Android-Window显示层级计算

1，Window前导知识

Window和View的关系：

Window是一个窗口的概念，Android中所有的视图都是通过Window来呈现的，不论是Activity、Dialog还是Toast，视图实际都可以看成是附加在window上，即Window是View的载体。

那什么是window，在Android的window机制中，每个view树都可以看成一个window。为什么不是每个view呢？因为view树中每个view的显示次序是固定的，例如我们的Activity布局，每一个控件的显示都是已经安排好的，对于window机制来说，属于“不可再分割的view”。

View树：

什么是view树？例如你在布局中给Activity设置了一个布局xml，那么最顶层的布局如LinearLayout就是view树的根，他包含的所有view就都是该view树的节点，所以这个view树就对应一个window。在activity中，最底层的布局就是一个View树，它没有父布局了；而对于一个自定义布局的Dialog来说，Dialog的顶层布局就不属于activity的View树，这是2个View树，所以是2个Window。

Window的相关属性：

1，Window的type属性：这个属性就决定了window的显示次序。window是有分类的，不同类别的显示高度范围不同，例如我把1-1000m高度称为低空，1001-2000m高度称为中空，2000以上称为高空。window也是一样按照高度范围进行分类，他也有一个变量Z-Order，决定了window的高度。window一共可分为三类：

应用程序窗口：应用程序窗口一般位于最底层，Z-Order在1-99
子窗口：子窗口一般是显示在应用窗口之上，Z-Order在1000-1999
系统级窗口：系统级窗口一般位于最顶层，不会被其他的window遮住，如Toast，Z-Order在2000-2999。如果要弹出自定义系统级窗口需要动态申请权限。

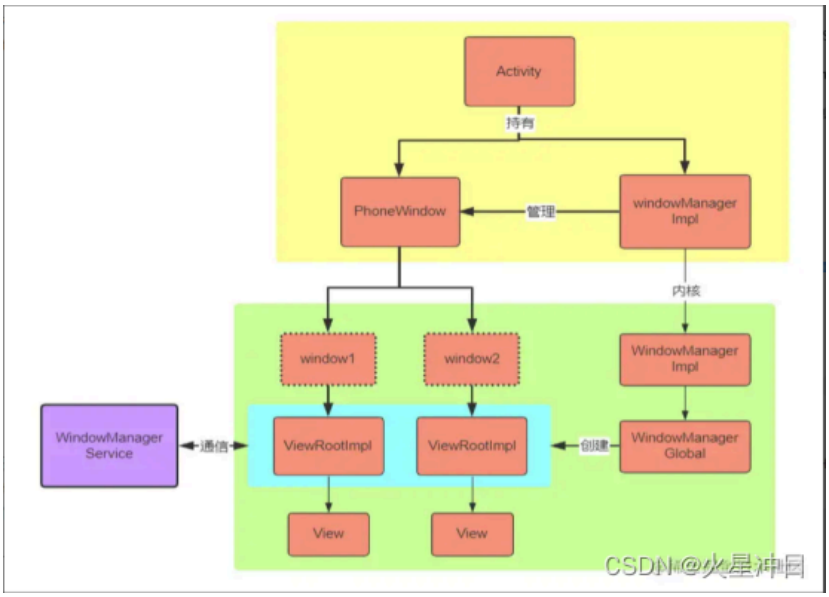
Z-Order越大，window越靠近用户，也就显示越高，高度高的window会覆盖高度低的window。

2，Window的flag属性：flag标志控制window的东西比较多，很多资料的描述是“控制window的显示”，但我觉得不够准确。flag控制的范围包括了：各种情景下的显示逻辑（锁屏，游戏等）还有触控事件的处理逻辑。控制显示确实是他的很大部分功能，但是并不是全部。

