Android应用程序UI架构

罗升阳

http://weibo.com/shengyangluo

http://blog.csdn.net/luoshengyang

About Me

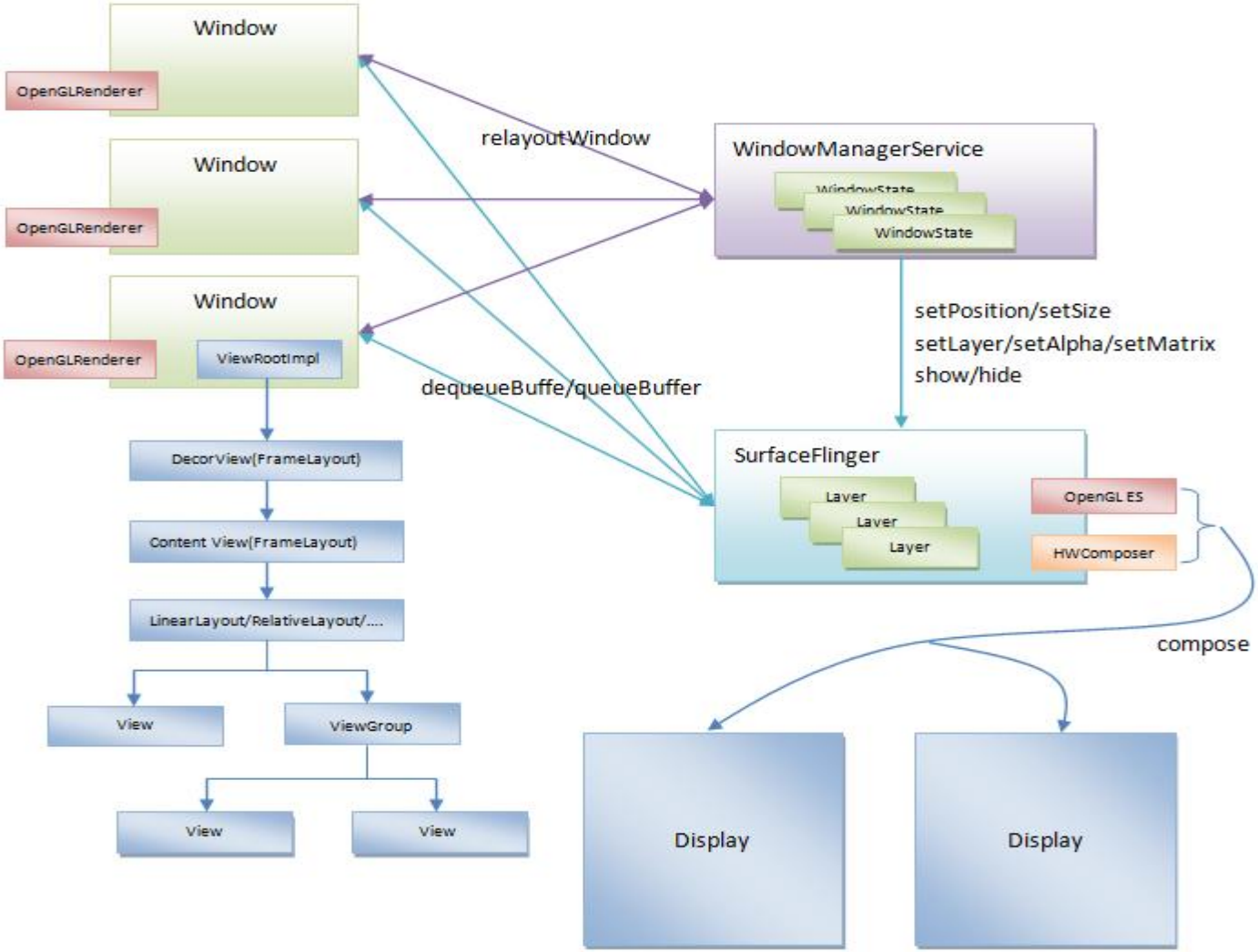
- 《老罗的Android之旅》博客作者
- 《Android系统源代码情景分析》书籍作者
- 博客：http://blog.csdn.net/Luoshengyang
- 微博：http://weibo.com/shengyangluo

Agenda

- Android UI架构概述
- Android应用程序UI框架
- WindowManagerService
- SurfaceFlinger
- Android多屏支持

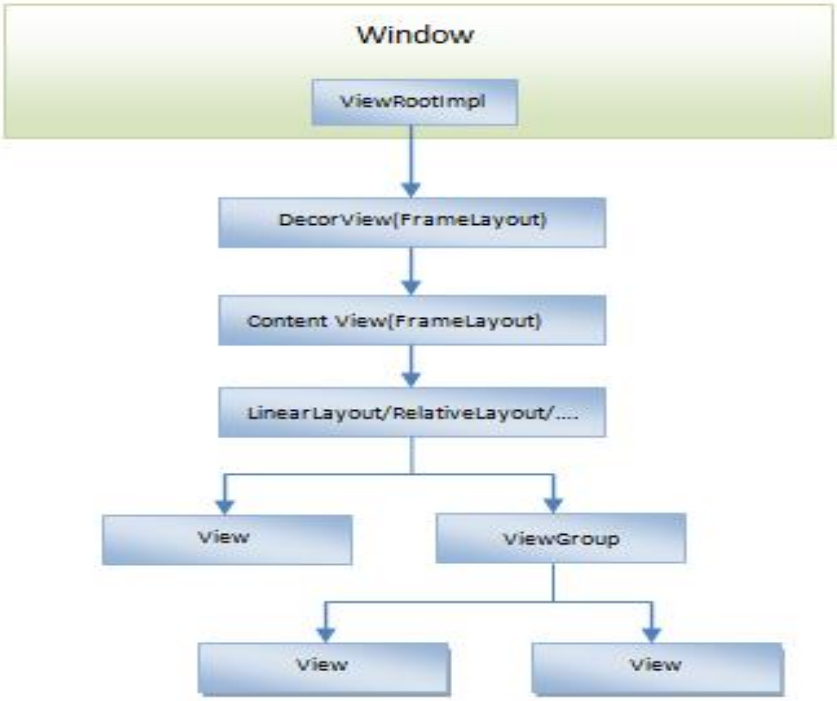
Android UI架构概述

- 总体架构



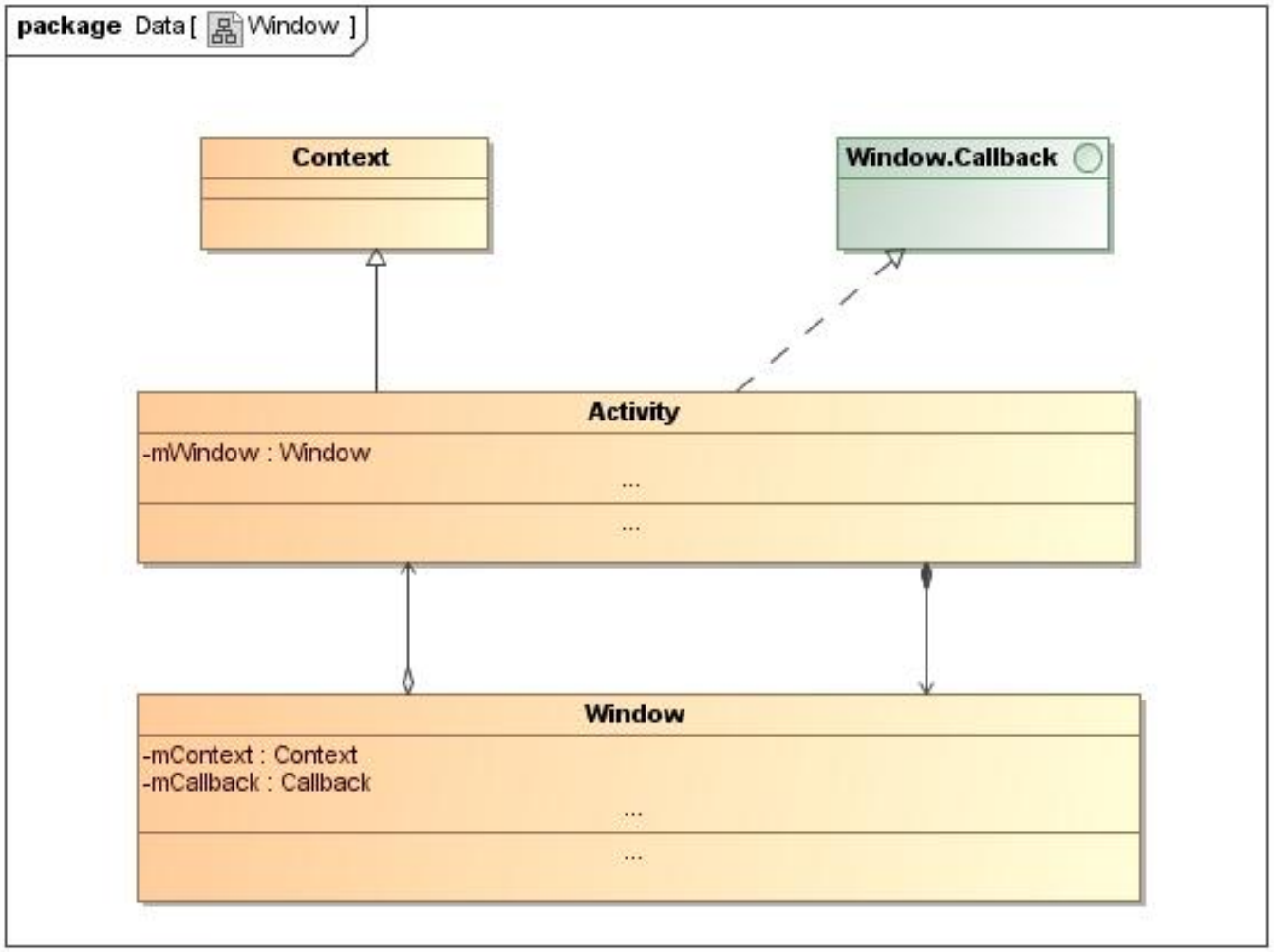
Android应用程序UI框架

- 窗口(Window)的结构
  - ViewRootImpl是一个虚拟根View，用来控制窗口的渲染，以及用来与WindowManagerService、SurfaceFlinger通信
  - DecorView是窗口的真正根View
  - ContentView描述窗口的主题风格



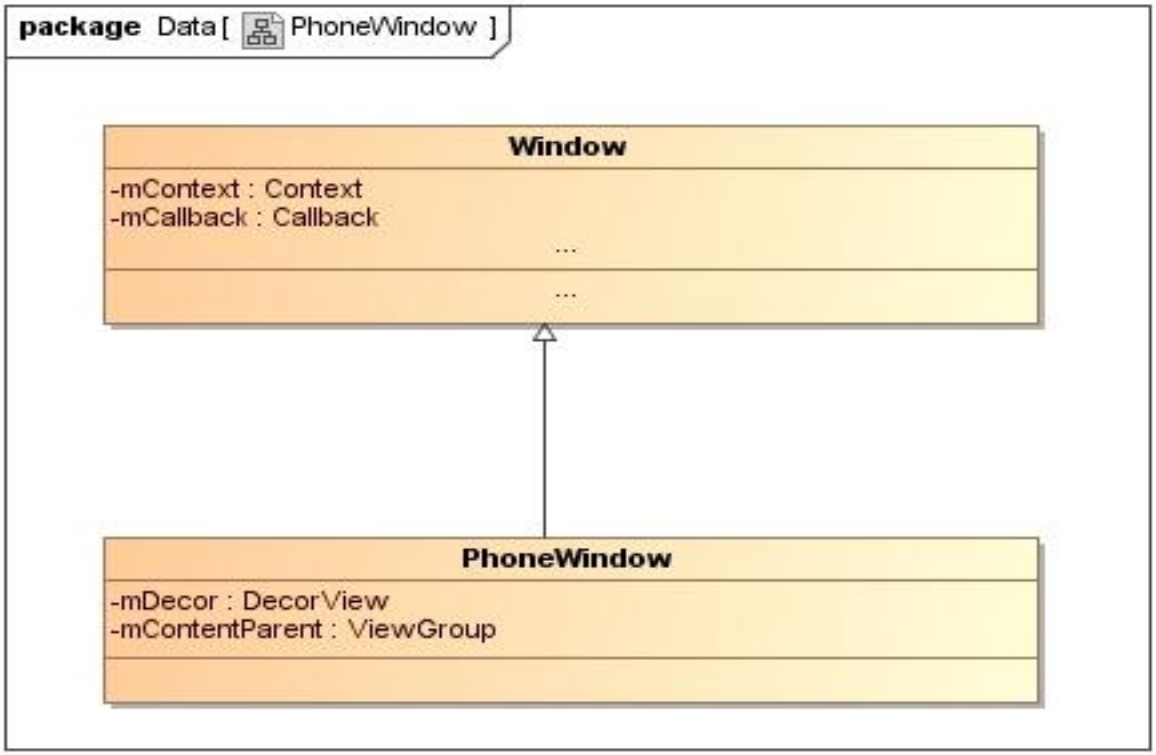
Android应用程序UI框架(续)

