Figure - Bug #116941

测试 Test-IT # 110961 (New): V2.0功能测试

测试 Test-IT # 111046 (New): AF-V2.0-系统手势

【AF】【EVT】【系统手势】冷启动搜狐视频HD，底部上滑返回桌面，再点击进入设置界面，等待几秒后自动跳转至搜狐视频HD主界面

2023-03-14 16:21 - CD TEST-方永红

Status:	CLOSED	**Start date:**	2023-03-14
Priority:	Normal	**Due date:**	
Assignee:	CD TEST-方永红	**% Done:**	100%
Category:	CD-FW	**Estimated time:**	0.00 hour
Target version:			
Need_Info:	--	**Found Version:**	FlatBuild_HH_MCE_FSE.M.R.user.01.00.0082.C101(X
Resolution:	WONTFIX	**Degrated:**	--
Severity:	Normal	**Verified Version:**	
Reproducibility:	Every time	**Fixed Version:**	2023-03-31
Test Type:	ST	**Root cause:**	问题原因　google的原生策略，针对后台应用在

Description

【前提条件】
1、设备已开机

【测试步骤】
1、冷启动搜狐视频HD
2、底部上滑返回桌面
3、再点击进入设置界面

【预期结果】
3、正常进入设置页面

【实际结果】
3、等待几秒后自动跳转至搜狐视频HD主界面

History

#1 - 2023-03-20 14:47 - CD APP-王营

- Status changed from New to ASSIGNED

#2 - 2023-03-20 18:12 - CD APP-王营

- File souhu.log added

- Assignee changed from CD APP-王营 to CD FW王武军

■ Current conclusion

从log分析来看是游戏模式启动了搜狐的界面，需要从游戏模式排查此问题。

■ My analysis

Setting界面启动的时间点如下：

11-11 02:26:44.688 1013 10190 I ActivityTaskManager: FOREGROUND_APPLICATION_CHANGED to com.android.settings.homepage.SettingsHomepageActivity

在Settings界面启动后，看起来是游戏模式启动了搜狐的界面，log如下

11-11 02:26:47.338 1013 10190 I thunderperf: Maybe need boostperf for current app!

11-11 02:26:47.338 22694 22744 D FgBoostPerfService: [FgBoostPerfService.perfBoostAcq]start: stat=true pkgName=com.sohu.tv

11-11 02:26:47.339 22694 22744 I FgBoostPerfService: [FgBoostPerfService.perfBoostAcq] not game-mode or game app!

11-11 02:26:47.340 1013 10190 I ThunderPerfManager: PerfAcq error : state=true pkgName=com.sohu.tv
11-11 02:26:47.341 22694 22744 I FgBoostPerfService: [GetGpuMode] isMode=0 isGame=false mGpuMode=0
11-11 02:26:47.343 22694 22744 I FgBoostPerfService: [GetGpuDpiType] isMode=0 isGame=false mGpuDpiType=0
11-11 02:26:47.343 649 659 I mpu_uart: [MSG-P:RECV]:No message received in 1000 ms
11-11 02:26:47.345 22694 22744 I FgBoostPerfService: [GetGpuDpiType] isMode=0 isGame=false mGpuStandardValue=0
11-11 02:26:47.346 1013 10190 I thunderperf: Current app pkgName=com.sohu.tv bm:com.thundercomm.perf.BoostPerfManager@98ca3d gpuMode=-1 gpuType=-1 gpuStandardVal=-1
11-11 02:26:47.346 1013 10190 I ThunderBoostPerfService: Resolution: Uid=10126 Pid=19246 gMode=-1 dType=-1 gpuStandardVal=-1
11-11 02:26:47.347 1013 10190 I thunderperf: PlatormCompat.setOverrides: Uid1=10126 Pid1=19246 Uid2=10126 Pid2=19246
11-11 02:26:47.347 1013 10190 I thunderperf: PlatormCompat.setOverrides: needCheck:false
11-11 02:26:47.347 1013 10190 I thunderperf: CompatConfig.addOverrideUnsafe: bef-state=0
11-11 02:26:47.347 1013 10190 I thunderperf: CompatConfig.addOverrideUnsafe: aft-state=0
11-11 02:26:47.347 1013 10190 W PackageManager: MATCH_ANY_USER flag requires INTERACT_ACROSS_USERS permission: UID 10126 requires android.permission.INTERACT_ACROSS_USERS_FULL or android.permission.INTERACT_ACROSS_USERS to access user 0.
11-11 02:26:47.347 1013 10190 E BoostPerfManager: UpdateResolution SecurityException : e=java.lang.SecurityException: MATCH_ANY_USER flag requires INTERACT_ACROSS_USERS permission: UID 10126 requires android.permission.INTERACT_ACROSS_USERS_FULL or android.permission.INTERACT_ACROSS_USERS to access user 0.
11-11 02:26:47.351 1013 1156 I EventSequenceValidator: inc AccIntentStartedEvents to 2_
11-11 02:26:47.352 1013 10190 I ActivityTaskManager: START u0 {cmp=com.sohu.tv/.ui.activitys.MainActivity} from uid 10126