3，Window的softInputMode属性：这一部分就是当软件盘弹起来的时候，window的处理逻辑，这在日常中也经常遇到，如：我们在微信聊天的时候，点击输入框，当软键盘弹起来的时候输入框也会被顶上去。如果你不想被顶上去，也可以设置为被软键盘覆盖。

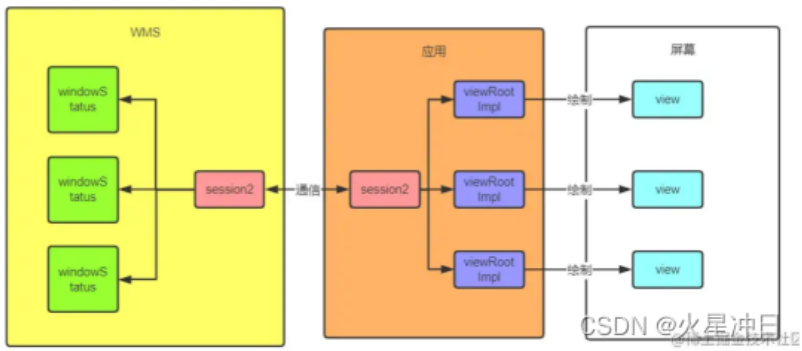
一些API概述：

ViewRootImpl：连接View和WindowManagerService的桥梁，另外还负责View的绘制工作，同时也通过IPC通信（BInder）让WindowManagerService创建Window。



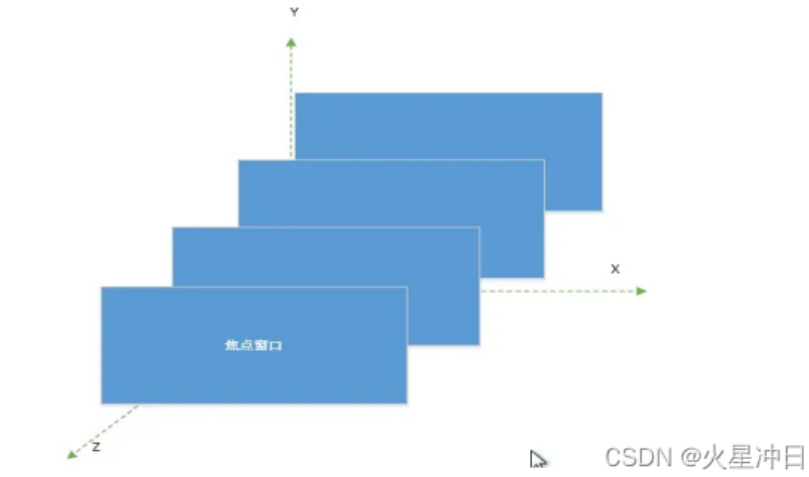
WindowManager：一个接口，继承自ViewManager，负责对Window的管理，其实现类是WindowManagerImpl，实际是WindowManagerGlobal（APP内单例）。在一个APP内可能有多个Activity，于是就有多个View树，多个ViewRootImpl，多个Window。在创建、删除Window时，实际的执行者是WindowManagerService（通过Binder通信）。

PhoneWindow：Window接口的唯一实现类。



Windowstate：windowstate负责管理一个具体的window，由WMS在addWindow（）创建，包含了窗口的所有属性，WMS管理多少window，其内部就有多少windowstate，windowstate内部定义了一些属性，可以管理window的显示层级与次序。

