

[5] Launcher3

初识Launcher

Launcher - Android桌面启动器

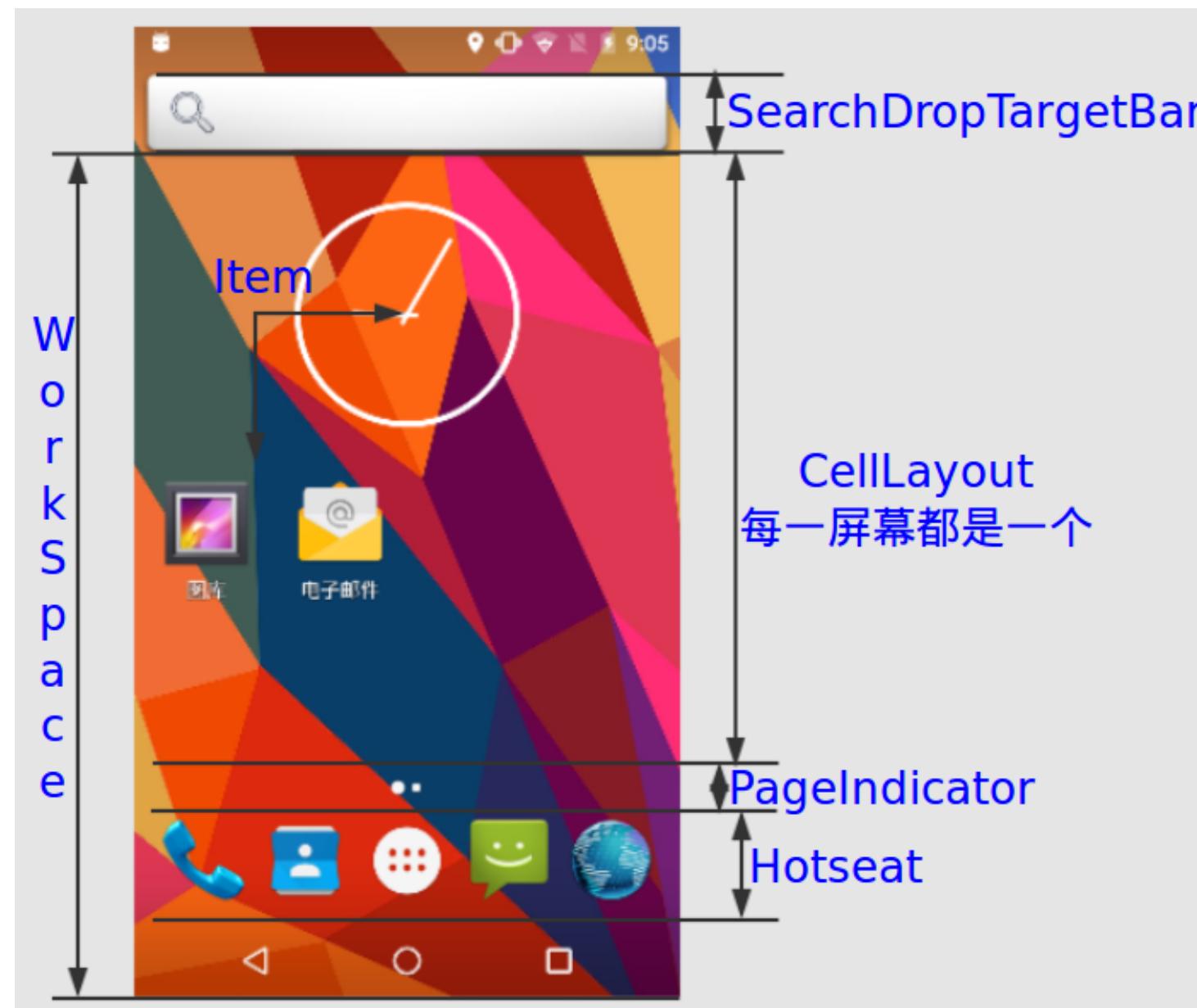
- Android的桌面UI统称Launcher。Launcher是Android的主要组件之一，主要功能是打开应用（通过App shortcut或Widget等方式）；
- Launcher类似Linux的KDE，Gnome等UI界面，对于系统来说可有可无。通过adb等方式可以不经Launcher打开一个应用，但缺乏Launcher，普通用户将无法操作。

