DexClassLoader(String dexPath, String optimizedDirectory, String libraryPath, ClassLoader parent)

非空

\* ClassLoader(ClassLoader parentLoader, boolean nullAllowed)

本参数永false

\* public BaseDexClassLoader(String dexPath, File optimizedDirectory, String libraryPath, ClassLoader parent)

\* public DexPathList(ClassLoader definingContext, String dexPath, String libraryPath, File optimizedDirectory)

\* private static ArrayList<File> splitDexPath(String dexPath)

\* private static ArrayList<File> splitPaths(String path1=dexPath, String path2=null, boolean wantDirectories=false)

\* private static void splitAndAdd(String searchPath= dexPath, boolean directoriesOnly, ArrayList<File> resultList)

JAVACALL Java\_ Posix\_ stat

android-4.4.4\_r1\libcore\dalvik\src\main\java\dalvik\system\DexClassLoader.java

DexClassLoader、BaseDexClassLoader、ClassLoader

DexPathList、

**DexClassLoader(String dexPath, String optimizedDirectory, String libraryPath, ClassLoader parent)**

Parent 不允许为null

dexPath 不允许为 null

**在这个入口所做的事情**

1. 为成员变量 ClassLoader 赋值
2. 遍历给定的参数 dexPath(包含子文件夹和子文件，没错dexPath可以是文件夹)并将其中的dex/zip/jar/apk规整到一个 Element[ ] 数组中
3. 将给定的 libraryPath 规整到 File[ ] 数组中

在 **DexClassLoader** 中有一些加载 dex 文件的代码我们详究一下

android-4.4.4\_r1\libcore\dalvik\src\main\java\dalvik\system\DexPathList.java

* DexPathList
* DexFile

**private static DexFile loadDexFile(File file, File optimizedDirectory)**

file dex File 对象

optimizedDirectory DexClassLoader 初始化的时候传入的参数

**这个入口所做的事情**

**openDexFileNative（path,null,0）**

**dvmRawDexFileOpen(sourceName, outputName, &pRawDexFile, false)**

**dexpath null null false**

感觉是要对dex 进行优化了，这种感觉尚需验证。

Fork()

但是怎么就做了 optimizition

dvmChangeStatus -> I:\AOSP\android-4.4.4\_r1\dalvik\vm\Thread.cpp