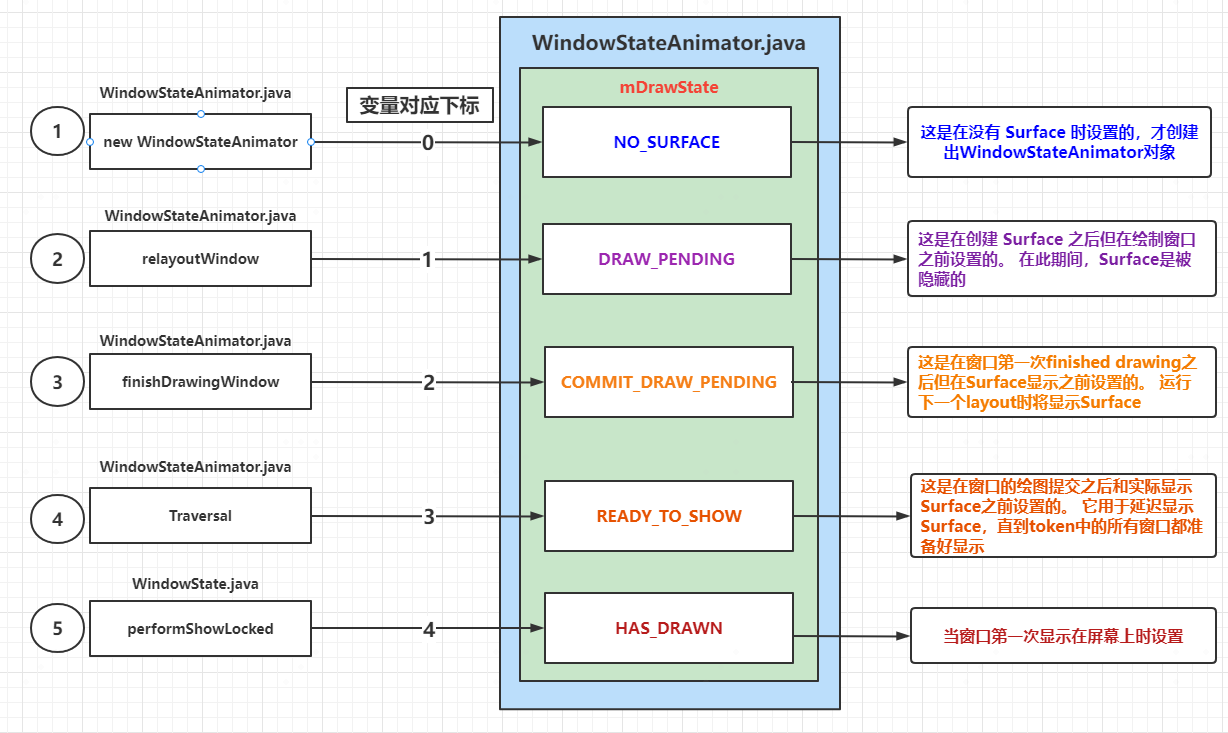
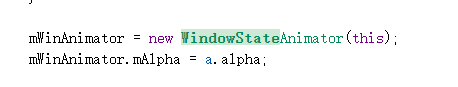
**mDrawState学习**