■ Next action
请武军帮忙确定是否是游戏模式会启动搜狐应用，log已上传到附件

#3 - 2023-03-23 16:24 - CD APP-王营

- Category changed from CD-APP to CD-FW

#4 - 2023-03-28 18:23 - CD FW王武军

【】当前状态
1、分析日志
从日志信息来看，启动的日志是正常的日志，之所以搜狐视屏的应用存在多个START 是因为
应用内部本身存在界面的跳转逻辑，启动了不同的界面。
2、现象分析和逻辑调查
现象分析：
在搜狐视屏这个应用上，进行测试，该问题是必现的。
必现方式：在启动搜狐视屏后，在启动页快速回到桌面，然后快速启动其他应用，搜狐界面会重新显示出来。

逻辑调查：
从该问题表现上来看，逻辑应该是系统原生的逻辑，为了验证我使用原生的设备进行了测试，在搜狐视屏中依旧存在同样的表现。
从目前的逻辑来看，之所以搜狐界面能够重新获取焦点并显示出来，是系统逻辑中针对后台应用启动界面有校验逻辑，
而搜狐的启动方式刚好满足这种校验逻辑，具体的校验逻辑还是分析中。

【】后续处理
调查系统的逻辑原理，明确该问题的发生的根本原因。

#5 - 2023-03-29 15:45 - CD FW王武军

【】当前状态
该种场景的触发逻辑大致如下：

```
    private int executeRequest(Request request) {
        if (!abort) {
```

```
            try {
                Trace.traceBegin(Trace.TRACE_TAG_WINDOW_MANAGER,
                        "shouldAbortBackgroundActivityStart");
                restrictedBgActivity = shouldAbortBackgroundActivityStart(callingUid,
                        callingPid, callingPackage, realCallingUid, realCallingPid, callerApp,
                        request.originatingPendingIntent, request.allowBackgroundActivityStart,
                        intent);
            } finally {
                Trace.traceEnd(Trace.TRACE_TAG_WINDOW_MANAGER);
            }
        }
    }
    boolean shouldAbortBackgroundActivityStart(int callingUid, int callingPid,
            final String callingPackage, int realCallingUid, int realCallingPid,
            WindowProcessController callerApp, PendingIntentRecord originatingPendingIntent,
            boolean allowBackgroundActivityStart, Intent intent) {
        // don't abort if the callerApp or other processes of that uid are allowed in any way
        if (callerApp != null) {
            // first check the original calling process
            if (callerApp.areBackgroundActivityStartsAllowed(appSwitchAllowed)) {
                if (DEBUG_ACTIVITY_STARTS) {
                    Slog.d(TAG, "Background activity start allowed: callerApp process (pid = "
                            + callerApp.getPid() + ", uid = " + callerAppUid + ") is allowed");
                }
                return false;
            }
            // only if that one wasn't allowed, check the other ones
            final ArraySet&lt;WindowProcessController&gt; uidProcesses =
                    mService.mProcessMap.getProcesses(callerAppUid);
            if (uidProcesses != null) {
                for (int i = uidProcesses.size() - 1; i >= 0; i--) {
                    final WindowProcessController proc = uidProcesses.valueAt(i);
                    if (proc != callerApp
                            && proc.areBackgroundActivityStartsAllowed(appSwitchAllowed)) {
                        if (DEBUG_ACTIVITY_STARTS) {
                            Slog.d(TAG,
                                    "Background activity start allowed: process " + proc.getPid()
                                            + " from uid " + callerAppUid + " is allowed");
                        }
                        return false;
                    }
                }
            }
        }
    }
```