- Window与Activity的关系



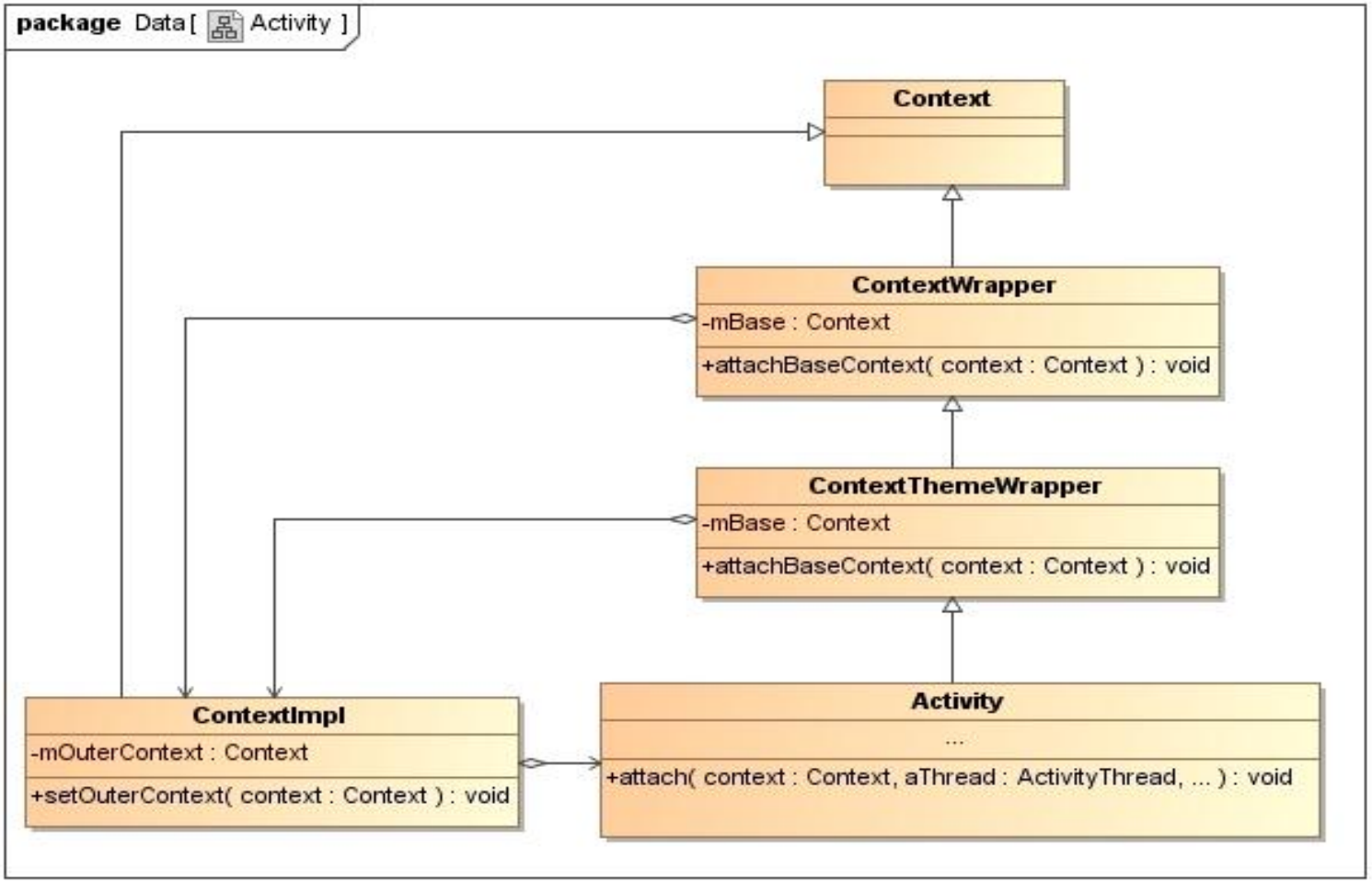
Android应用程序UI框架(续)

- Activity所对应的Window实际上是一个PhoneWindow



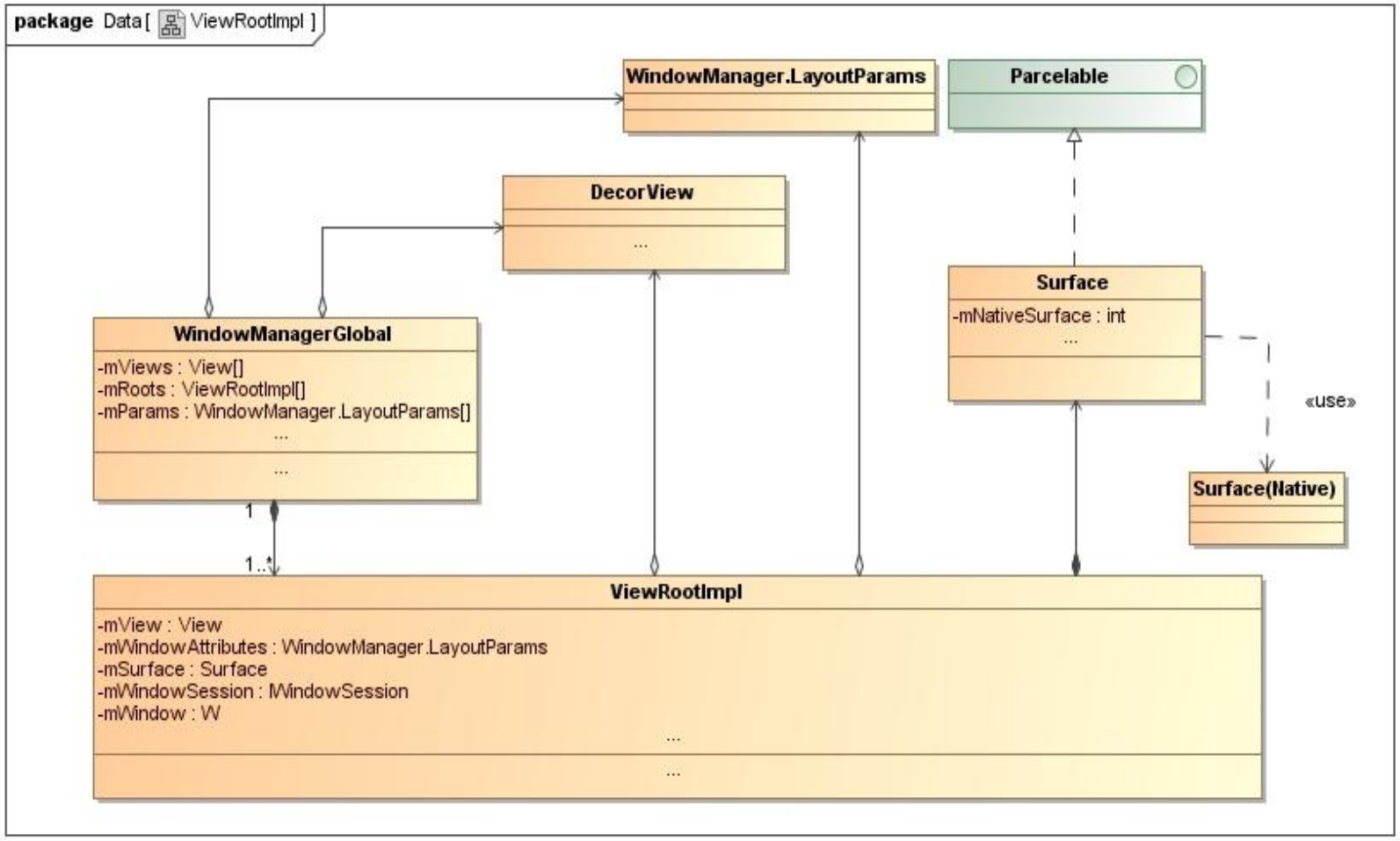
Android应用程序UI框架(续)

- Activity/Window的上下文



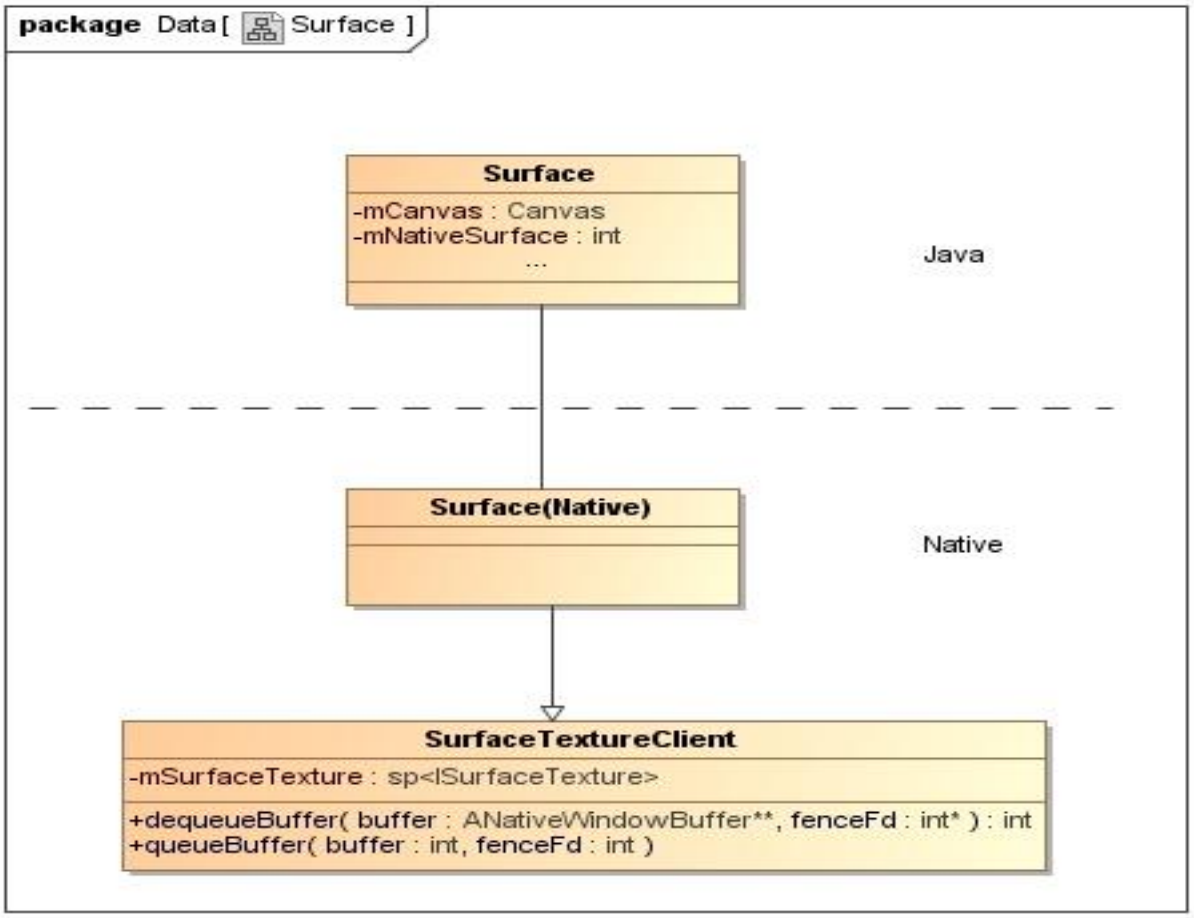
Android应用程序UI框架(续)

- Window的虚拟根View -- ViewRootImpl



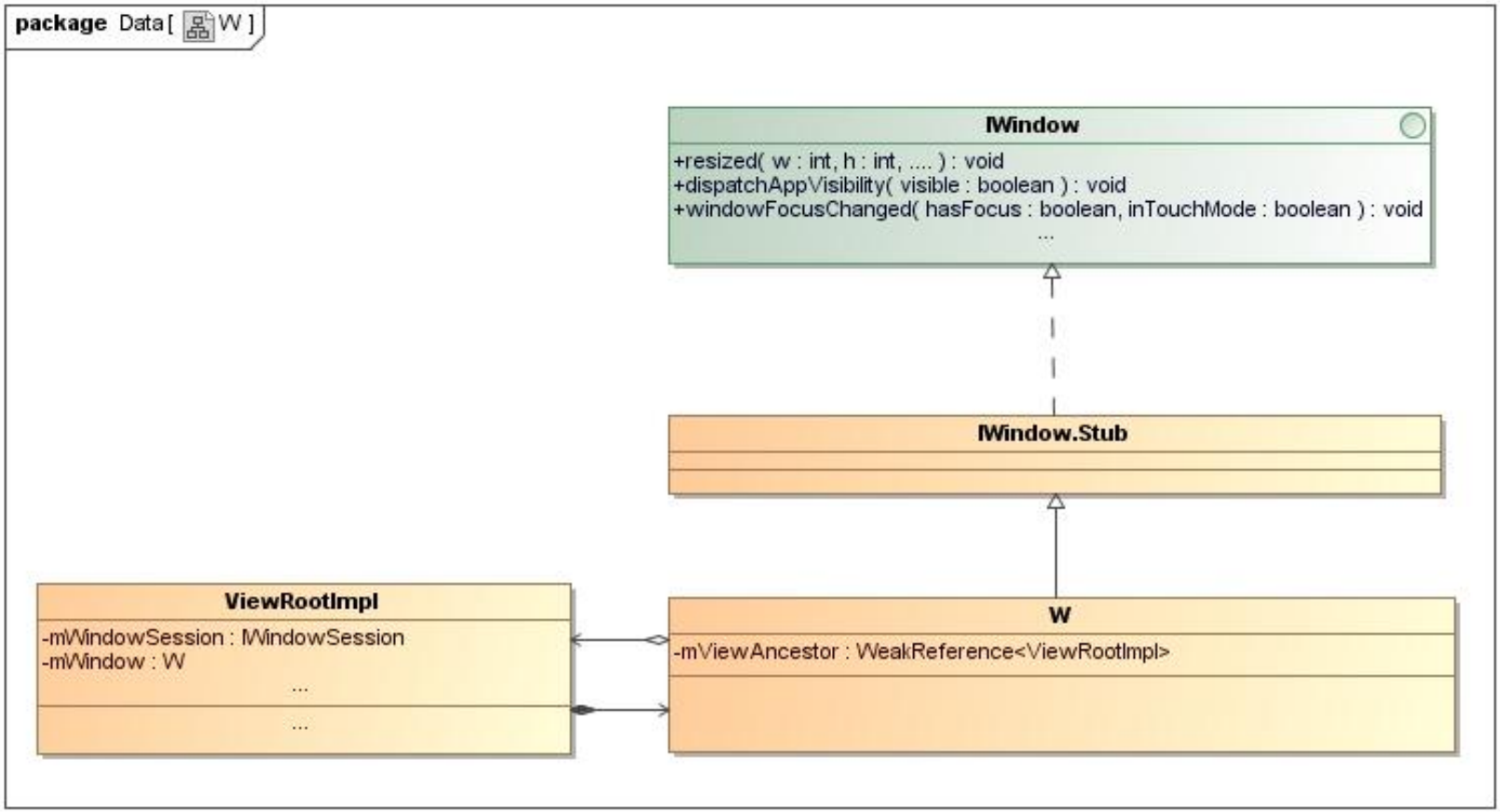
Android应用程序UI框架(续)