**NO\_SURFACE:**

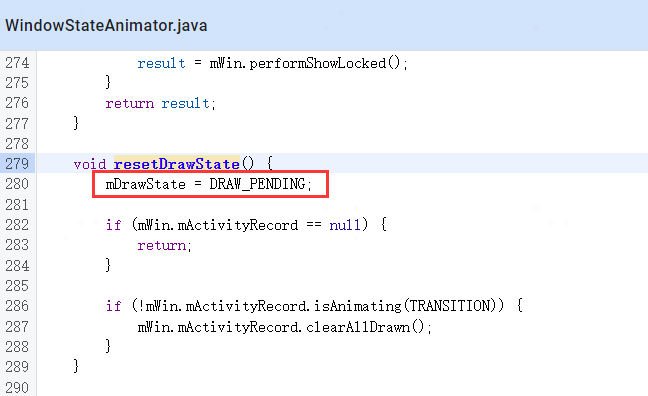
**new WindowState():**



**DRAW\_PENDING:**

**addWindow阶段应用进程已经往chorrographt注册一个CALLBACK\_TRAVERSAL的Runnable回调，在下一帧的Vsync信号到来时会执行TraversalRunnable，TraversalRunnable的作用就是触发measure、layout、draw这三个动作并通知WMS去relayoutWindow，relayoutWindow时WMS会给新创建的WindowStateAnimator设置状态mDrawState为DRAW\_PENDING以及为窗口创建BufferStateLayer图层的SurfaceControl对象。**

**createsurfaceLocked->resetDrawState():**



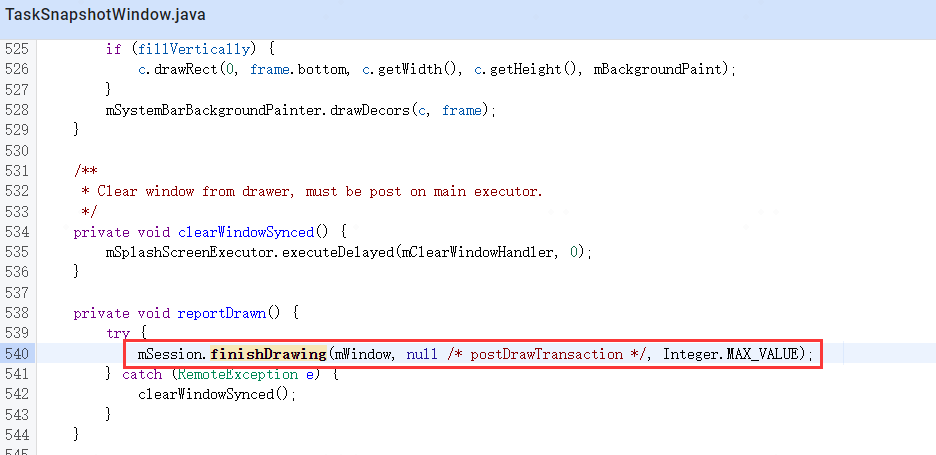
**COMMIT\_DRAW\_PENDING：**

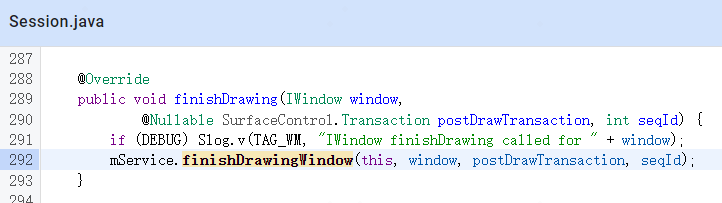
**在执行performDraw中，触发DecorView的draw从根节点一直遍历到叶子节点计算各个View的位置，绘制时要区分是否硬件加速场景，硬件加速场景：注册硬件加速绘制完成的回调，在回调中post一个pendingDrawFinished的任务通知系统绘制完毕。**

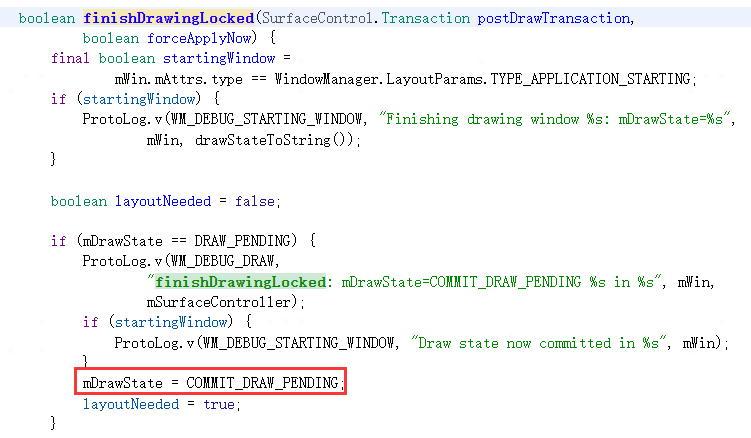
**硬件加速时绘制调用堆栈：**

**ThreadedRenderer.draw >> ThreadedRenderer.updateRootDisplayList >> ThreadedRenderer.updateViewTreeDisplayList >> View.updateDisplayListIfDirty >> View.draw；**