上面两个方法的逻辑大致是：
检查当前启动的task是否满足don't abort的条件，如果满足某一种场景，那么就会将restrictedBgActivity修正为false，
就会影响ActivityStarter.mAvoidMoveToFront的值状态，mAvoidMoveToFront值为true，那么就表示当前的这次后台启动的task不会显示出来。

```
    private void setTargetRootTaskIfNeeded(ActivityRecord intentActivity){
        if (differentTopTask && !mAvoidMoveToFront) {
            mStartActivity.intent.addFlags(Intent.FLAG_ACTIVITY_BROUGHT_TO_FRONT);
```

```
            if (mSourceRecord == null || (mSourceRootTask.getTopNonFinishingActivity() != null
                && mSourceRootTask.getTopNonFinishingActivity().getTask()
        mSourceRecord.getTask())) {
              // We really do want to push this one into the user's face, right now.
              if (mLaunchTaskBehind && mSourceRecord != null) {
                intentActivity.setTaskToAffiliateWith(mSourceRecord.getTask());
              }
              final Task launchRootTask = getLaunchRootTask(mStartActivity, mLaunchFlags,
                  intentTask, mOptions);
              if (launchRootTask  null || launchRootTask  mTargetRootTask) {
                // TODO(b/151572268): Figure out a better way to move tasks in above 2-levels
                //  tasks hierarchies.
                if (mTargetRootTask != intentTask
                    && mTargetRootTask != intentTask.getParent().asTask()) {
                  intentTask.getParent().positionChildAt(POSITION_TOP, intentTask,
                      false /* includingParents */);
                  intentTask = intentTask.getParent().asTask();
                }
                // If the task is in multi-windowing mode, the activity may already be on
                // the top (visible to user but not the global top), then the result code
                // should be START_DELIVERED_TO_TOP instead of START_TASK_TO_FRONT.
                final boolean wasTopOfVisibleRootTask = intentActivity.mVisibleRequested
                    && intentActivity  mTargetRootTask.topRunningActivity();
                // We only want to move to the front, if we aren't going to launch on a
                // different root task. If we launch on a different root task, we will put the
                // task on top there.
                // Defer resuming the top activity while moving task to top, since the
                // current task-top activity may not be the activity that should be resumed.
                mTargetRootTask.moveTaskToFront(intentTask, mNoAnimation, mOptions,
                    mStartActivity.appTimeTracker, DEFER_RESUME,
                    "bringingFoundTaskToFront");
                mMovedToFront = !wasTopOfVisibleRootTask;
              } else {
                intentTask.reparent(launchRootTask, ON_TOP, REPARENT_MOVE_ROOT_TASK_TO_FRONT,
                    ANIMATE, DEFER_RESUME, "reparentToTargetRootTask");
                mMovedToFront = true;
              }
              mOptions = null;
            }
          }
        }
```

综合原生逻辑和实际现象表现来看：

1）搜狐视频在启动过程中，应用内部会存在启动两次activity的逻辑，而且二者执行时间是有一定的间隔

03-26 04:23:46.996 1066 12329 I ActivityTaskManager: START u0 {act=android.intent.action.MAIN cat=[android.intent.category.LAUNCHER] flg=0x10200000 cmp=com.sohu.tv/.ui.activitys.WelcomeActivity bnds=[726,388][1026,680]} from uid 10076

03-26 04:23:54.649 1066 3744 I ActivityTaskManager: START u0 {cmp=com.sohu.tv/.ui.activitys.MainActivity} from uid 10102

2) 整个流程

在冷启动搜狐应用时，执行了第一个WelcomeActivity的启动，然后迅速回到home，启动其他的应用，其他应用的界面虽然获取了焦点显示了，但是搜狐应用内部还会自动启动另一个MainActivity，

而启动过程中系统的检查逻辑发现搜狐的这次启动满足后台启动task并显示的逻辑，所以搜狐应用重新获取了焦点将界面显示出来了。

3）在系统层面针对搜狐这样的启动和响应方式是支持的，至少操作流程是满足条件的，所以这个问题从系统功能层面来看不需要刻意处理和修复。

【】后续处理

综合结论：

1、发生同样问题的应用不多，这与应用自身的启动流程和逻辑有一定的关系。

2、该问题的发生是原生逻辑正常执行的结果，从逻辑层面来看该问题不用刻意修改。

#6 - 2023-03-30 14:15 - CD FW王武军

- Status changed from ASSIGNED to RESOLVED

- Assignee changed from CD FW王武军 to CD TEST-方永红

- % Done changed from 0 to 100

- Resolution changed from -- to WONTFIX

- Fixed Version set to 2023-03-31

- Root cause set to 问题原因

google的原生策略，针对后台应用在一定时间和条件范围内，允许从后台重启获取窗口焦点并显示出来

【】当前状态

综合之前的分析和调查，针对该票建议做如下处理：

1）不刻意修改问题描述的情况，该票维持现状。

2）请测试进行确认。

#7 - 2023-03-30 17:03 - CD TEST-方永红

- Status changed from RESOLVED to VERIFIED

3.30

原生问题，确认不做处理

#8 - 2023-03-30 17:04 - CD TEST-方永红

- Status changed from VERIFIED to CLOSED

Files

log1613.txt	4.78 MB	2023-03-14	CD TEST-方永红
Phone-STS40X190078 2023-03-14 16-20-41.mp4	11.1 MB	2023-03-14	CD TEST-方永红
souhu.log	555 KB	2023-03-20	CD APP-王营