Launchers

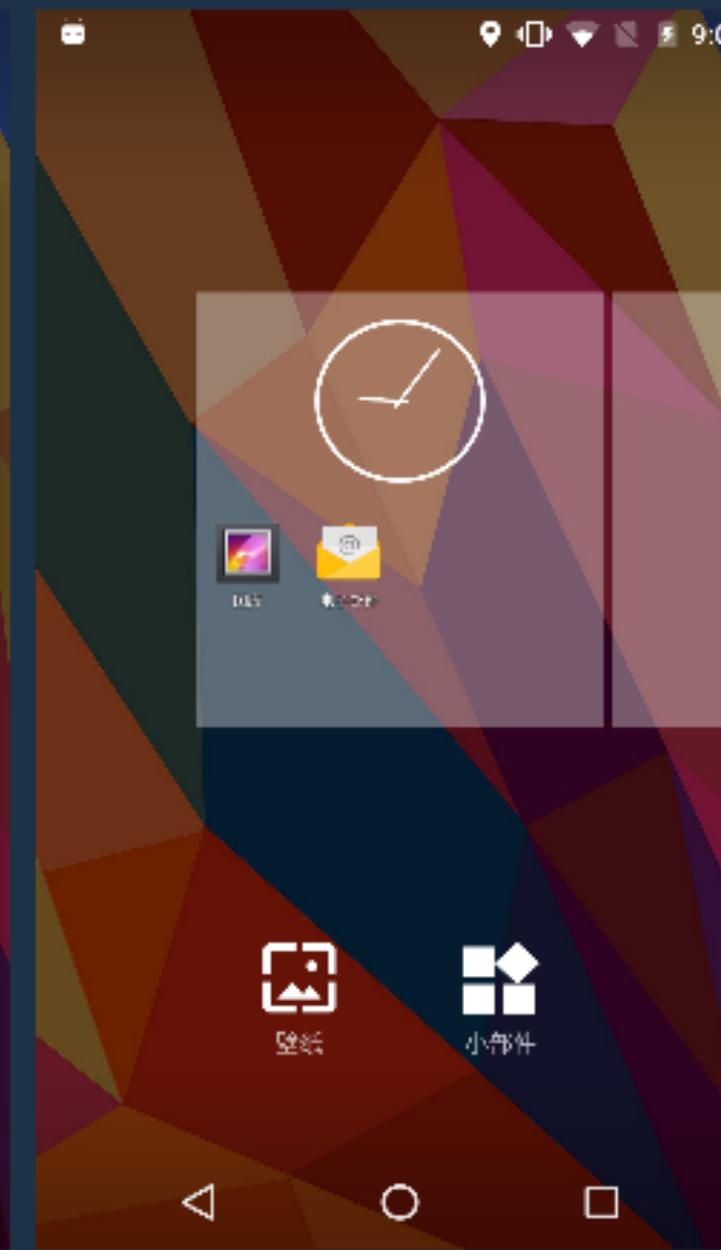
