**PackageManager之uninstall**

1. API：deletePackageAsUser

public void deletePackage(final String packageName,

final IPackageDeleteObserver2 observer, final int userId, final int flags)

参数：

packageName ：卸载的包名

Observer： 卸载状态的回调对象

Flags：卸载时使用的标志

public static final int DELETE\_KEEP\_DATA = 0x00000001;

public static final int DELETE\_ALL\_USERS = 0x00000002;

public static final int DELETE\_SYSTEM\_APP = 0x00000004;

1. deletePackageX函数的解析：

1.创建PackageRemovedInfo对象，该对象的InstallArgs args成员用来清除安装包、lib库、odex文件。

2.检查flags是否是DELETE\_ALL\_USERS

3.检查每个user的Installed状态。

int[] **allUsers**;

boolean[] **perUserInstalled**;

synchronized (mPackages) {

PackageSetting ps = mSettings.mPackages.get(packageName);

**allUsers** = sUserManager.getUserIds();

perUserInstalled = new boolean[allUsers.length];

for (int i = 0; i < allUsers.length; i++) {

**perUserInstalled**[i] = ps != null ? **ps.getInstalled(allUsers[i])** : false;

}

}

通过PackageSetting检查每个user的Installed状态保存在perUserInstalled数组中。

4.调用deletePackageLI函数。

private boolean deletePackageLI(String packageName, UserHandle **user**,

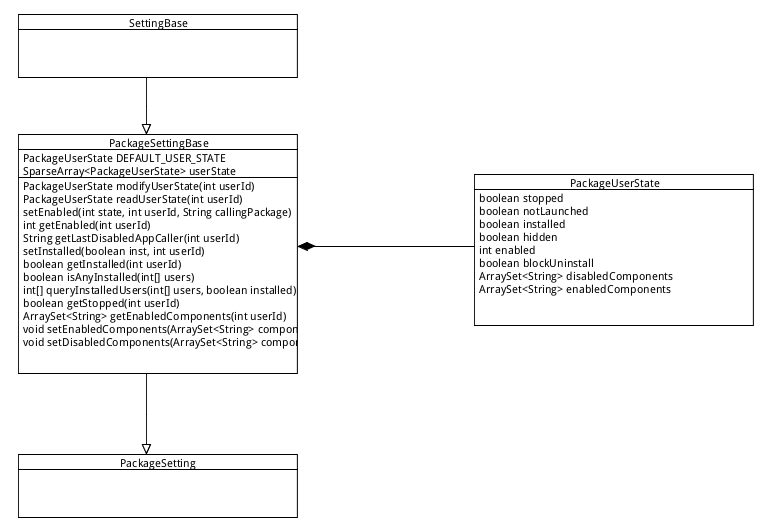
boolean **deleteCodeAndResources**, int[] allUserHandles, boolean[] perUserInstalled,

int flags, PackageRemovedInfo outInfo,

boolean writeSettings) {

传入deleteCodeAndResources = true writeSettings =true.

当user的id不为-1，说明只是当前用户删除该package，通过PackageSetting的setUserState函数设置user的状态。



ps.setUserState(userId,

COMPONENT\_ENABLED\_STATE\_DEFAULT,

false, //installed

true, //stopped

true, //notLaunched

false, //hidden

null, null, null,

false, // blockUninstall

ps.readUserState(userId).domainVerificationStatus, 0);

当package为非system package或者设置的PackageManager.DELETE\_SYSTEM\_APP flag，通过PackageSetting的isAnyInstalled函数来检查是否有其他user进行了该app的安装操作。

如果有其他user对该package进行了installed的动作，那么调用scheduleWritePackageRestrictionsLocked函数进行Restrictions xml文件的更新，同时调用mInstaller.clearUserData、removeKeystoreDataIfNeeded等函数进行相应的清除工作返回。针对设置的PackageManager.DELETE\_SYSTEM\_APP flag的system

Package也进行同样的清除动作。

当没有其他user具有该packgae的install的动作，那么检查是否为system app

当为system app的时候调用deleteSystemPackageLI函数进行清除动作，当为非系统app的时候调用killApplication函数后调用deleteInstalledPackageLI。

**deleteInstalledPackageLI函数解析：**

调用removePackageDataLI函数进行remove Package Data操作：

1.调用removePackageLI函数将package所有的信息从packagemanager中的容器中删除。

2.调用removeDataDirsLI函数删除user data目录中的数据。

3.调用mSettings.removePackageLPw函数来删除settings中信息包括PackageSetting和appid。

4.updatePermissionsLPw：删除该app中定义的permission。

5.mSettings.writeLPr：更新config 文件。

6.removeKeystoreDataIfNeeded

调用createInstallArgsForExisting函数创建InstallArgs来删除 application code and resources。

5.发送broadcast。

6.调用info.args.doPostDeleteLI函数进行删除安装包、lib库、odex文件。

通过调用step 4中创建的InstallArgs的doPostDeleteLI函数来删除 application code and resources