图形的绘制，我们就是把要绘制指定的图形区域显示某块特定的屏幕区域，如果屏幕区域保持恒定，图形区域缩小，则在屏幕上显示的效果就是图形放大，反之就是缩小。当然同理也可以保持绘制的图形区域恒定，增大屏幕区域范围，则显示效果是放大。一般缩放都不会采用改变屏幕显示区域，这样让用户感觉迷惑，除了一些分屏之类情况除外。

主要介绍屏幕区域恒定，并且是整个客户区的区域的绘图情况，并且X/Y的缩放比例是一致的(如果不一致则可能会导致图形显示出现扭曲现象，除非对于某些特效，一般都采用X/Y方向的比例一致)。其他情况原理一样。

1. 计算出当前屏幕客户区中心点的坐标。这个点称作为屏幕缩放基准点(Xsc,Ysc)，以及X/Y方向的距离(DXsc,DYsc)。正常情况下只要不是手动调整客户大小或者最大化，客户区的缩放基准点和X/Y方向距离应该是不变的，所以一般情况计算出一次以后，下次就可以忽略这一步，直接从第二步开始。
2. 计算出当前图形显示区域的中心点。这个点称作为图形缩放基准点(Xmap,Ymap)，在初始情况下，该基准点就是图形中心点，X/Y方向的距离应该是图形的宽度和高度(DXmap,DYmap)。
3. 计算出缩放量zoom。如果用鼠标滚轮，则缩放量跟滚轮滚动的角位移成正比(缩放量也可以预定义)。注意：缩放量应该有上限和下限，例如当屏幕显示整个图形时，不应该在继续缩小。放大也是放大到一定级别的时候应该禁止继续放大。
4. 根据缩放量，计算出要显示的图形区域X/Y方向的距离，以图形基准点为中心重新构筑一个图形显示区域。DXmap = DXmap \*zoom,DYmap = DYmap \* zoom。
5. 算出图形区域对应屏幕区域的比例数据。ScaleX = DXmap / DXsc，ScaleY = DYmap / DYsc。注意：ScaleX应该等于ScaleY，否则图形会发生扭曲。
6. 图形裁剪。如果图形数据不在裁剪区内或者不与之相交，则忽略该数据。这一步的难点主要是判断图形是否与裁剪区相交，如直线两点都在区域外，但是直线穿过裁剪区。不过这一步可以先缓一缓，但有这一步绘制图形的效率会更高。
7. 重画图形。新图形的坐标点Xnew = Xreal – Xcenter + DXmap /2，其中Xnew为新图形区域X坐标，Xreal为图形的实际坐标，Xcenter为图形区域的中心点实际坐标。同理Y的坐标为 Ynew = Yreal – Ycenter + DYmap /2。映射到屏幕的坐标为 Xscreen = Xnew \* ScaleX，Yscreen = Ynew \* ScaleY
8. 注意1：第一次开始计算图形X/Y方向的距离时，应该参考屏幕区域的X/Y方向的距离。以某个方向距离为基准，如ScaleX和ScaleY这两个值以最大的一个为基准，这样能够显示整个图形区域。假设ScaleX > ScaleY，则ScaleY = ScaleX，DYmap = ScaleY \* DYsc。
9. 注意2：鼠标滚轮缩放时，Klayout采用以鼠标当前点位置为缩放基准点，我们可以不采用这种情况，直接以屏幕客户区中心点进行缩放。
10. 如果直接在图形画出一个矩形，要放大显示这个矩形里面的内容。这种缩放属于窗口缩放。首先计算出矩形的中心点坐标以及对应的图形实际坐标，接着把这个图形中心点实际坐标作为图形缩放基准点(Xmap,Ymap)。其次计算出矩形的左下角和右上角对应的图形实际坐标，根据这些值计算出图形的宽度和高度(DXmap,DYmap)。最后根据客户区屏幕的X/Y方向的距离(DXsc,DYsc)调整(DXmap,DYmap)。
11. 平移是一种特殊的缩放，只有基准点改变，其他不变。可以直接把图形的基准点加上平移量。不过要考虑如果平移出了图形的边界，则不应该允许继续平移。
12. 重新绘图室对于比例尺以及一些提示文本，不需要进行缩放，这些东西必须在最后绘制。
13. 如果是OpenGL或者DX绘图，则更简单，只需要设置一个裁剪区域就搞定。