**非硬件加速时绘制 直接调用drawSoftware->View.draw;进行绘制，绘制完毕直接执行pendingDrawFinished不需要异步监听**

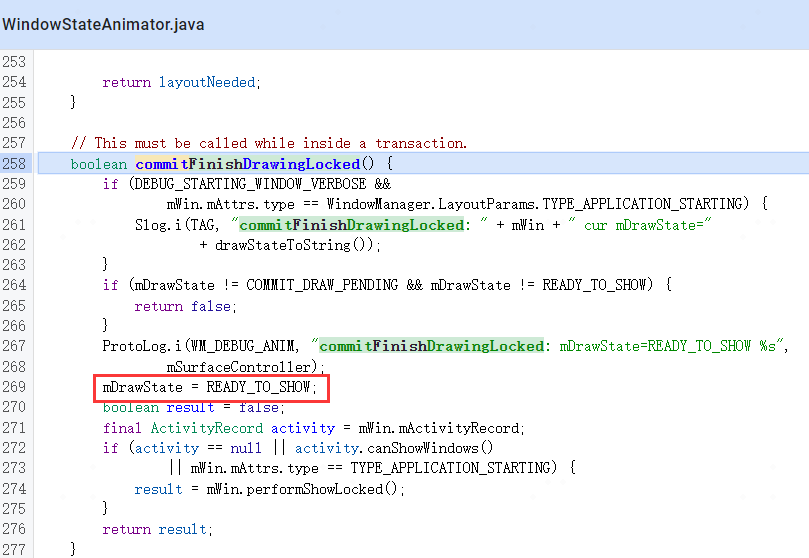






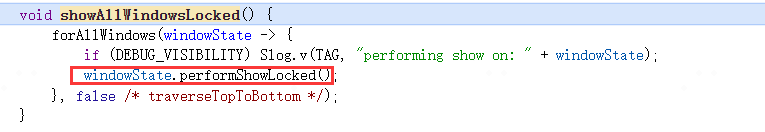
**READY\_TO\_SHOW：**

**finishDrawingWindow方法会触发requestTraversal去post一个Traverser任务（该任务内部最多可有6次循环），该任务被执行时会把WindowState状态mDrawState设置为READY\_TO\_SHOW**