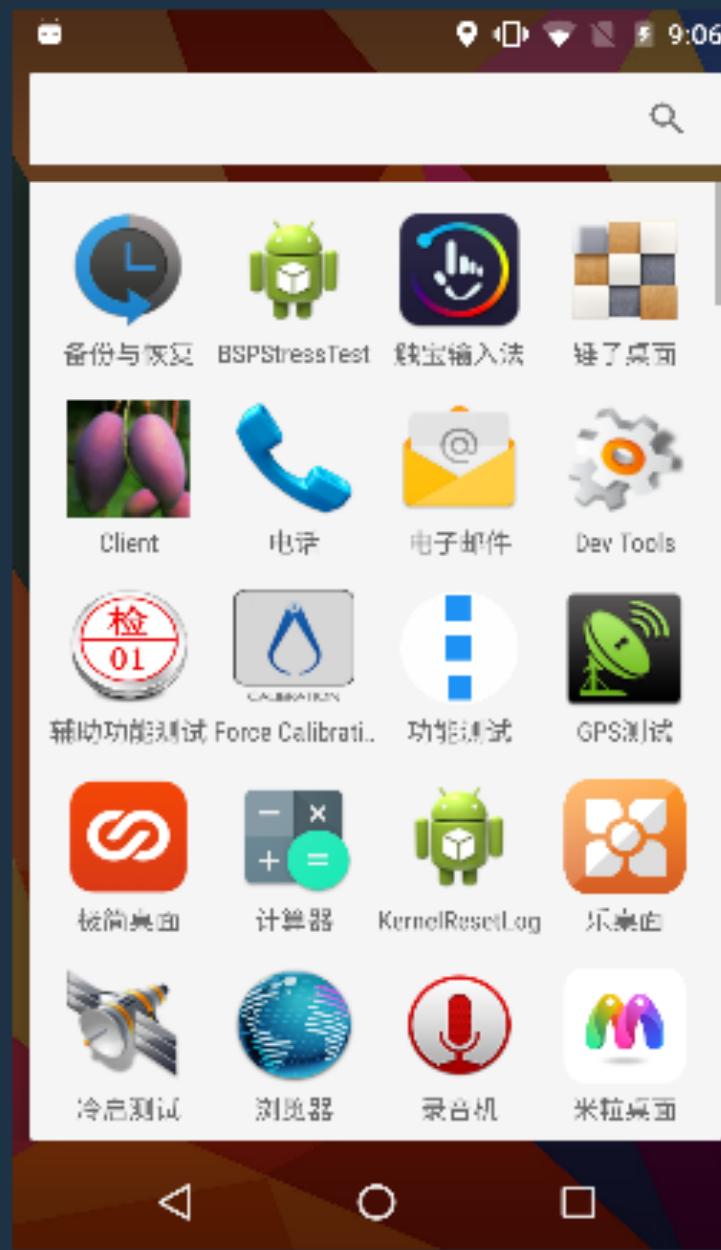
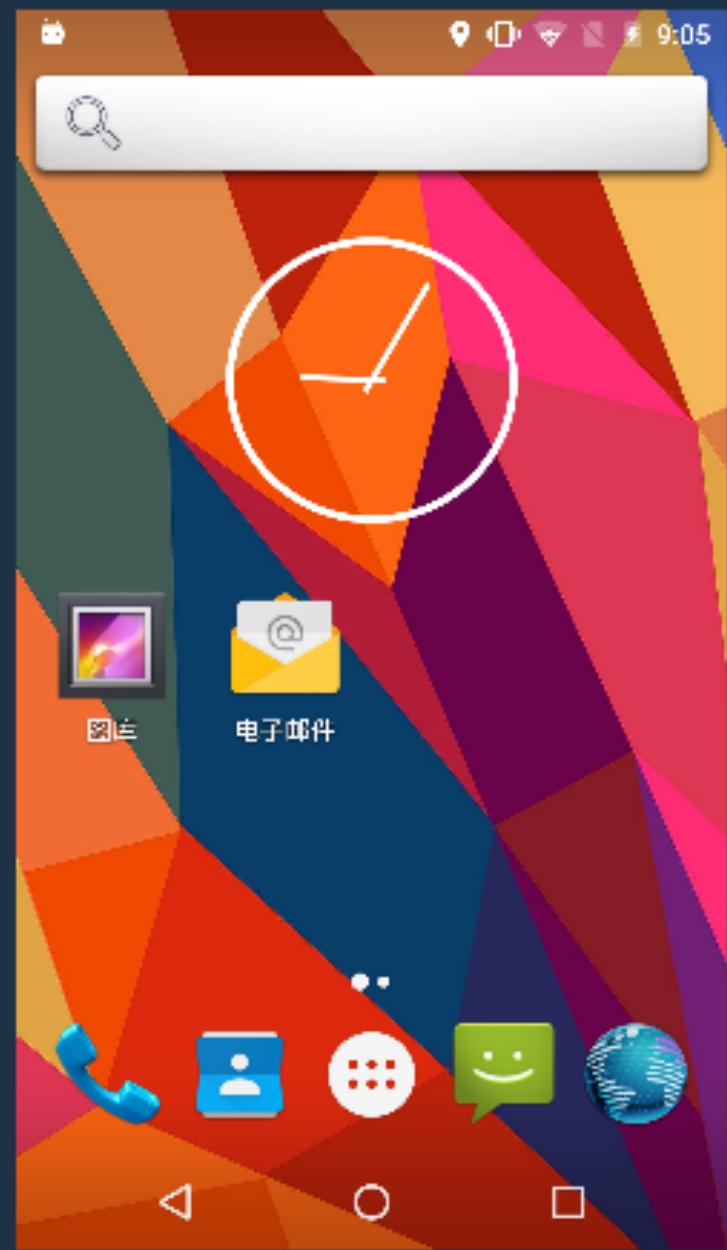