- 窗口绘图表面 -- Surface



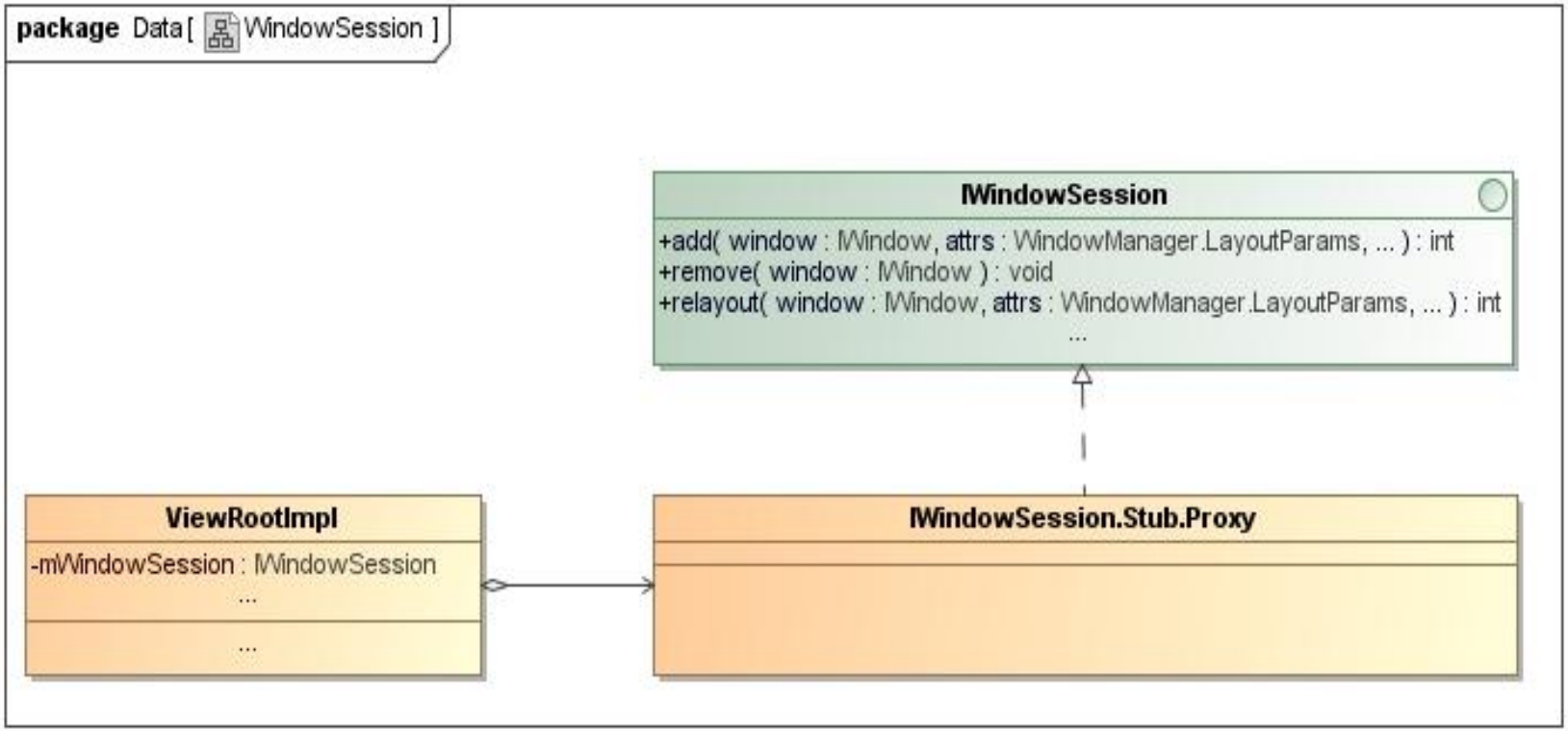
Android应用程序UI框架(续)

- 窗口标志 -- W



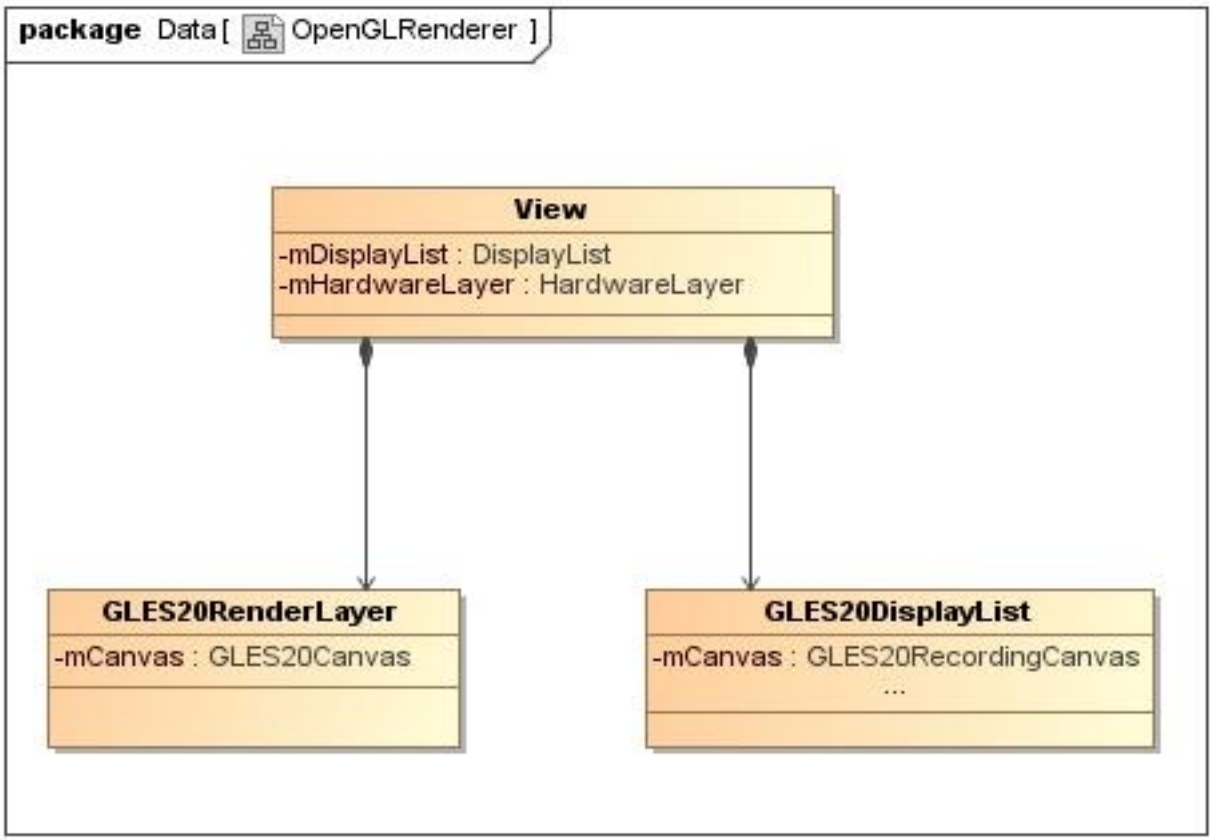
Android应用程序UI框架(续)

- 窗口会话 -- Session



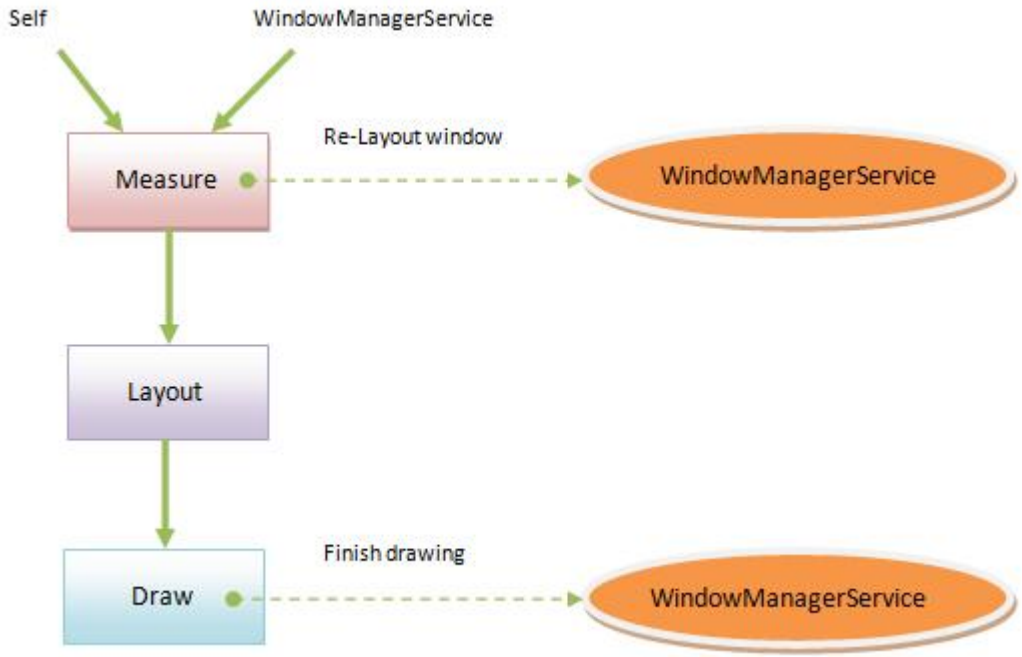
Android应用程序UI框架(续)

- 窗口视图 -- View



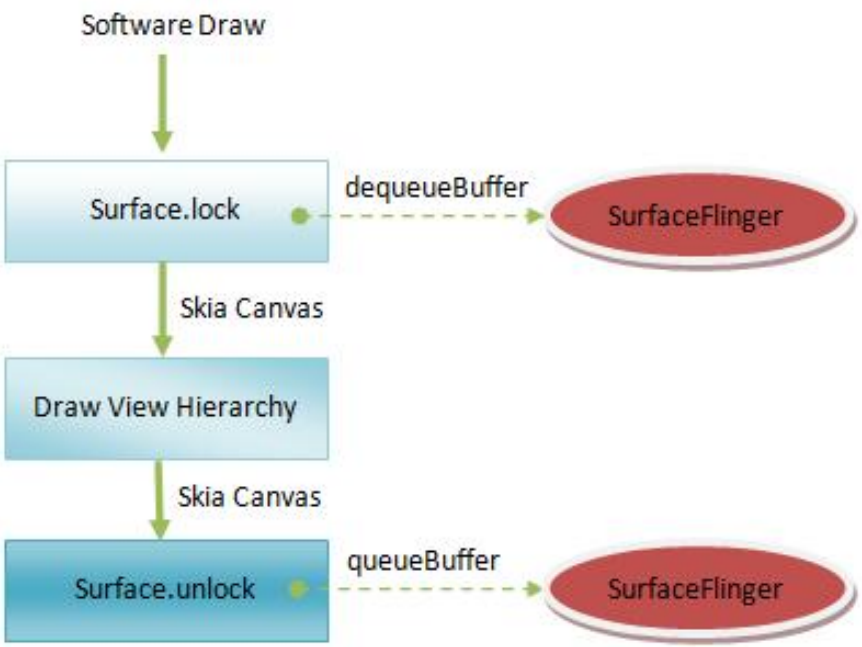
Android应用程序UI框架(续)

- Android应用程序UI的绘制过程



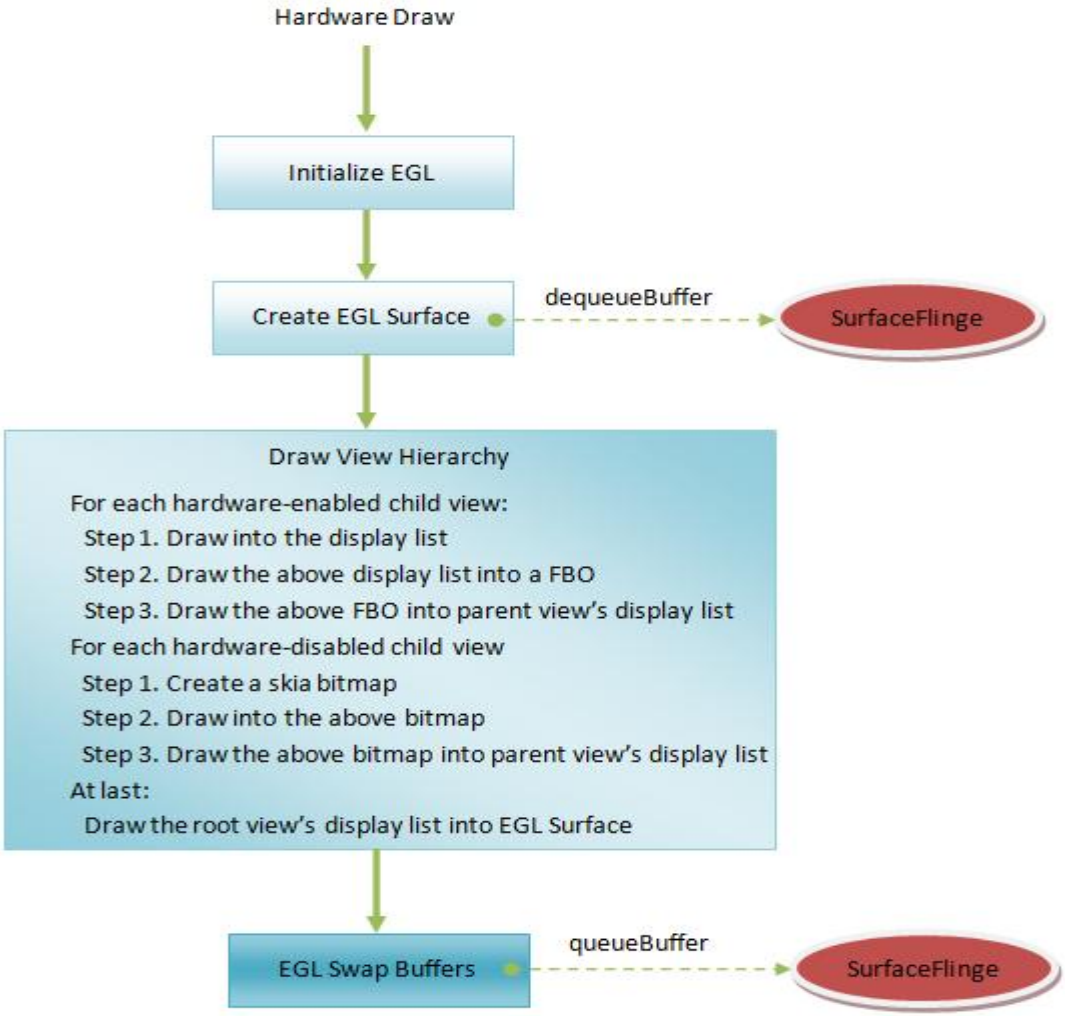
Android应用程序UI框架(续)

- 软件渲染过程



Android应用程序UI框架(续)

