1. Launcher的启动

该方法是system\_server在启动的最后阶段，向系统查询 Home Intent，然后启动Launcher, 这个启动是在Home Stack里启动的，系统的第一个ActivityStack, stack id为0， 启动过程很简单， Launcher的启动模式定义成了”single task” . 注意 SourceRecord/ResultTo的 Activity都为空.

2. am start xxxx/.xxx

这时会分为两种情况

1) 如果系统里只有home stack,

新建一个ActivityStack(不用HomeStack), 然后新建Task并跑到新建立的ActivityStack里

2) 如果系统里不只有home stack

那么这时候会re-use最顶层的 ActivityStack (不用HomeStack), 然后新建Task运行到找到的ActivityStack中

该方法是在 adb shell里启动的，启动过程和Launcher的启动过程基本一样， 新建一个 ActivityStack, TaskRecord

注意，系统里只有两个stack, 一个home stack, 一个其它stack.

3. 在Launcher中启动

(Launcher.java) onClick -> startAppShortcutOrInfoActivity -> startActivitySafely -> startActivity (Launcher.java)

private boolean startActivity(View v, Intent intent, Object tag) {

intent.addFlags(Intent.FLAG\_ACTIVITY\_NEW\_TASK);

//可以看出从Launcher里启动的app全部会运行在新建立的task 里

…

startActivity(intent, optsBundle);

}

如果是打开第一个app, 那么主要流程是

新建一个ActivityStack 并move到front , 在新的ActivityStack中创建 一个新的Task -> 对HomeStack 中mResumedActivity(即Launcher activity)进行Pause ->

当pause完成后 –> 让处于front的ActivityStack 进行resume (创建进程,attach/bind/launch)

如果是打开的非第一个app, 那么主要流程是

找到非HomeStack的那个ActivityStack 并move到front , 在新的ActivityStack中创建 一个新的Task -> 对HomeStack 中mResumedActivity(即Launcher activity)进行Pause ->

当pause完成后 –> 让处于front的ActivityStack 进行resume (创建进程,attach/bind/launch)

HelloWorld 里的Activity情况,

MainActivity -> FooActivity (没有定义launchMode, 那么默认为standard模式)

计算出 focused stack, 在新的stack里去运行task, 调度activity

ActivityStack computeStackFocus(ActivityRecord r, **boolean** newTask) {  
 **final** TaskRecord task = r.task;  
  
 *// On leanback only devices we should keep all activities in the same stack.* **if** (!mLeanbackOnlyDevice &&  
 (r.isApplicationActivity() || (task != **null** && task.isApplicationTask()))) {  
  
 ActivityStack stack;  
 **if** (task != **null** && task.stack != **null**) {  
 stack = task.stack;  
 **return** stack;  
 }

//如果要启动的Activity的task已经运行在某个栈里，那么对这个Activity，就使用该栈  
  
 **final** ActivityContainer container = r.mInitialActivityContainer;  
 **if** (container != **null**) {  
 *// The first time put it on the desired stack, after this put on task stack.* r.mInitialActivityContainer = **null**;  
 **if** (***DEBUG\_FOCUS*** || ***DEBUG\_STACK***) Slog.*d*(***TAG\_FOCUS***,  
 "computeStackFocus: from container");  
 **return** container.mStack;  
 }

// isEligibleForNewTasks() 永远返回true, 这里直接返回当前的focused stack  
 **if** (mFocusedStack != mHomeStack && (!newTask ||  
 mFocusedStack.mActivityContainer.isEligibleForNewTasks())) {  
 **if** (***DEBUG\_FOCUS*** || ***DEBUG\_STACK***) Slog.*d*(***TAG\_FOCUS***,  
 "computeStackFocus: Have a focused stack=" + mFocusedStack);  
 **return** mFocusedStack;  
 }  
  
 **final** ArrayList<ActivityStack> homeDisplayStacks = mHomeStack.mStacks;  
 **for** (**int** stackNdx = homeDisplayStacks.size() - 1; stackNdx >= 0; --stackNdx) {  
 stack = homeDisplayStacks.get(stackNdx);  
 **if** (!stack.isHomeStack()) {  
 **if** (***DEBUG\_FOCUS*** || ***DEBUG\_STACK***) Slog.*d*(***TAG\_FOCUS***,  
 "computeStackFocus: Setting focused stack=" + stack);  
 **return** stack;  
 }  
 }

//从所有stacks的最新开始查找，使用最新的stack, 当然除掉 home stack. 这种情况用在当前是focused stack是 HomeStack的情况， 即launcher界面  
  
 *// Need to create an app stack for this user.* stack = createStackOnDisplay(getNextStackId(), Display.***DEFAULT\_DISPLAY***);  
 **if** (***DEBUG\_FOCUS*** || ***DEBUG\_STACK***) Slog.*d*(***TAG\_FOCUS***, "computeStackFocus: New stack r="  
 + r + " stackId=" + stack.mStackId);

//如果所有情况都不满足，那么直接生成一个新的ActivityStack来运行Activity  
 **return** stack;  
 }  
 **return** mHomeStack;  
}

从computeStackFocus可以看出 Android系统只有两个stack, 一个HomeStack 运行Launcher, 其它的activity全部运行在另一个ActivityStack里