依次是 Launcher3, MiHome, Go桌面, iOS8

Google Launcher3



Launcher 相关的几大状态 (State 枚举) :



DragLayer
Workspace(PagedView)
CellLayout
SearchDropTargetBar
BubbleTextView
PageIndicator
Hotseat

AllAppsContainerView
AllAppsRecyclerViewContainerView
AllAppsRecyclerView

CellLayout
overview_panel

WidgetsContainerView
WidgetsRecyclerView

下载源码

Google 源码地址 <https://android.googlesource.com>

Launcher3 <https://android.googlesource.com/platform/packages/apps/Launcher3>

Github fork <https://github.com/jzhangs/Launcher3>, Tag: **android-6.0.1_r46.**

设置Android Studio开发环境

- 1.** 更新Gradle脚本 (version 2.1.0; 重命名applicationID, 避免进程名冲突)
- 2.** 删除多余的<action android:name="android.intent.action.MAIN" />, 加入category Launcher
- 3.** 修改AndroidManifest.xml里的自定义权限
- 4.** 修改provider的authority

见 <https://github.com/jzhangs/Launcher3/commits/dev>

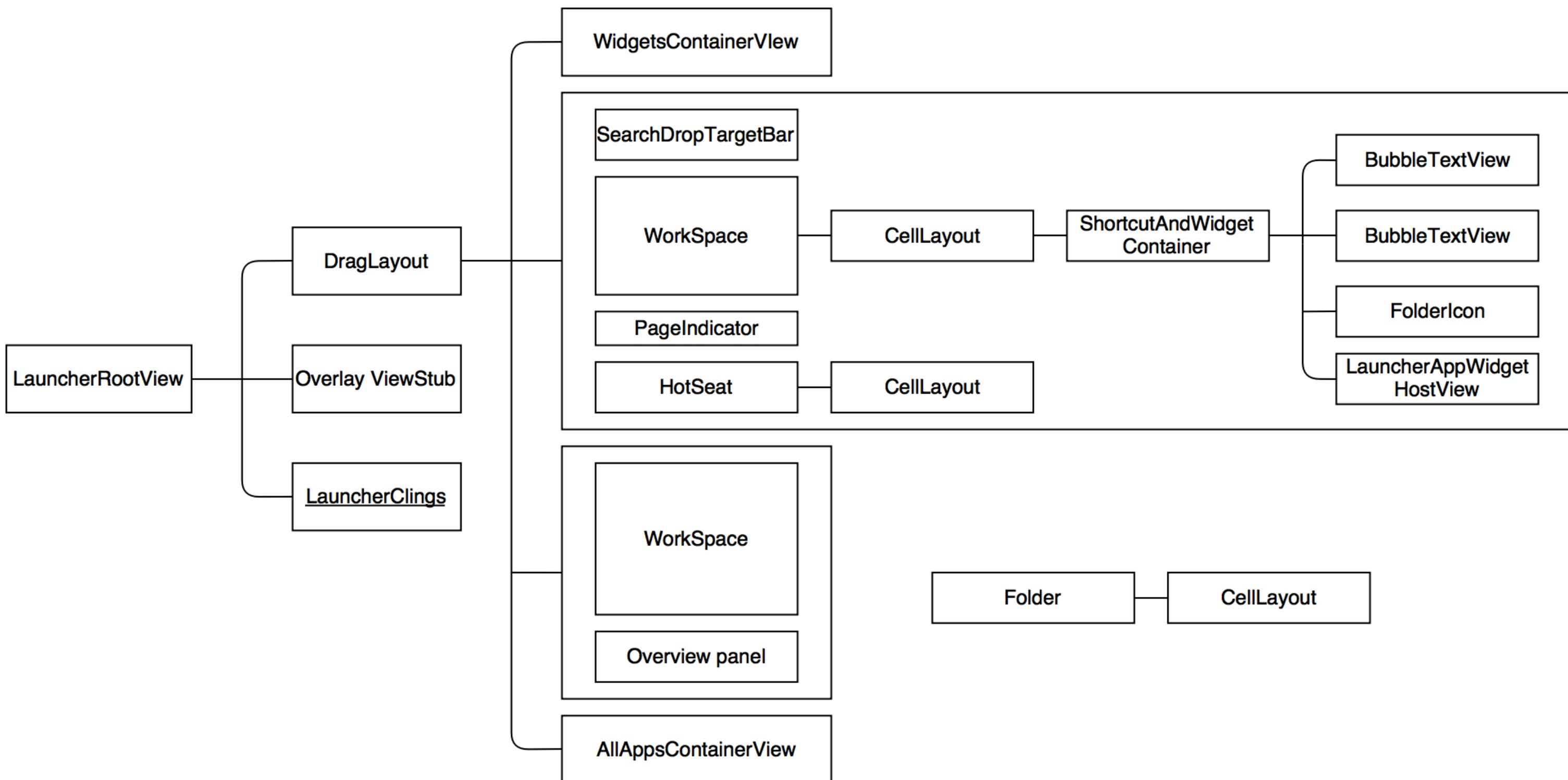
Launcher3的布局

AndroidManifest.xml

```
activity: Launcher, WallpaperPicker, WallpaperCrop, Settings  
provider: LauncherProvider  
receiver: WallpaperChange, InstallShortcut, AppWidgetRestore, Startup  
service: MemoryTracker
```

Launcher.java::setContentView, R.layout.launcher

launcher.xml



Launcher.java是最主要的Activity，`setContentView`的参数是`R.layout.launcher`。只考虑桌面竖屏的情况，相关源码在`res/layout-port/launcher.xml`。

`launcher.xml`布局文件的最外面是自定义的**LauncherRootView**，实际上是一个`FrameLayout`，其中嵌套了同样继承自`FrameLayout`的`DragLayer`。`DragLayer`的主要功能就是处理拖拽事件。当拖拽一个图标的时候，将它对应的view放到`DragLayer`里，跟随手指移动。