- 硬件渲染过程

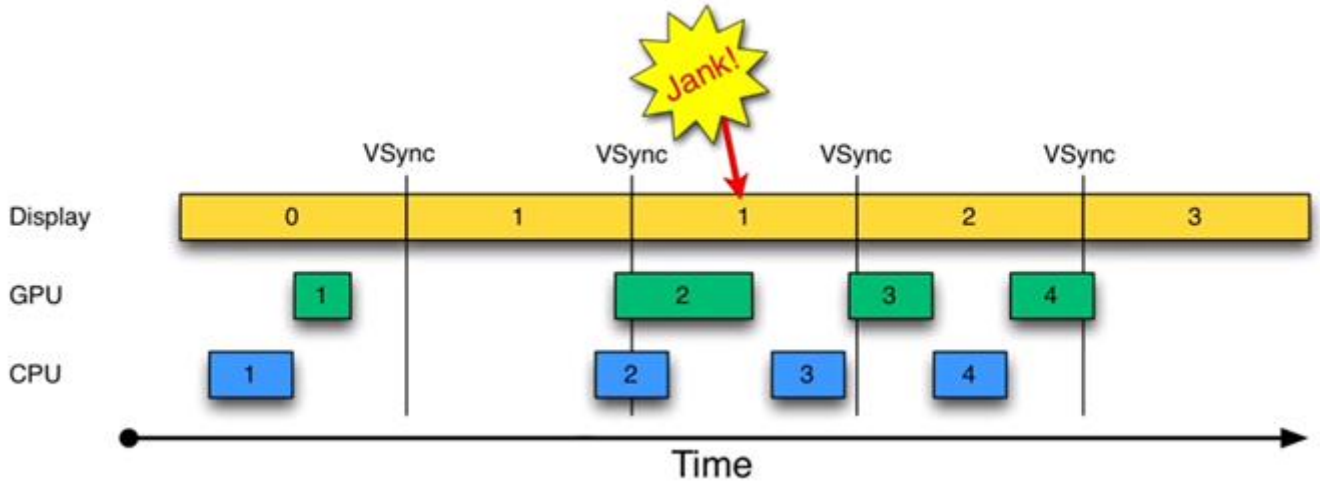


Android应用程序UI框架(续)

- Display List是什么？
  - Display List是一个缓存绘制命令的Buffer
- Display List的好处？
  - 当View的某些属性发生改变时，只需要修改相应的Buffer中对应的属即可，例如Alpha属性，而无需对整个View进行重绘

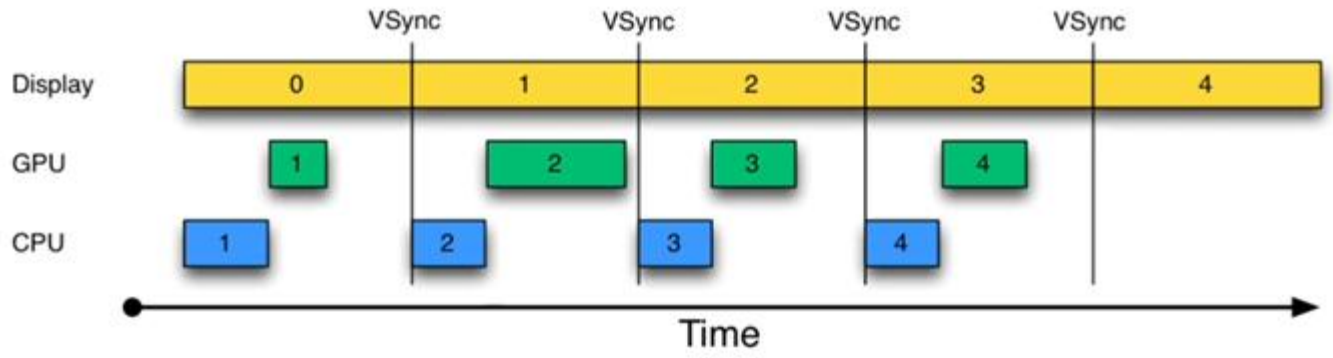
Android应用程序UI框架(续)

- Android应用程序UI的绘制时机 – Without Vsync -- Jank



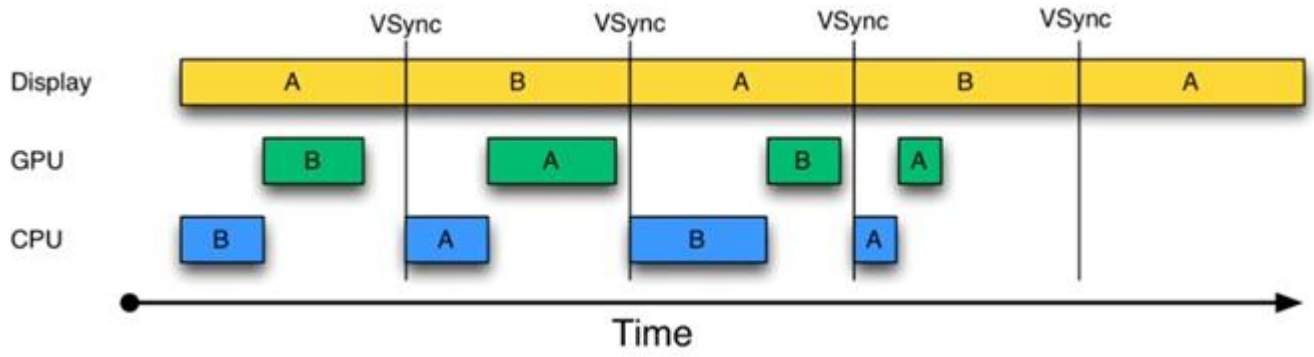
Android应用程序UI框架(续)

- Android应用程序UI的绘制时机 – With VSync



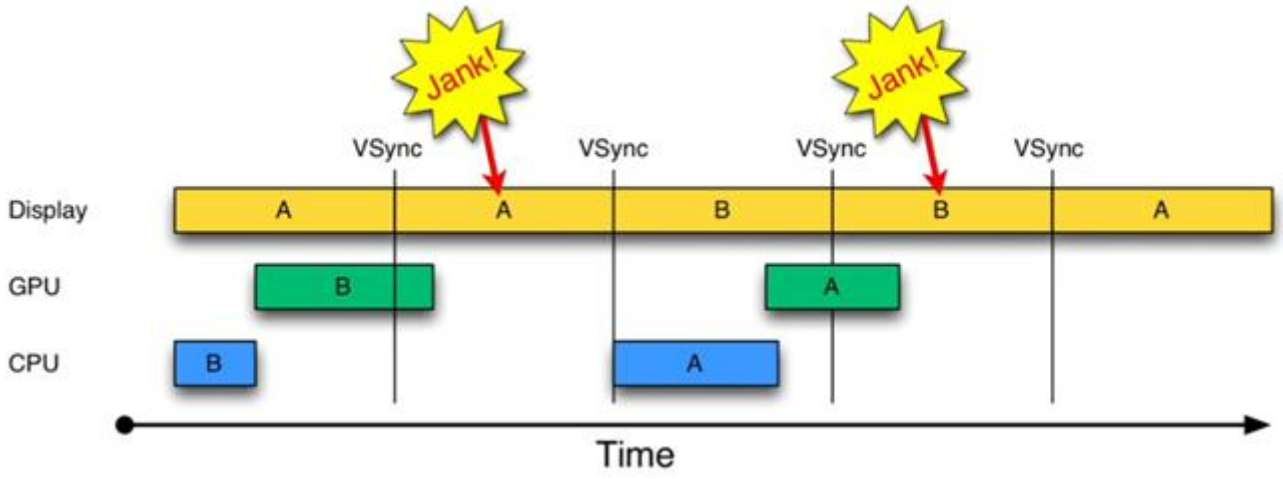
Android应用程序UI框架(续)

- Android应用程序UI的绘制时机 – With Vsync and Double Buffering



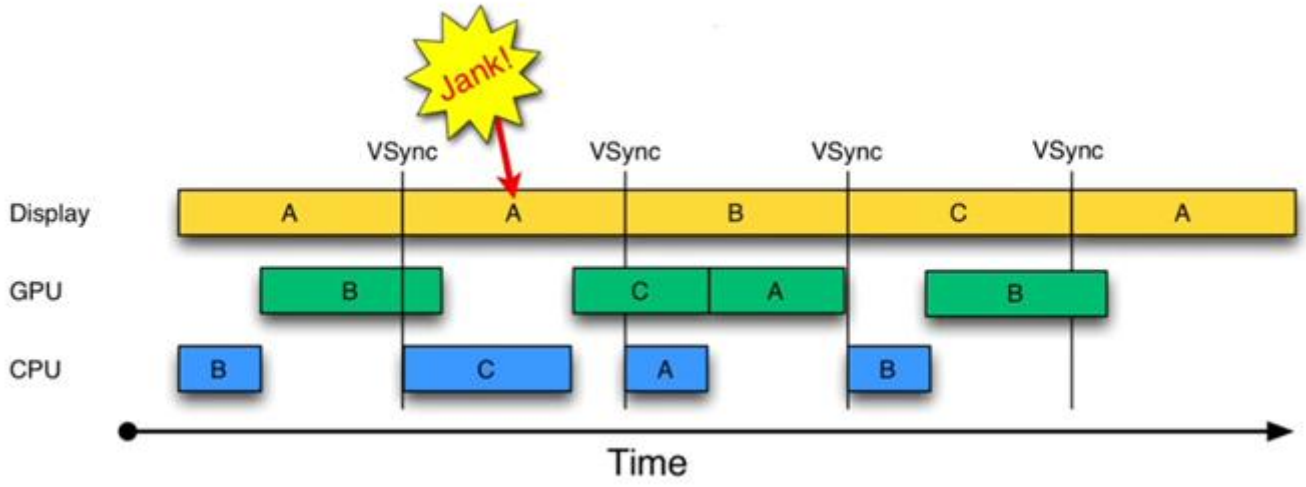
Android应用程序UI框架(续)

- Android应用程序UI的绘制时机 – With Vsync and Double Buffering -- Jank



Android应用程序UI框架(续)

- Android应用程序UI的绘制时机 – With Vsync and Triple Buffering



Android应用程序UI框架(续)

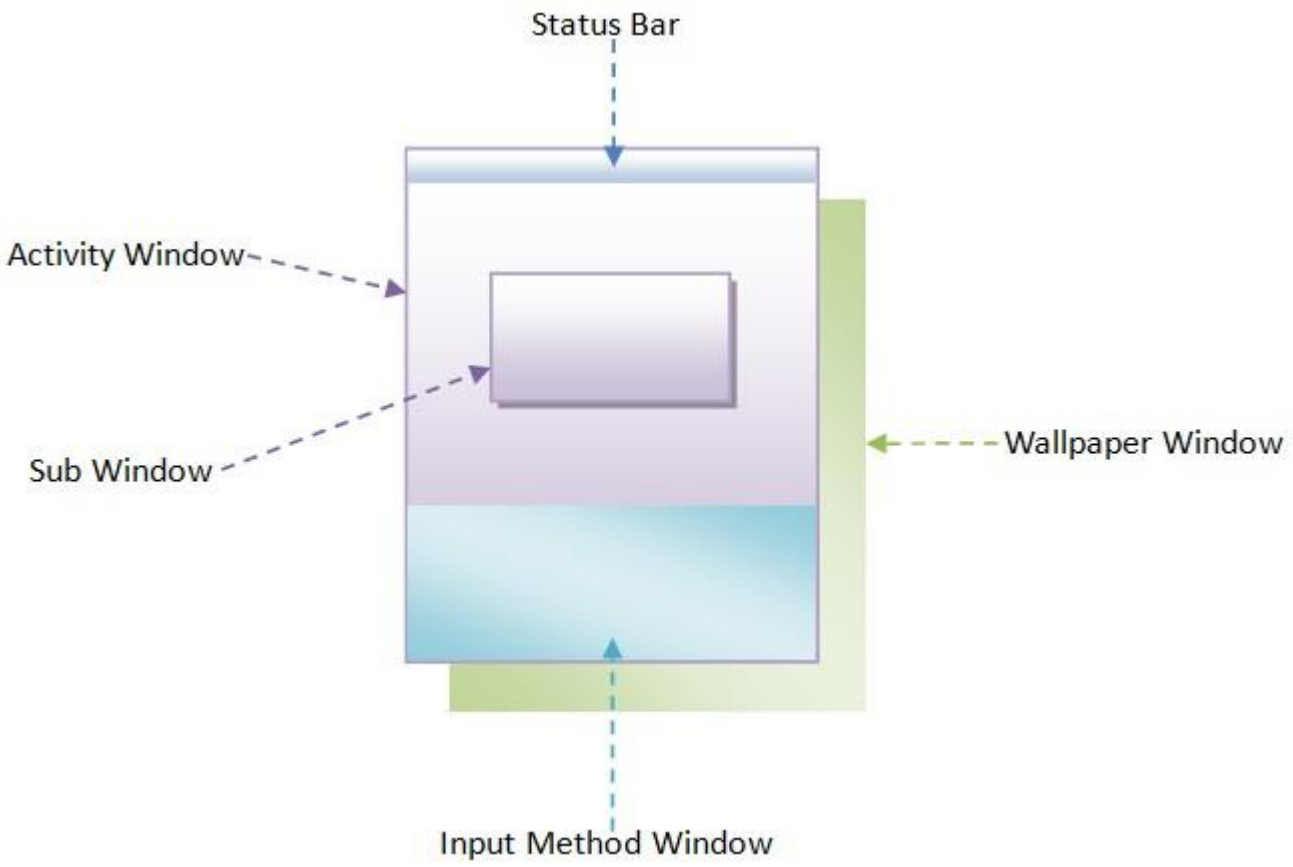
- Android系统的VSync实现
  - SurfaceFlinger内部维护有一个EventThread，用来监控显卡的VSync事件
  - Android应用程序通过注册一个DisplayEventReceiver来接收SurfaceFlinger的VSync事件
  - Android应用程序接收到重绘UI请求，通过前面注册的DisplayEventReceiver向SurfaceFlinger请求在下一个VSync事件到来时产生一个VSync通知
  - Android应用程序获得VSync通知的时候，才会真正执行重绘UI的请求