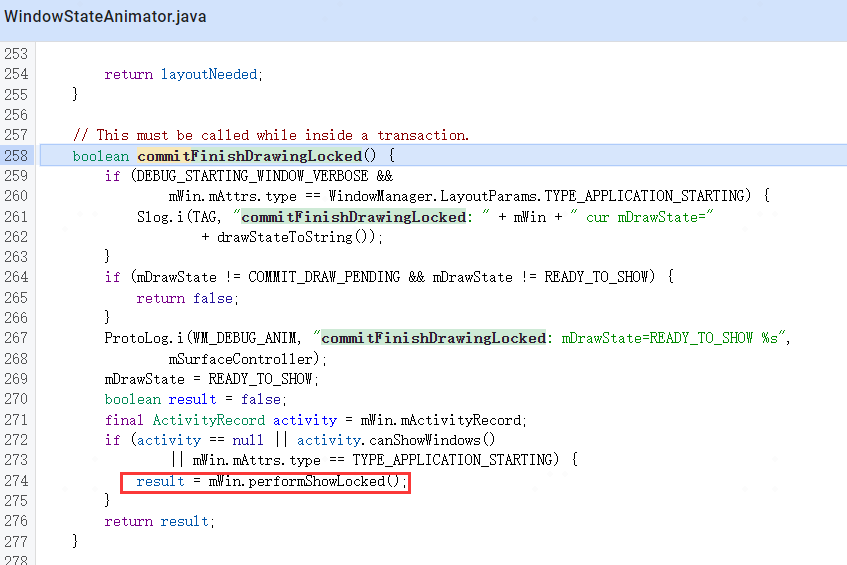


**HAS\_DRAWN：**

**第一种：如果应用内栈内跳转，那么没有StartingWindow，主窗口的最终状态HAS\_DRAWN由该轮Traversal任务执行到GOOD TO GO 流程中ActivityRecord.showAllWindowsLocked去触发WindowState.performShowLocked设置:**

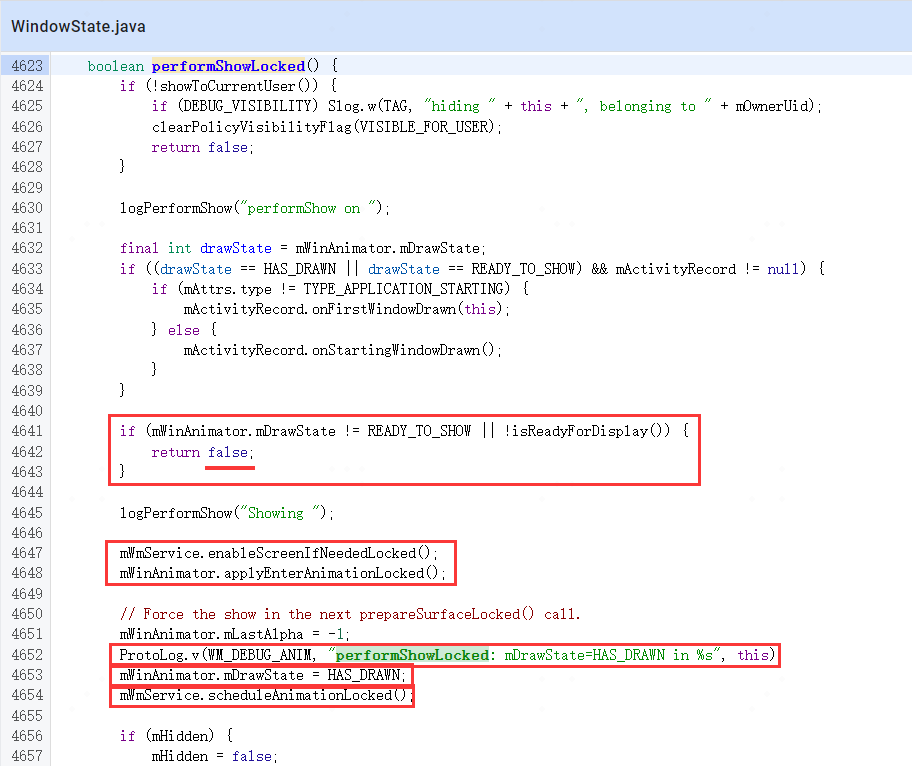


**第二种：如果应用内跨栈跳转，由StartingWindow,由于跨栈跳转场景startingWindow绘制完毕会触发Good To Go ，但此时主窗口还没有绘制完毕，所以此流程会执行的到 GOOD TO GO 之后的主窗口绘制完毕请求的那次Traversal中，主窗口真正绘制完毕那一次requestTraversal请求的Traversal任务并不会触发GOOD TO GO，因为有StartingWindow的页面的GOOD TO GO只会在startingWindow窗口绘制完毕时触发，等主窗口绘制完毕时，此时不会在触发GOOD TO GO，所以这种场景下主窗口最终状态HAS\_DRAWN由WindowStateAnimator.commitFinishDrawingLocked触发WindowState.performShowLocked设置:**



**这一阶段如果状态异常，需要从系统侧出发进行分析。**

**最终都是通过 WindowState.performShowLocked设置**



**Traverser流程图：**