SearchDropTargetBar 搜索框 拖动图标到上面时，会被替换成“删除”；

WorkSpace用来管理页面，父类是**PagedView**（处理左右滑动的`ViewGroup`）。每个页面是一个`CellLayout`；

PageIndicator，顾名思义，页面指示符；

HotSeat，放置常用应用，同样是一个**CellLayout**布局。

Launcher3 常用类

Launcher 实现Launcher功能的核心Activity，包括除WallpaperSet和Settings之外的View

LauncherAppState 单例，初始化对象（LauncherModel ..），注册各类广播（应用安装、卸载、更新，配置变化等）

LauncherModel 数据处理类，保存桌面的运行状态，提供读写数据库的API，内部类**LoaderTask**用来初始化桌面

LauncherProvider 核心数据库类，负责launcher.db的创建与维护

LauncherRootView 竖屏模式下根布局，继承了InsettableFrameLayout，控制是否显示在状态栏等下面

InvariantDeviceProfile 不变的设备相关参数管理类

LauncherAppsCompat 获取已安装App列表信息的兼容抽象基类， Compat - Compatibility

AppWidgetManagerCompat 获取AppWidget列表的兼容抽象基类

LauncherStateTransitionAnimation 各类动画总管处理执行类

WidgetPreviewLoader 存储Widget信息的数据库， 内部创建了数据库widgetpreviews.db

LauncherAppWidgetHost AppWidgetHost子类， 是桌面插件宿主， 为了方便拖拽等才继承处理的

LauncherAppWidgetHostView AppWidgetHostView子类， 配合LauncherAppWidgetHost得到HostView

DragLayer 负责分发事件的ViewGroup

DragController 拖拽处理接口

DragSource/DropTarget DragSource表示图标从哪开始拖，DropTarget表示图标被拖到哪去

DragView 拖动图标时跟随手指移动的View

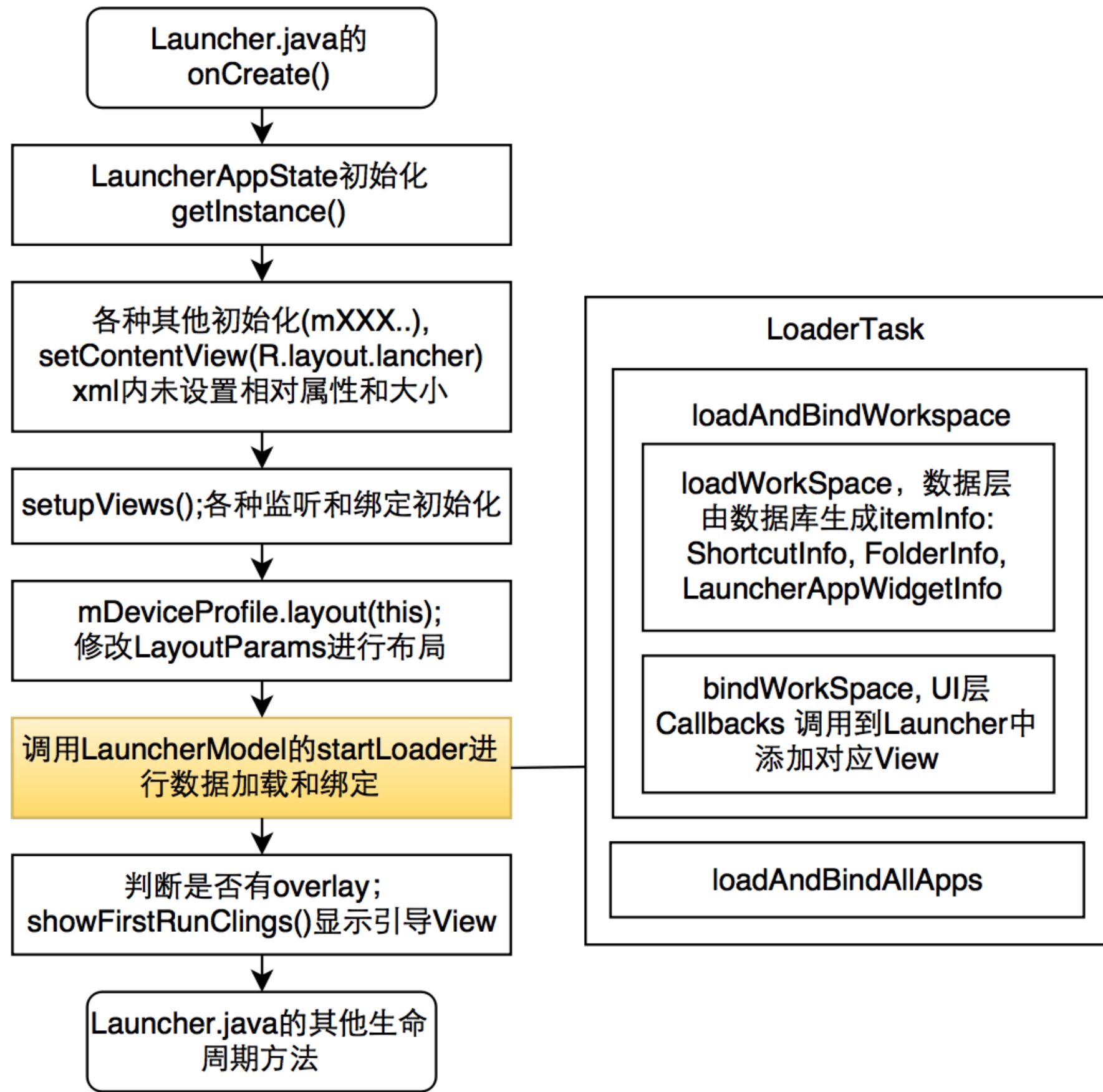
ItemInfo 桌面上每个Item的信息数据结构，包括行、列、宽高、第几屏等信息；该对象与数据库中记录一一对应；有多个子类，譬如FolderIcon的FolderInfo、BubbleTextView的ShortcutInfo等

