1. 开机扫描：

**构造函数：**

private static final File mCwAppInstallDir = getPreloadDir();

if (null != PackageManager.SENSE\_VERSION) {

scanDirLI(mCwAppInstallDir, PackageParser.PARSE\_IS\_DATA\_PRELOAD

| PackageParser.PARSE\_IS\_DATA\_PRELOAD\_DIR, scanFlags | scanIgnoreNativeLibrary, 0);

}

**scanPackageLI(File):**

if (null != PackageManager.SENSE\_VERSION && mSettings.mUpdatedDataPreloadApps.get((ps != null ? ps.name : pkg.packageName)) != null) {

parseFlags |= PackageParser.PARSE\_IS\_DATA\_PRELOAD;

}

**App --->system app:**

if (null != PackageManager.SENSE\_VERSION && mSettings.mUpdatedDataPreloadApps.containsKey(ps.name)) {

parseFlags &= ~PackageParser.PARSE\_IS\_DATA\_PRELOAD;

mSettings.mUpdatedDataPreloadApps.remove(ps.name);

}

**scanPackageDirtyLI(PackageParser.Package):**

If (null != PackageManager.SENSE\_VERSION && (parseFlags&PackageParser.PARSE\_IS\_DATA\_PRELOAD) != 0) {

pkg.applicationInfo.HTCFlags |= ApplicationInfo.FLAG\_DATA\_PRELOAD;

}

**replaceNonSystemPackageLI(PackageParser.Package):**

if (null != PackageManager.SENSE\_VERSION && isDataPreloadApp(deletedPackage)) {

if (isPreloadCodePath(deletedPackage.codePath)) {

mSettings.mUpdatedDataPreloadApps.put(pkgName, getApkName(deletedPackage.codePath));

}

pkg.applicationInfo.HTCFlags |= ApplicationInfo.FLAG\_UPDATED\_DATA\_PRELOAD\_APP;

parseFlags |= PackageParser.PARSE\_IS\_DATA\_PRELOAD;

}

scanPackageLI(pkg);

设置系统应用：

scanPackageDirtyLI(PackageParser.Package）

if ((parseFlags&PackageParser.PARSE\_IS\_SYSTEM) != 0) {

pkg.applicationInfo.flags |= ApplicationInfo.FLAG\_SYSTEM;

} else {

// Only allow system apps to be flagged as core apps.

}