WindowManagerService

- 职责
  - 计算窗口大小
  - 计算窗口z轴位置
  - 管理输入法窗口
  - 管理壁纸窗口
  - 执行窗口切换

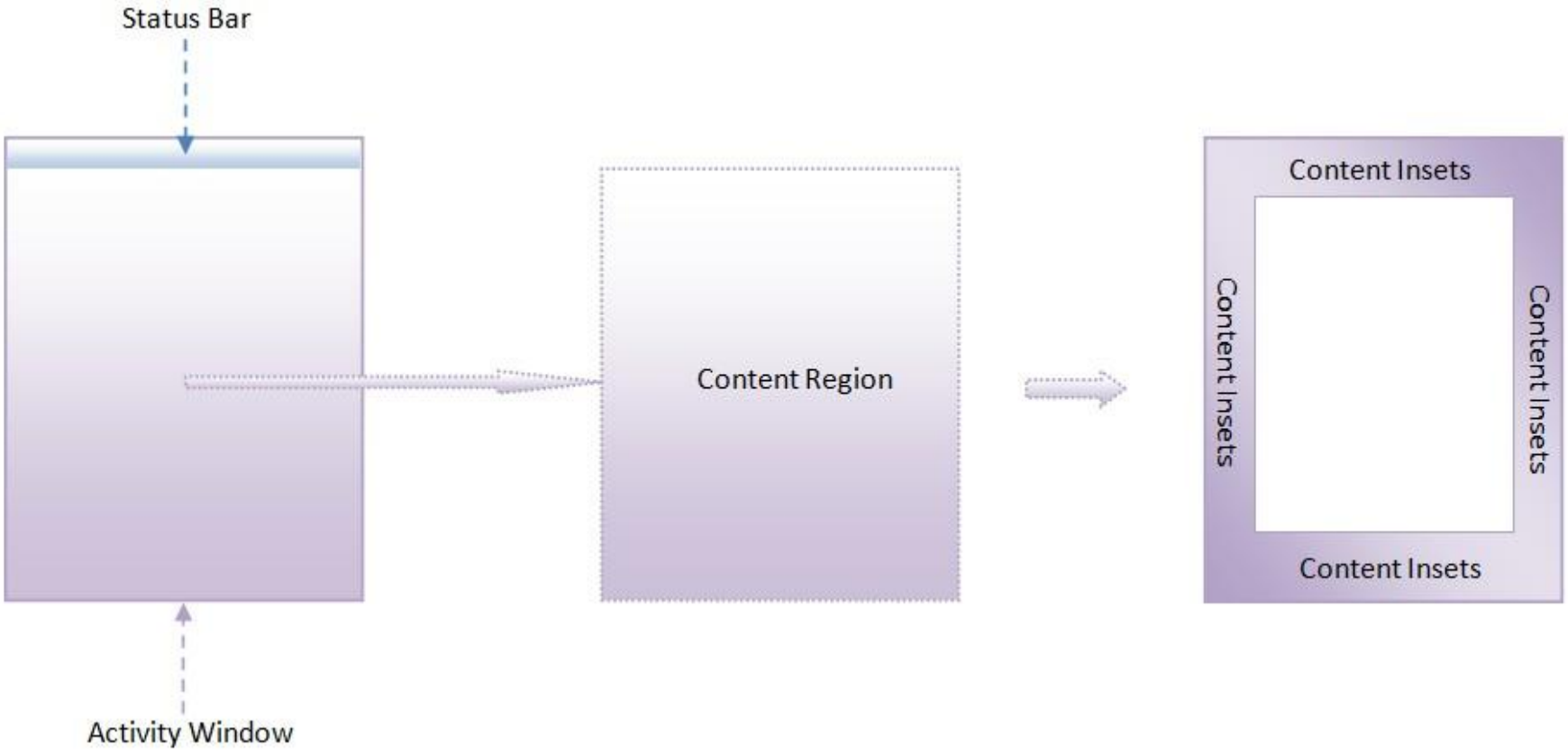
WindowManagerService

- 屏幕的基本结构



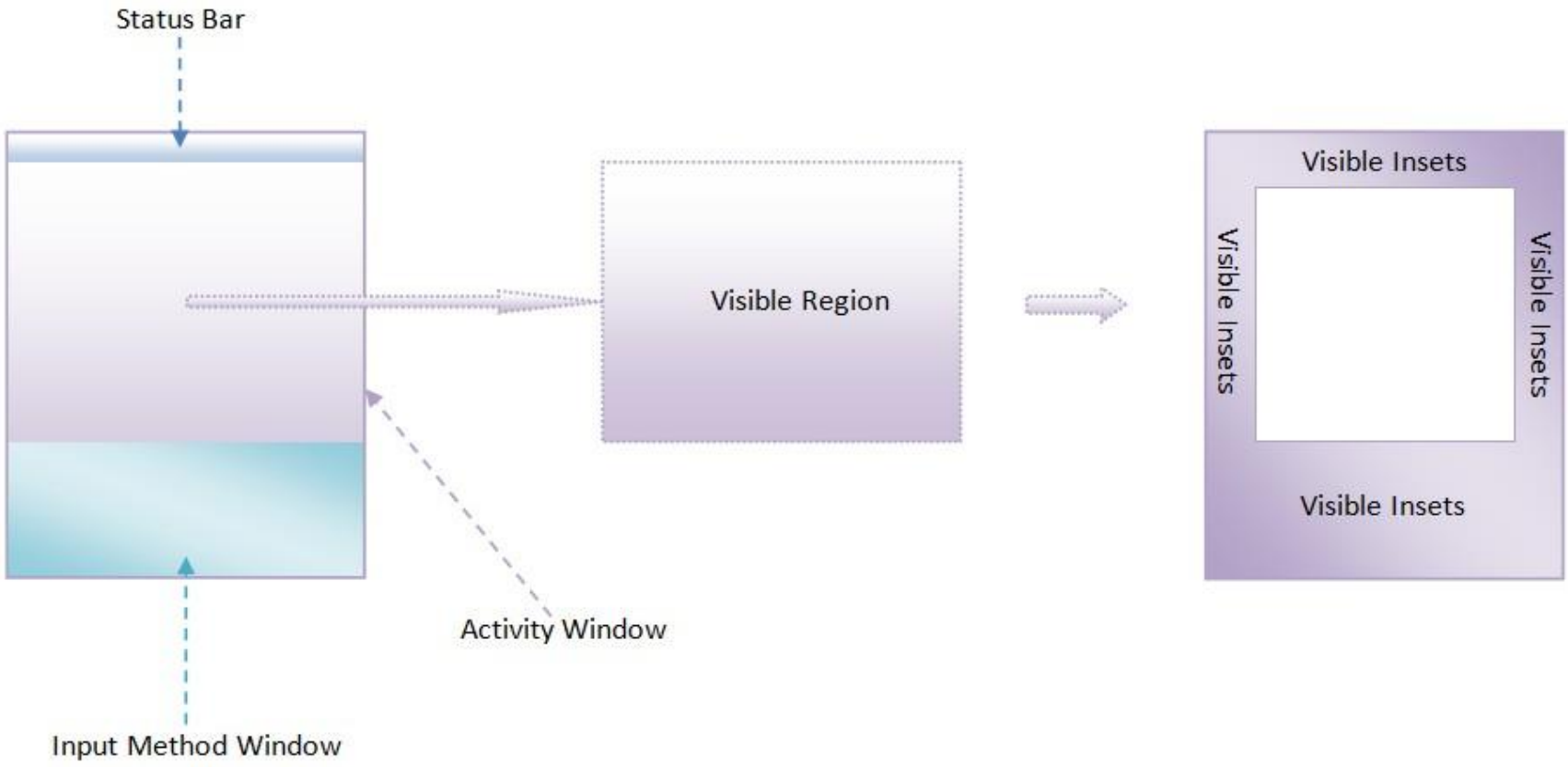
WindowManagerService

- 计算窗口大小 – Content Region



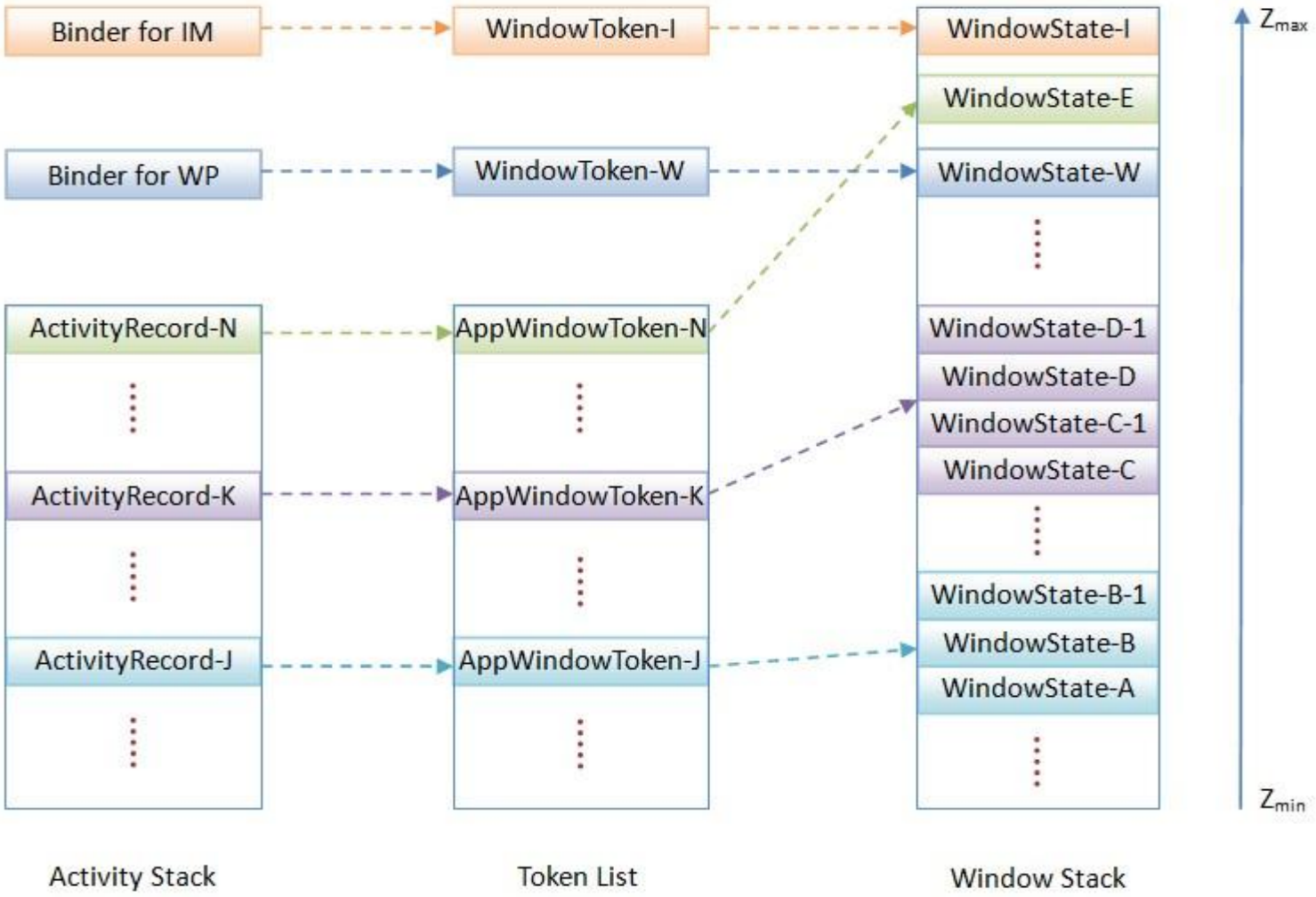
WindowManagerService

- 计算窗口大小 – Visible Region



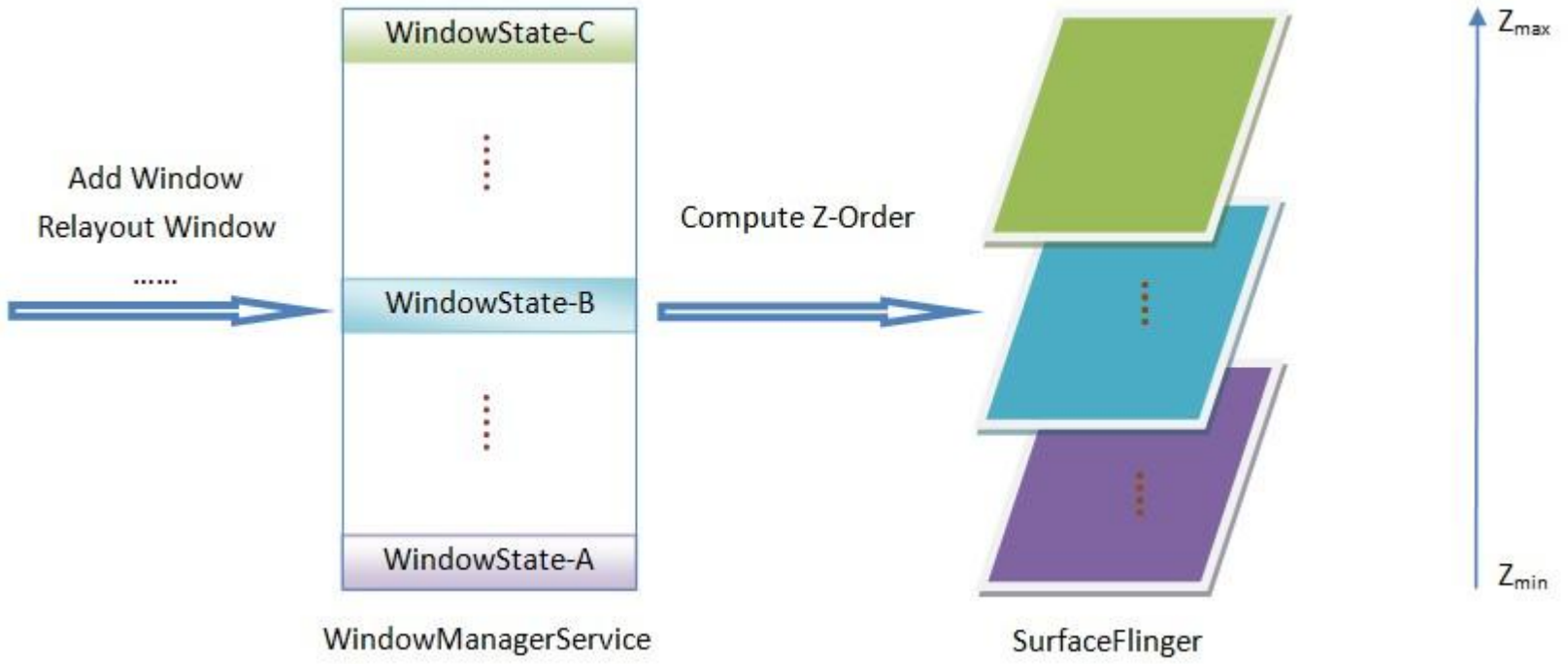
WindowManagerService

- 计算窗口Z轴位置 – Window Stack



WindowManagerService

- 计算窗口Z轴位置 – 计算时机



WindowManagerService

- 计算窗口Z轴位置 – 计算公式

Z = Base Layer + WINDOW_LAYER_MULTIPLIER(5)

Base Layer = T * TYPE_LAYER_MULTIPLIER(10000) + TYPE_LAYER_OFFSET(1000)