二，窗口层级计算：



窗口主序计算：

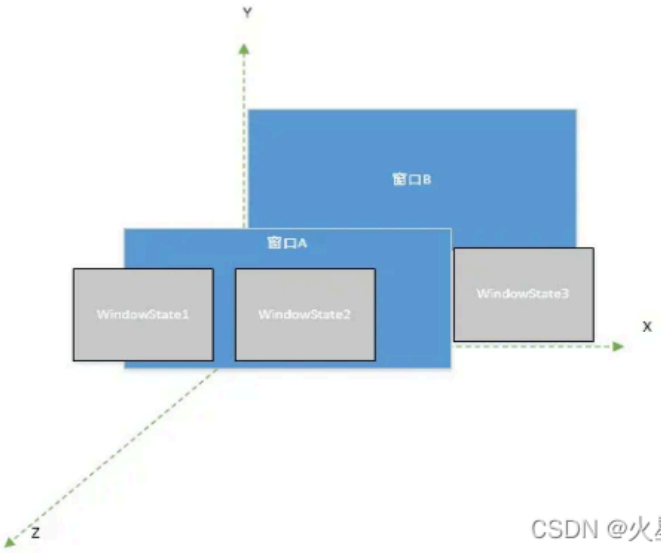
WindowState其中一个属性为mLayer，表示窗口在Z轴的位置，mLayer值越小，窗口越靠后，mLayer值越大，窗口越靠前，最前面的一个窗口就作为焦点窗口，可以接收触摸事件。因为窗口的切换，切换后的Z序(窗口的显示次序称为 Z 序)就可能不同，所以mLayer的值不是固定不变的。mLayer是通过WindowState的另一个成员变量mBaseLayer的值计算得到，mBaseLayer的值是固定不变的，只和窗口类型有关。mBaseLayer(称为主序)是WindowState的构造方法中赋值。

mBaseLayer =窗口类型×10000+1000

窗口类型取值范围是[1,36]，比如应用程序的窗口是 2 ，电话类型的窗口是 3 ， Toast窗口是 8 …，因为系统中同类型的窗口比较多，所以对返回的整数值乘以10000(WindowManagerService.TYPE_LAYER_MULTIPLIER)，在加上1000(WindowManagerService.TYPE_LAYER_OFFSET)，相当把 Z 轴划分成了31个值域，不同类型的窗口的Z轴位置都是处于两个不相交的值域之中，互相不打扰。OK，通过mBaseLayer，我们知道了窗口是如何排序的，一个窗口有可能需要依附在一个父窗口上，作为一个子窗口，所以除了主序的概念外，还有子序。

窗口子序计算：

SubLayer(称为子序)，SubLayer值是用来描述一个窗口是否属于另外一个窗口的子窗口，或者说SubLayer值是用来确定子窗口和父窗口之间的相对位置的。



CSDN @火星冲日

一个Activity中有三个子窗口WindowState1、WindowState 2 、WindowState 3 ， WindowState1WindowState 2 在窗口 A 的前面，WindowState 3 在 A 的后面，这几个兄弟窗口为什么可以这样排序呢，这就是mSubLayer的作用，子序越大，则相对其他兄弟窗口越靠前，反之，越靠后，如果为负数，就处在父窗口的后面，如窗口 A 中的WindowState3，子序是根据窗口类型调用subWindowTypeToLayerLw确定的，subWindowTypeToLayerLw同样是在Window的构造方法中调用的。

子序的范围是[-2,3];

Z序调整：

当WindowState创建完成，并且被添加到WMS维持的数组里面后，就需要调用WindowLayersController的assignLayersLocked(windows)，进行Z序的调整。

一些疑问和解答：

1，Layer值为何乘以1W?

这是因为相同类型的窗口的Z轴位置都是有着相同的值域，而不同类型的窗口的Z轴位置都是处于两个不相交的值域，又由于每一种类型的窗口的数量是不确定的，因此，WindowManagerService服务就需要为每一种类型的窗口都预留一个范围足够大的值域，以便可以满足要求。

2，layer值乘以1W后为何还要再加上1000?

窗口在打开或者关闭的时候，为了不那么突兀，都会设置一个动画，所以Z序号设置1000的偏移量就是为了这个动画层级属性预留的空间。

3，上文中提到Window的type属性用于控制显示层序，这里又有WindowState的layer属性也可以控制显示层级，那么最终起作用的是谁呢?

是Layer，实际上，Layer就是通过Window的type属性计算而来。