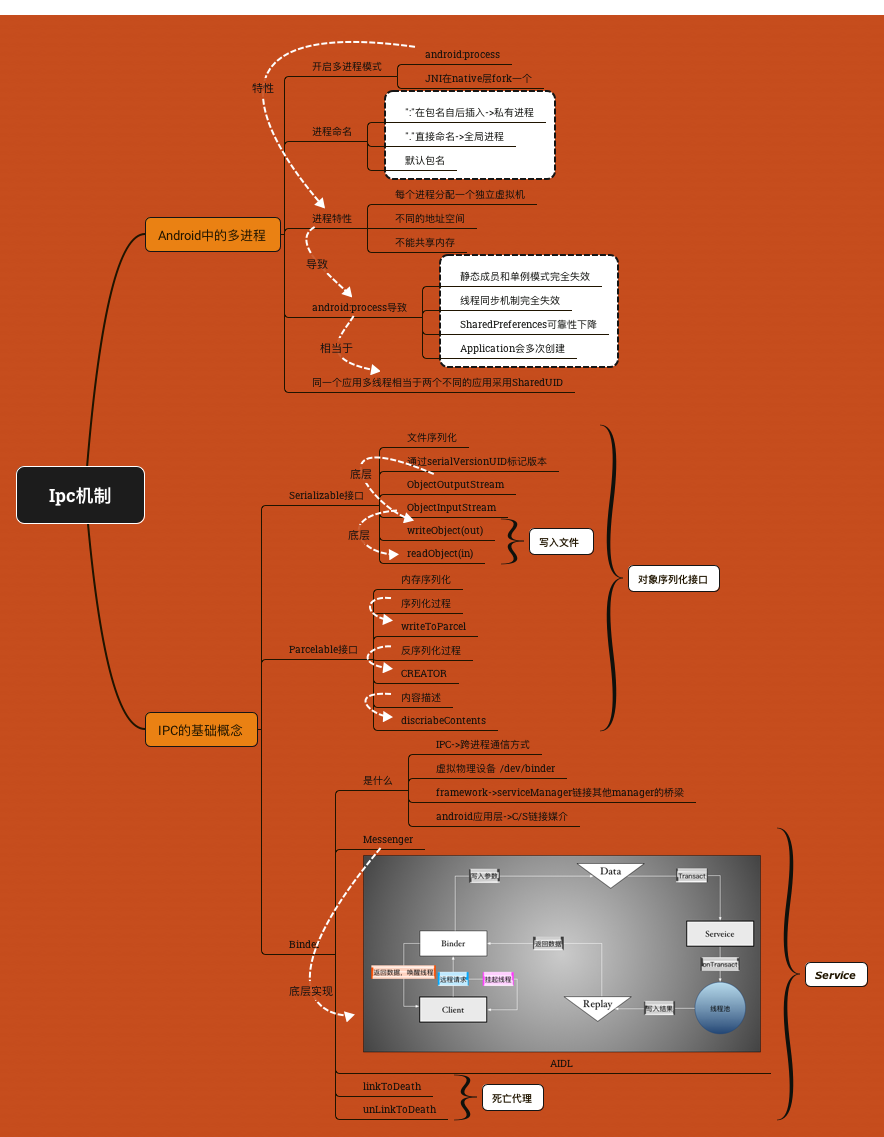
obj：只支持非空的系统实现Parcelable接口数据

AIDL

in：由客户端流向服务端，服务端不能改变客户端

out：服务端流向客户端，服务端改变，客户端改变

inout：双向流通



Dalvik

1、允许有限内存中同时运行多个虚拟机实例，并且每一个Dalvik作为独立进程——防止虚拟机崩溃时，所有程序都被关闭

2、应用每次运行的时候，字节码都需要通过即时编译器转换为机器码

Dalvik监视所有的程序，并且创建依存关系树，为每个程序优化代码并存储在Dalvik缓存中。第一次加载后会生成Cache文件，以提供下次快速加载。

Dalvik解释其采用预先算好的Goto地址，每个指令对内存的访问都在64字节边界上对接。这样可以节省一个指令后进行查表的时间。

ART 2014androidL

1、应用第一次安装的时候，字节码就预先编译成机器码

2、机器码占用存储空间更大，

3、安装时间变长

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Dalvik | 执行时编译 | 安装快 | 开启慢 | 占用空间小 |
| ART | 安装时编译 | 安装慢 | 开启快 | 占用空间大 |