WindowManagerService

- 计算窗口Z轴位置 – 窗口主类型

Window Type	Layer Value	Description
TYPE_STATUS_BAR(2000)	STATUS_BAR_LAYER(8)	状态栏
TYPE_STATUS_BAR_PANEL(2014)	STATUS_BAR_PANEL_LAYER(5)	从状态栏滑出来的面板
TYPE_SYSTEM_DIALOG(2008)	SYSTEM_DIALOG_LAYER(6)	从状态栏滑出来的面板
TYPE_SEARCH_BAR(2001)	SEARCH_BAR_LAYER(4)	搜索栏
TYPE_PHONE(2002)	PHONE_LAYER(3)	拨号窗口
TYPE_KEYGUARD(2004)	KEYGUARD_LAYER(14)	锁屏窗口
TYPE_KEYGUARD_DIALOG(2009)	KEYGUARD_DIALOG_LAYER(15)	锁屏窗口显示的对话框
TYPE_SYSTEM_ALERT(2003)	SYSTEM_ALERT_LAYER(10)	系统警告窗口
TYPE_SYSTEM_ERROR(2010)	SYSTEM_ERROR_LAYER(11)	系统错误窗口
TYPE_INPUT_METHOD(2011)	INPUT_METHOD_LAYER(12)	系统输入法窗口
TYPE_INPUT_METHOD_DIALOG(2012)	INPUT_METHOD_DIALOG_LAYER(13)	系统输入法对话框，位于系统输入法窗口之上
TYPE_SYSTEM_OVERLAY(2006)	SYSTEM_OVERLAY_LAYER(16)	重叠窗口
TYPE_SECURE_SYSTEM_OVERLAY(2015)	SECURE_SYSTEM_OVERLAY_LAYER(17)	重叠窗口，但是只有系统可以创建，应用程序不可以创建
TYPE_PRIORITY_PHONE(2007)	PRIORITY_PHONE_LAYER(9)	拨号窗口，可以位于锁屏窗口之上
TYPE_TOAST(2005)	TOAST_LAYER(7)	短暂停留的通知窗口
TYPE_WALLPAPER(2013)	WALLPAPER_LAYER(2)	壁纸窗口
TYPE_APPLICATION_*	APPLICATION_LAYER(2)	应用程序窗口

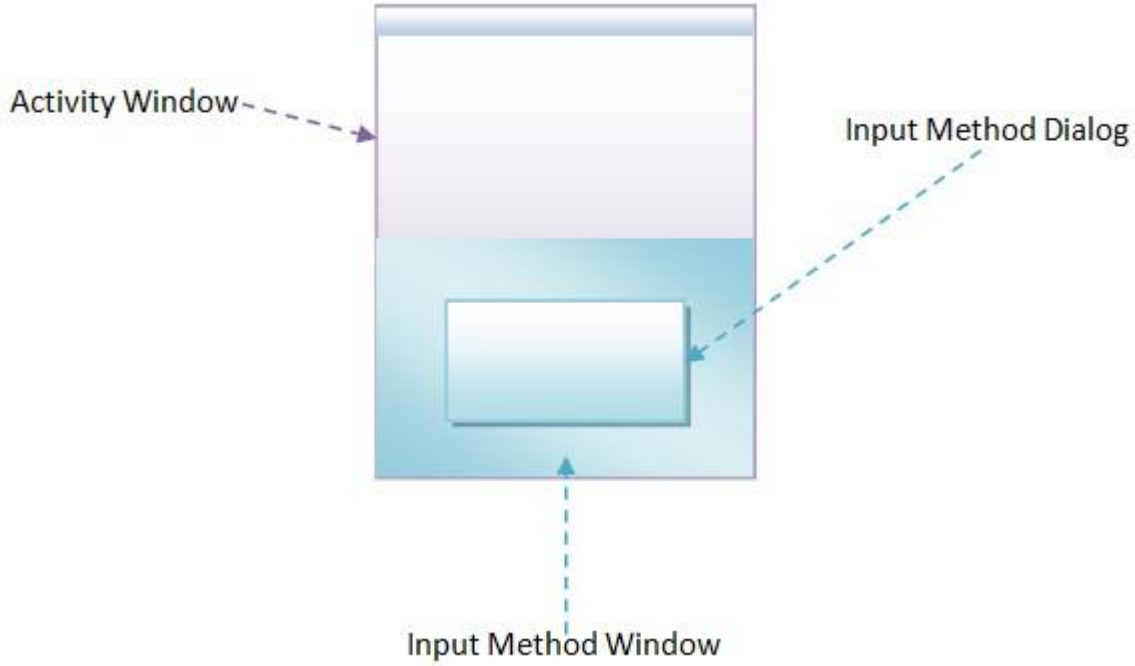
WindowManagerService

- 计算窗口Z轴位置 – 窗口子类型

Window Type	Sub Layer Value	Description
TYPE_APPLICATION_PANEL(1000)	APPLICATION_PANEL_SUBLAYER(1)	附加在其它窗口之上的窗口
TYPE_APPLICATION_ATTACHED_DIALOG(1003)	APPLICATION_PANEL_SUBLAYER(1)	同TYPE_APPLICATION_PANEL，但是它是以顶层窗口出现的，不附加在任何一个窗口之上
TYPE_APPLICATION_MEDIA	APPLICATION_MEDIA_SUBLAYER(-2)	多媒体窗口
TYPE_APPLICATION_MEDIA_OVERLAY	APPLICATION_MEDIA_OVERLAY_SUBLAYER(-1)	显示在多媒体窗口上面的重叠窗口
TYPE_APPLICATION_SUB_PANEL	APPLICATION_SUB_PANEL_SUBLAYER(2)	附加在其它窗口的窗口，它同时可以附加在类型为TYPE_APPLICATION_PANEL的窗口之上