BubbleTextView 图标都基于他，继承自TextView

Folder 打开文件夹展示的View

FolderIcon 文件夹图标

IconCache 应用程序icon和title的缓存，内部类创建了数据库app_icons.db



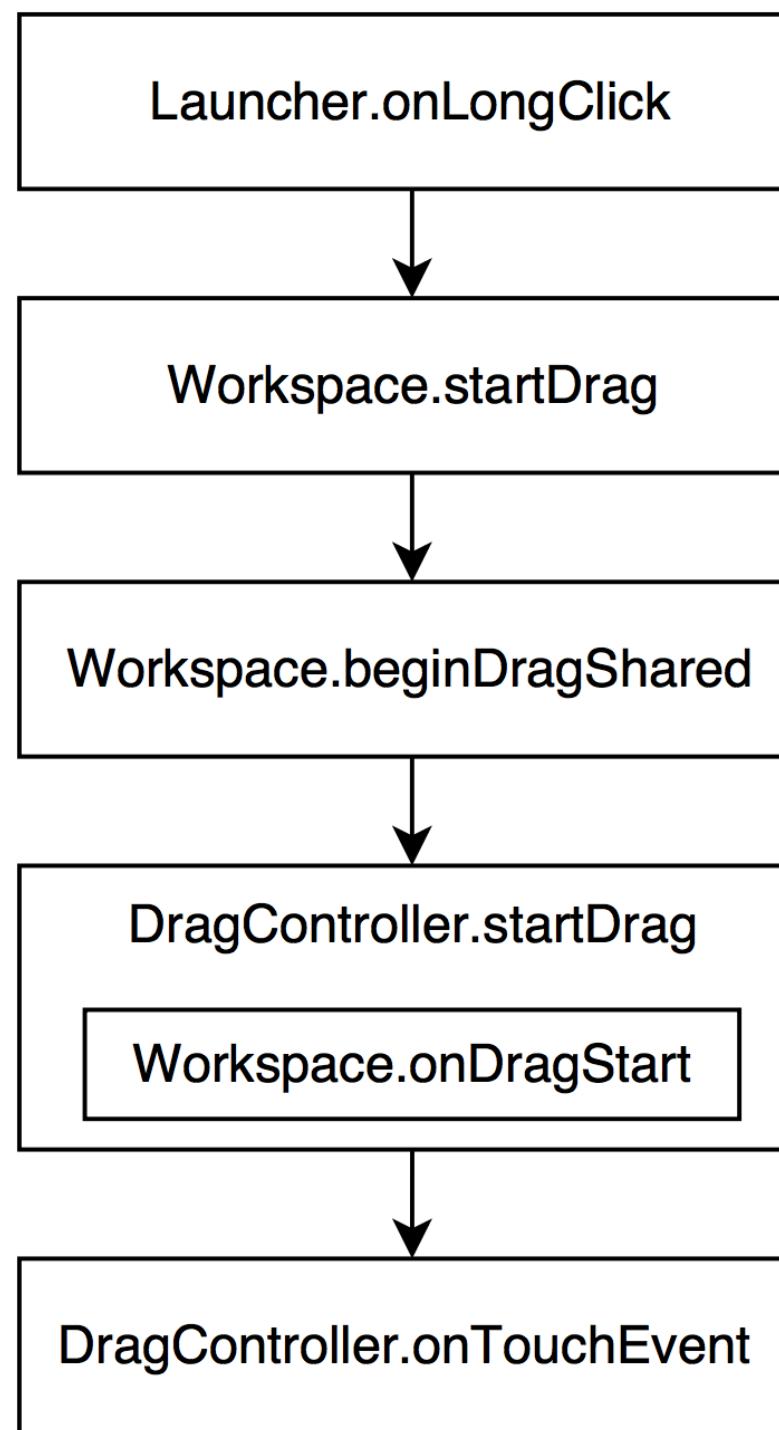
Launcher启动流程

LauncherProvider

- DatabaseHelper, **onCreate/onUpgrade**
- **LauncherSettings** 两张表: favorites, workspaceScreens.
 - LauncherModel中调用CONTENT_URL做CRUD.
- ItemInfo (id, itemType, ..)

拖拽流程

· 长按



长按icon或folder

隐藏原先的图标

绘制跟随手指移动的Bitmap

生成DragView并显示，设置Drag状态
Workspace实现了DragController.DragListener接口

调用自DragLayer, 处理MOVE事件

. 移动

- DropTarget 接口 (Workspace, Folder ..)
- **handleMoveEvent**
 - **findDropTraget**
 - **checkTouchMove** (dropTraget.onDragEnter, onDragExit ..)
 - **checkScrollState**

. 放开

DragController.drop -> Workspace.onDrop

DragController.enDrag

TODO

- PagedView
- Custom content
- IconCache
- Wallpaper
- Widgets
- Animations
- ..

New Features

- Folder auto organization
- Themes
- Download indicator
- Icon subscripts
- ..