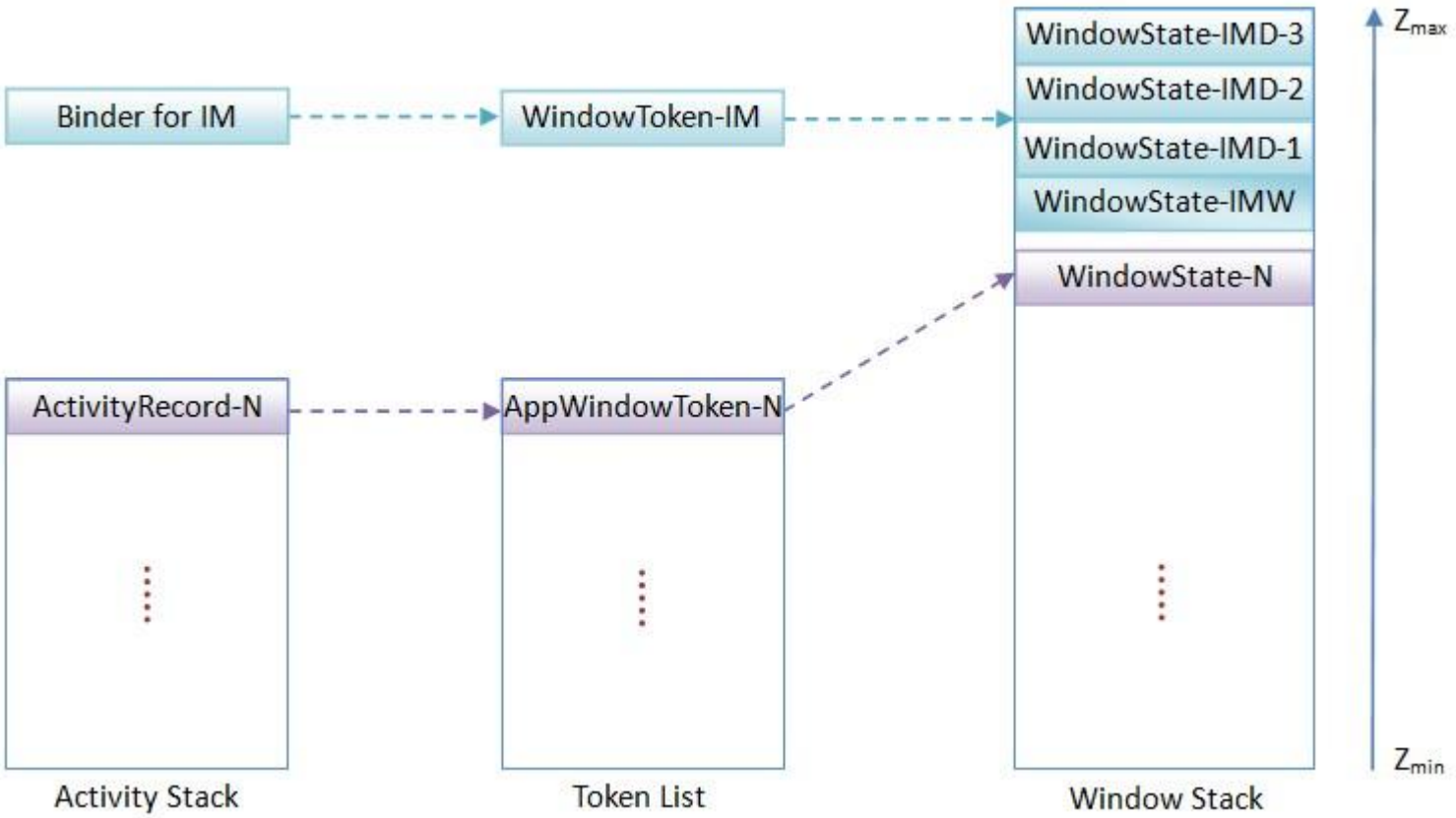
WindowManagerService

- 管理输入法窗口



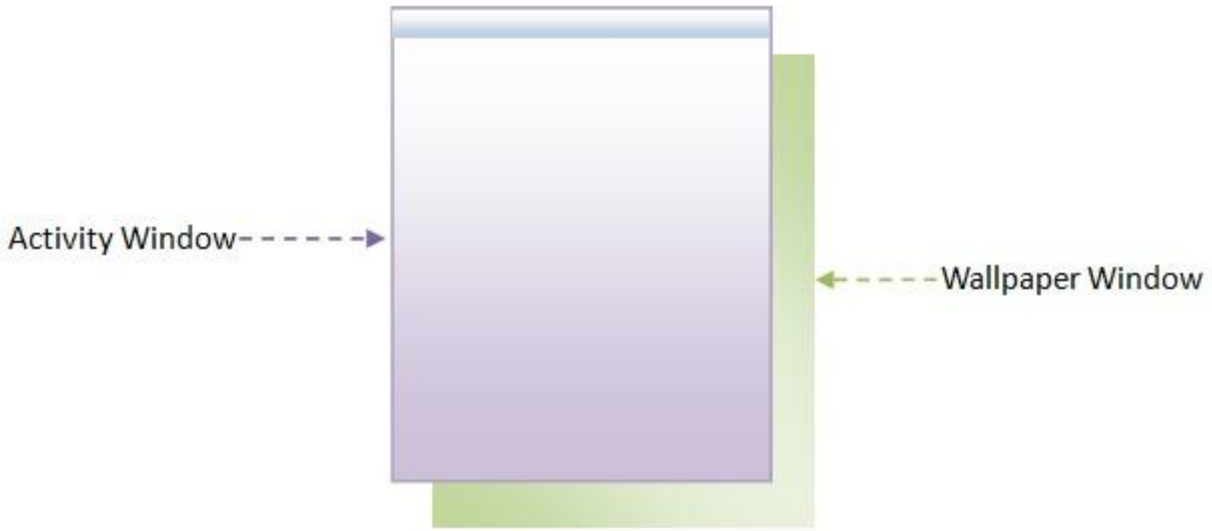
WindowManagerService

- 输入法窗口在Window Stack的位置



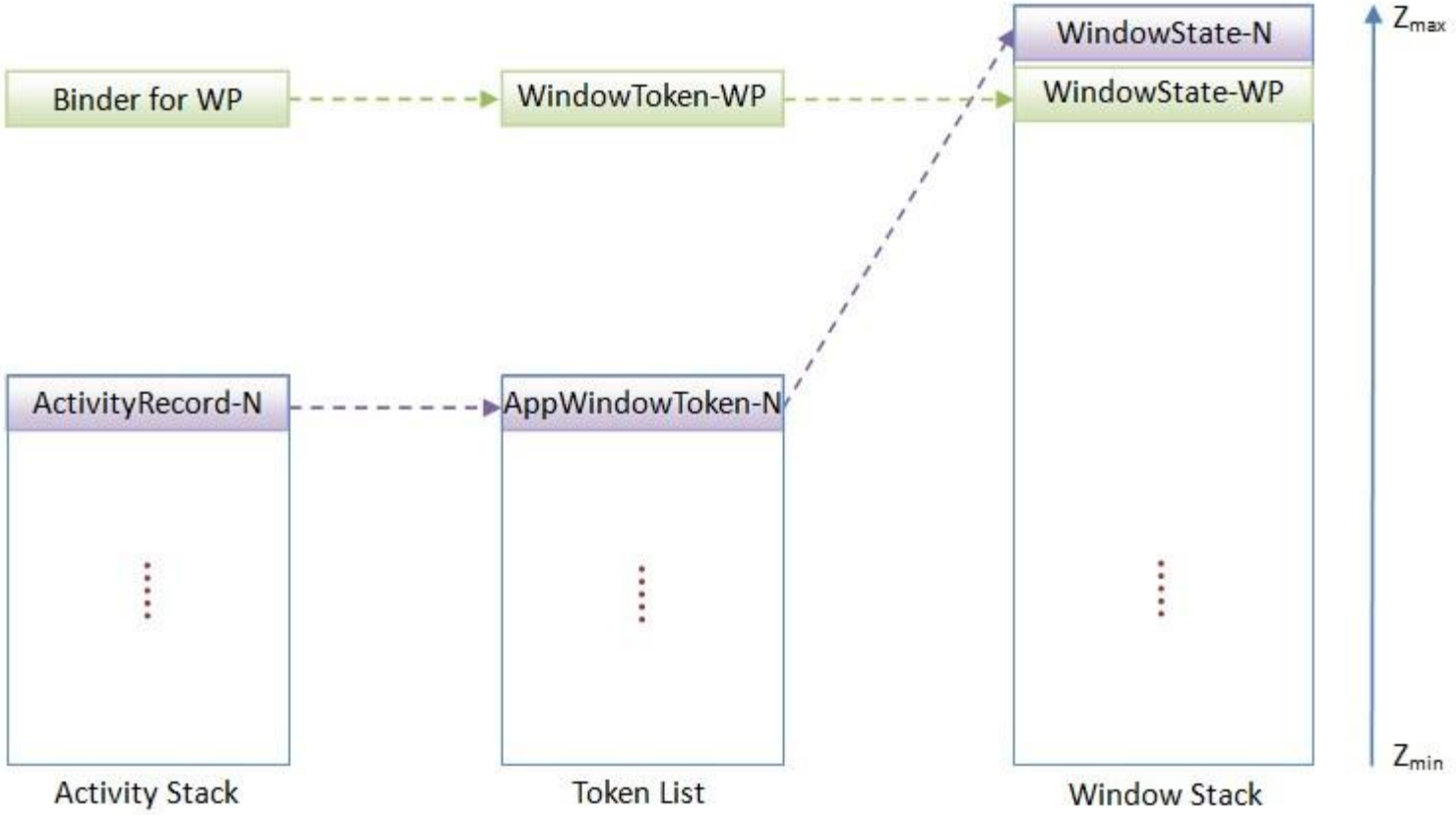
WindowManagerService

- 管理壁纸窗口



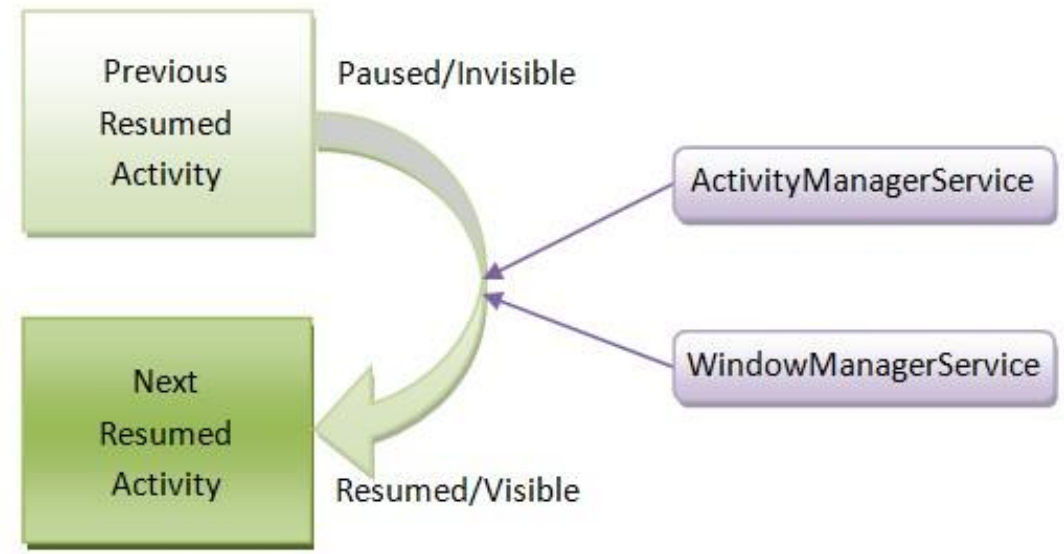
WindowManagerService

- 壁纸窗口在Window Stack的位置



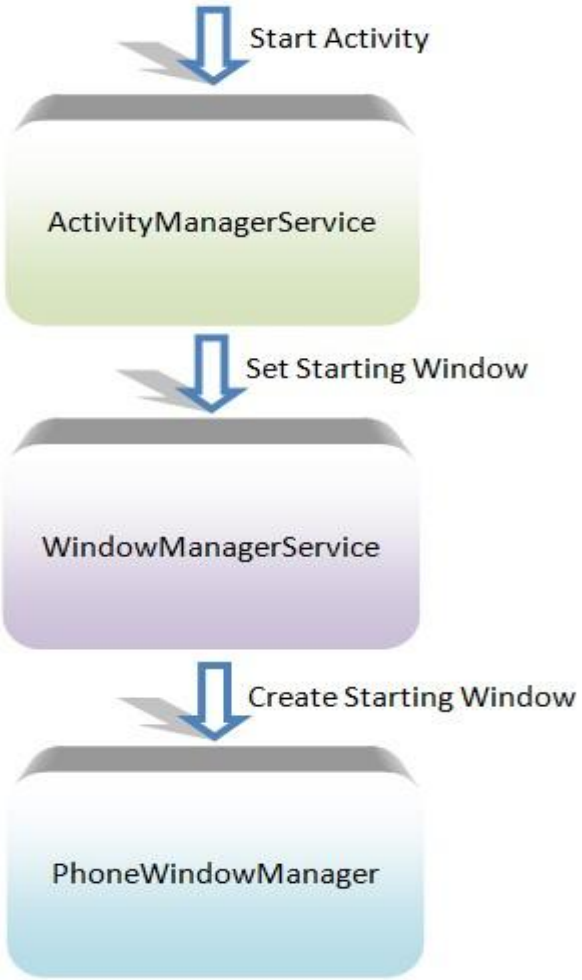
WindowManagerService

- 执行窗口切换



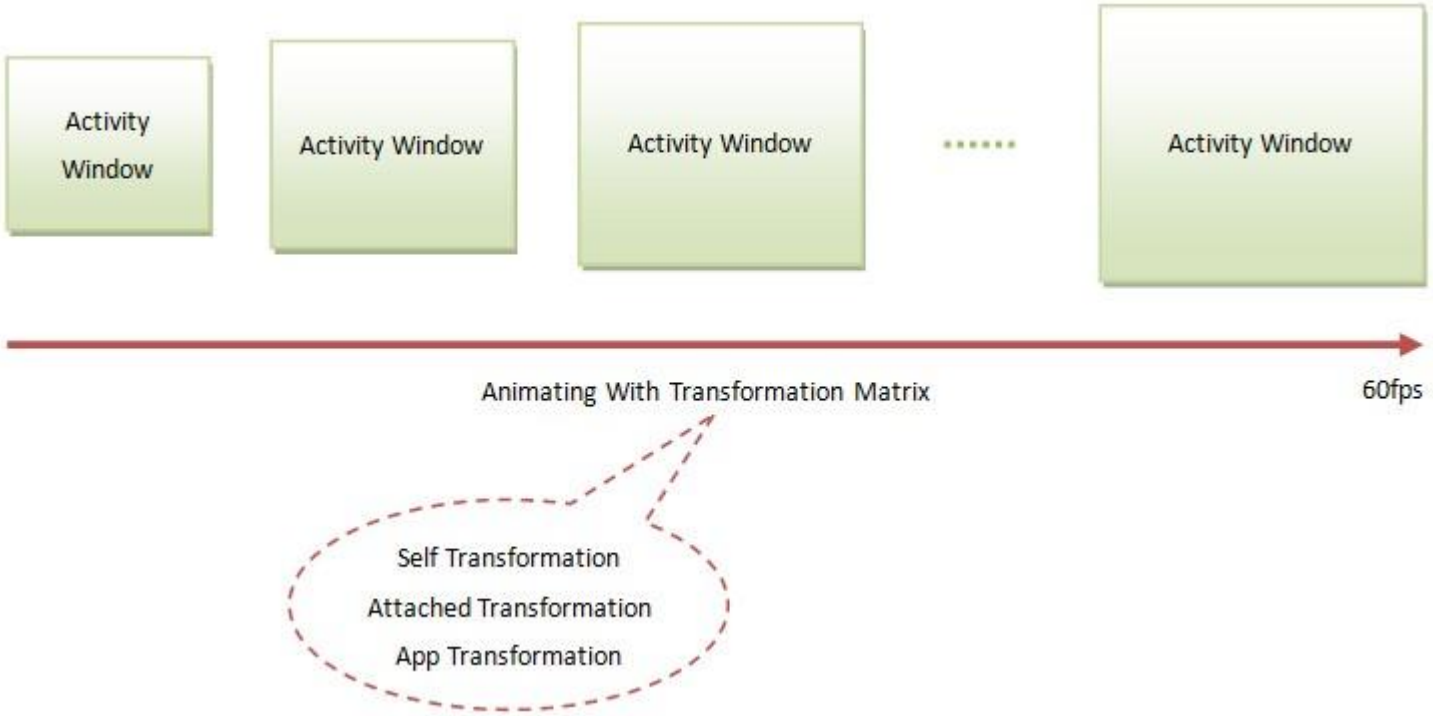
WindowManagerService

- 执行窗口切换 – Starting Window



WindowManagerService

- 执行窗口切换 – 动画

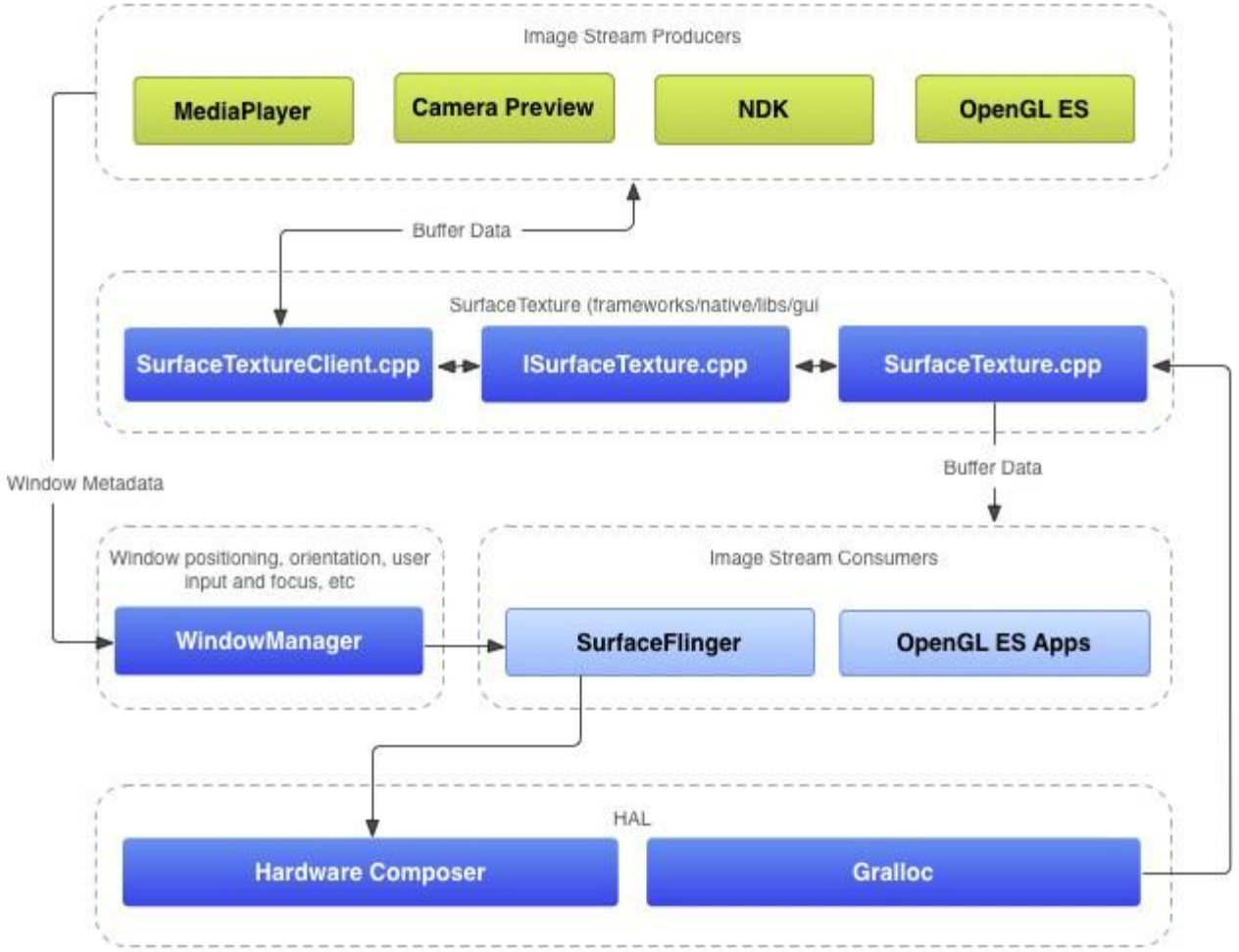


SurfaceFlinger

- 职责
  - 分配图形缓冲区
  - 合成图形缓冲区
  - 管理VSync事件

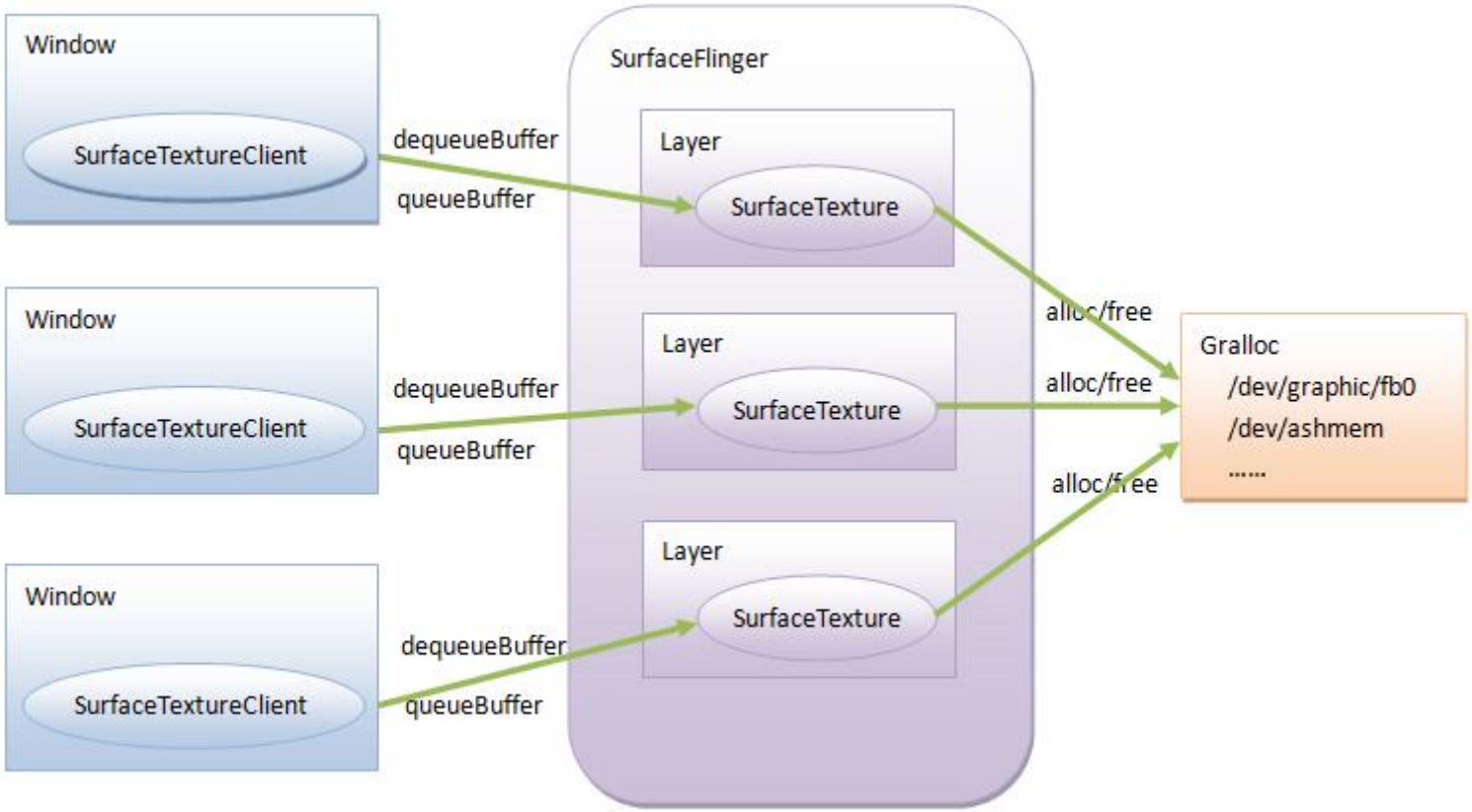
SurfaceFlinger

- 渲染流程



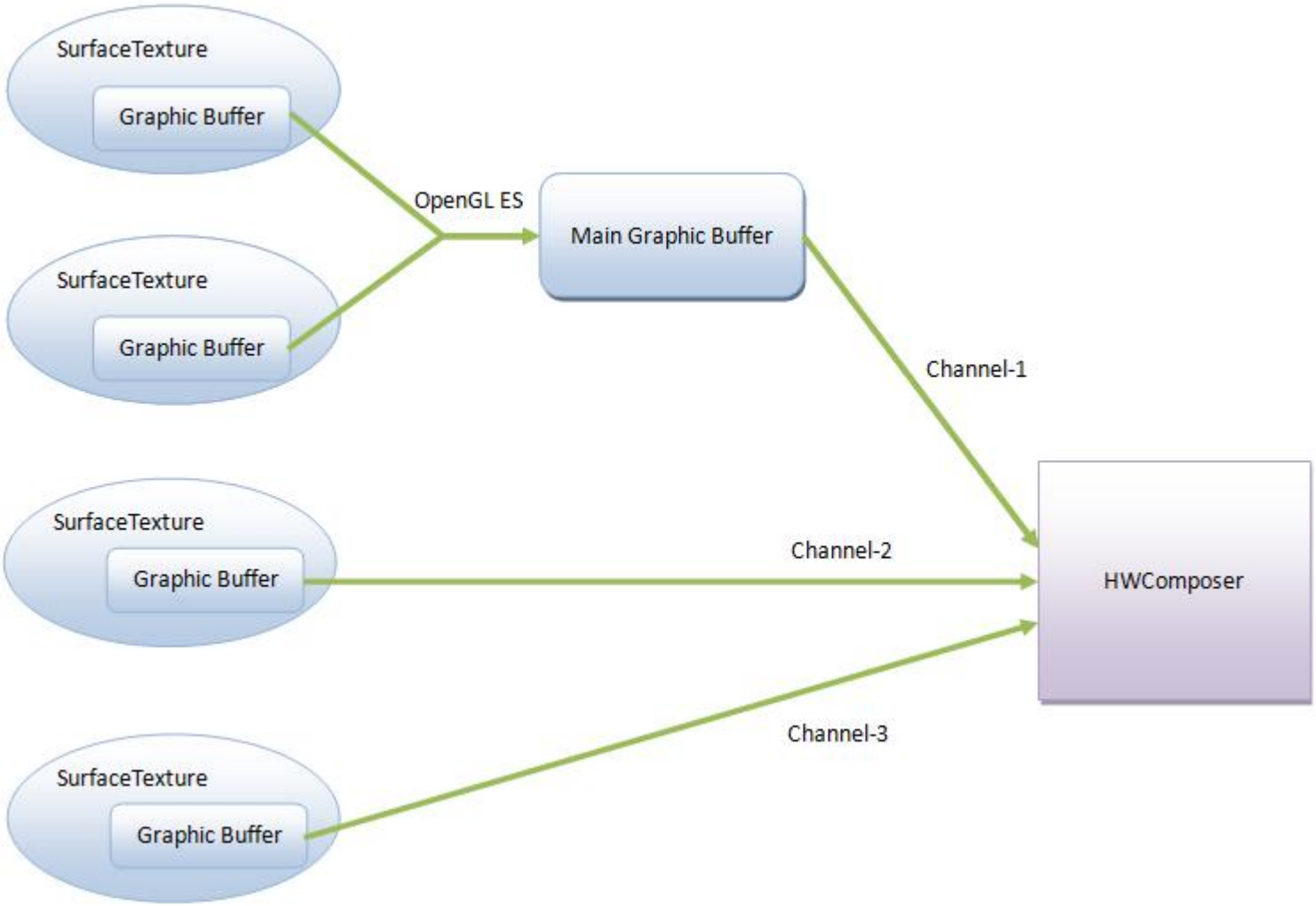
SurfaceFlinger

- 分配图形缓冲区



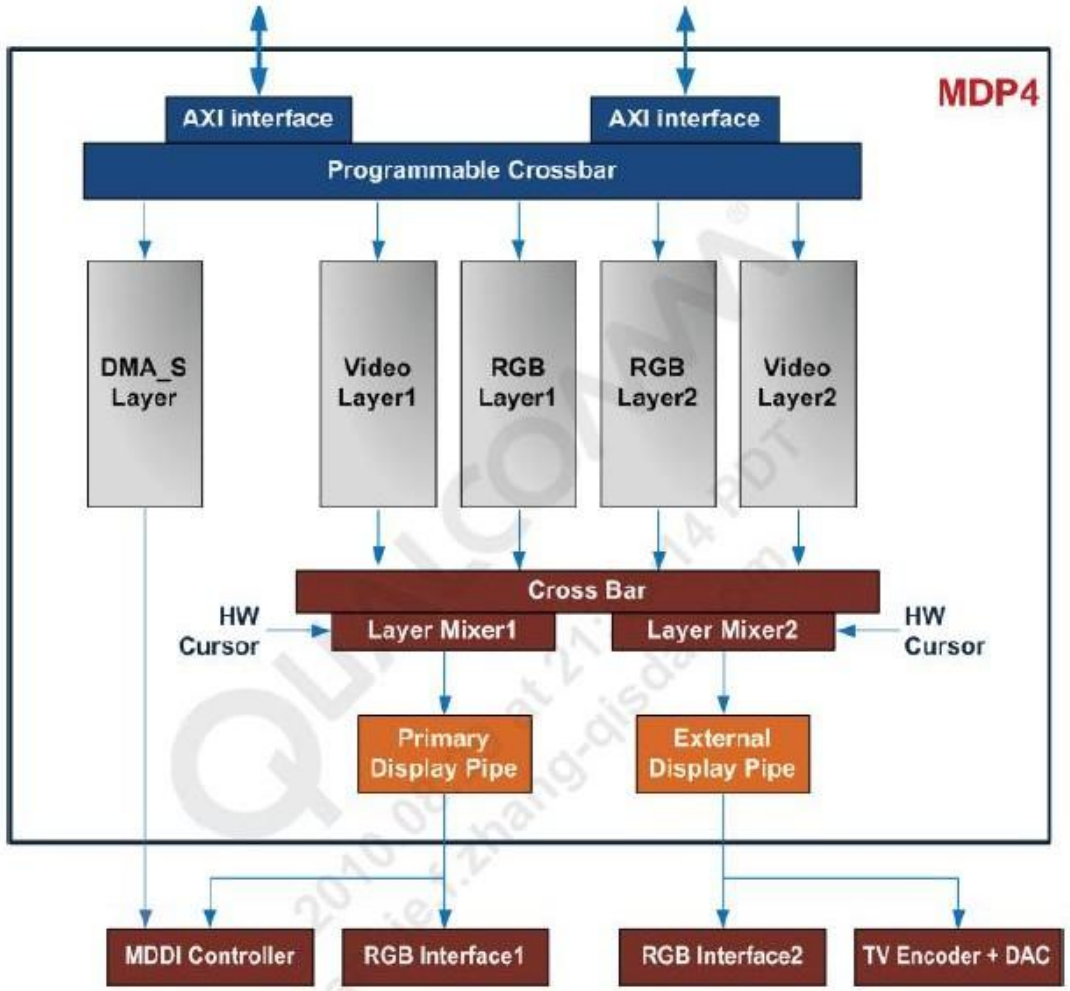
SurfaceFlinger

- 合成图形缓冲区



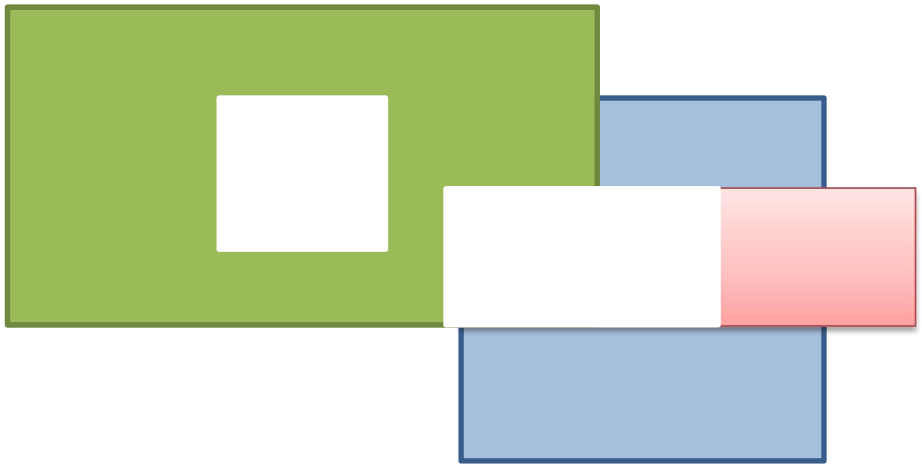
SurfaceFlinger

- HWComposer实例：高通MDP4.0



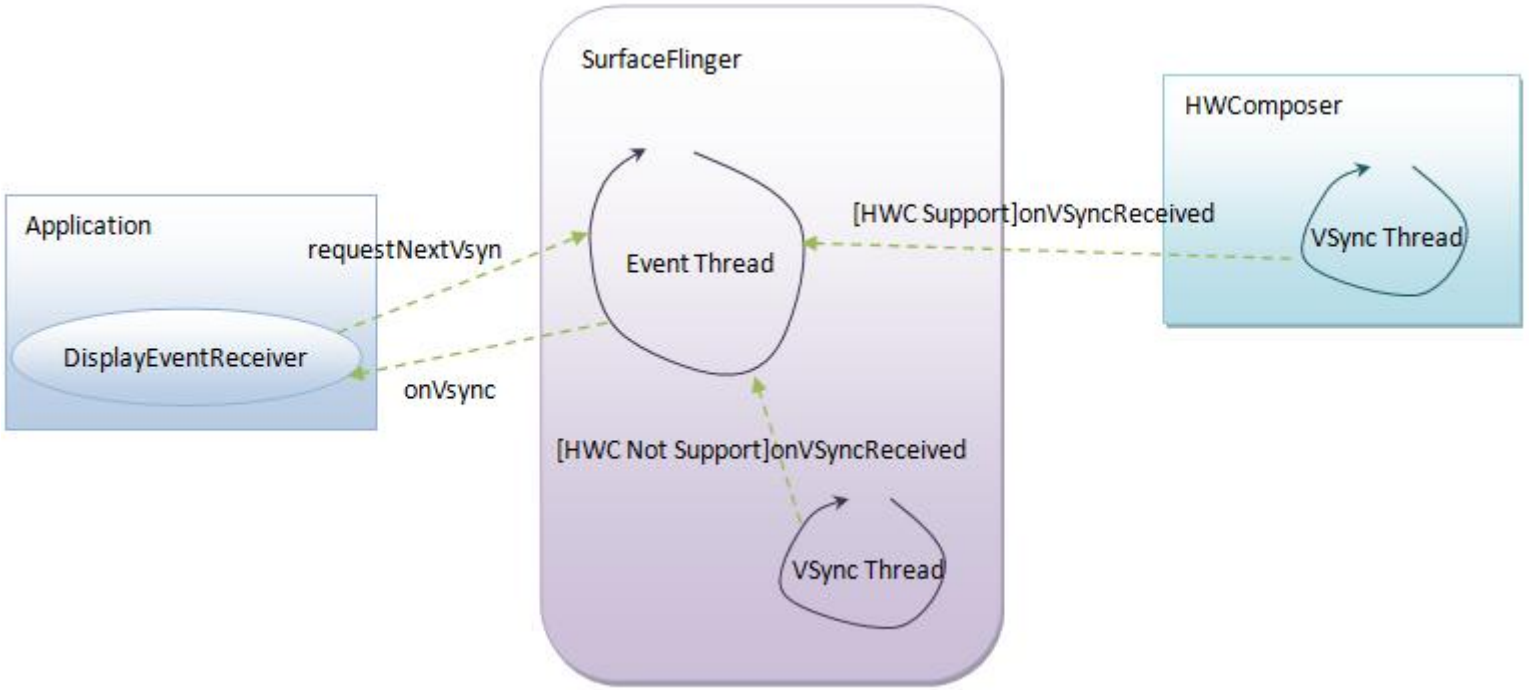
SurfaceFlinger

- 合成图形缓冲区 – 可见性计算



SurfaceFlinger

- 管理VSync事件



Android多屏支持

- 从4.2开始支持多屏幕

Android多屏支持

- 屏幕类型
  - Primary Display
    - 设备自带的屏幕
    - 由SurfaceFlinger管理
  - External Display
    - 通过HDMI连接
    - 由SurfaceFlinger监控和管理
  - Virtual Display
    - 通过Miracast连接(基于Wifi Direct技术)
    - 由DisplayManagerService监控和管理

Android多屏支持

- App通过android.app.Presentation接口在指定的屏幕上创建窗口
  - http://developer.android.com/reference/android/app/Presentation.html

```
DisplayManager displayManager = (DisplayManager) context.getSystemService(Context.DISPLAY_SERVICE);
Display[] presentationDisplays = displayManager.getDisplays(DisplayManager.DISPLAY_CATEGORY_PRESENTATION);
if (presentationDisplays.length > 0) {
    // If there is more than one suitable presentation display, then we could consider
    // giving the user a choice.  For this example, we simply choose the first display
    // which is the one the system recommends as the preferred presentation display.
    Display display = presentationDisplays[0];
    Presentation presentation = new MyPresentation(context, presentationDisplay);
    presentation.show();
}
```

```
private final class MyPresentation extends Presentation {
    ......
}
```

Q&A

Thank You