为了绘制标尺，我们把标尺作为特殊的图形对象。只针对刻度标尺做描述，并且刻度线与标尺垂直并且只在一侧，为标尺方向逆时针旋转90度。其他的形式的标尺原理一样，就是画法稍微有些区别。

标尺对象至少包含如下数据：

起始点(Xb,Yb)和结束点(Xe,Ye)

沿标尺方向的基本刻度长度HLen

刻度线基本长度VLen

基本刻度长度HLen和刻度线基本长度Vlen都可以采用动态数据，不过为了简单起见，刻度先可以固定长度，等以后熟悉了这些长度数据可以跟着图形缩放而实时变化。

在绘制标尺阶段，随着鼠标的移动，结束点(Xe,Ye)坐标实时变化的。在每一次变化过程中，需要做如下的工作：

1. 计算出标尺的长度RulerLen。
2. 计算出标尺的方向单位向量(DirX, DirY)。DirX = (Xe - Xb) / RulerLen， DirY = (Ye - Yb) / RulerLen。
3. 计算出刻度线的方向单位向量(MarkDirX, MarkDirY)。MarkDirX = - DirY，MarkDirXY= DirX。
4. 根据标尺的长度RulerLen和基本刻度长度HLen计算出标尺中间需要画出多少个刻度。在画刻度线时可以以这个值作为循环上限。
5. 起始点(Xb,Yb)和结束点(Xe,Ye)连线就是标尺线。任何一段刻度线的起始点坐标MarkBeginX = Xb + DirX \* HLen \* n，MarkBeginY = Yb + DirY \* HLen \* n。n表示第几段刻度线，0表示标尺开始点的刻度线起点。第n段刻度线的终点坐标MarkEndX = MarkBeginX + MarkDirX \*VLen，MarkEndY = MarkBeginY + MarkDirY \* Vlen。
6. 如果RulerLen小于或者等于基本刻度长度HLen，则只需要画起始点和终止点的刻度线。如果长度RulerLen正好是基本刻度长度HLen的整数倍，则循环结束后不需要再画标尺终点的刻度线。
7. 如果需要画辅助刻度线。例如每五段或者两段中间的刻度线比五段或者两段边界的刻度线短。画刻度线方法还是一样，只是计算中间刻度线终点坐标时刻度线基本长度VLen采用一个稍短的长度(推荐采用0.7 = 根号2除以2)。
8. 在鼠标移动绘图时，最好采用采用操作方式为当前笔的与背景异或，这样在鼠标移动时，擦除以前的标尺时只需要把以前的标尺重画一边，而不需要刷新图形区域（当然也可以通过刷新图形的区域来做到，只是效率低了一点点）。如果采用异或方式，也许刻度线的起始点可能需要沿着刻度线方向移动一个像素(或者标尺线的宽度占用的像素)。
9. 标尺绘制完毕后，如果标尺里面的坐标保存的不是图形坐标系统的坐标，则需要从屏幕坐标转换为图形的坐标，以便适应以后的缩放。如果基本刻度长度HLen和刻度线基本长度Vlen采用动态变化的，在每次缩放时必须计算出新的值然后再根据这些值来画出标尺。一旦标尺作为一个图形对象，就跟其他的如直线等图形对象一样的对待了。

Klayout里面的标尺比较复杂，有各种各样的样式，完全实现他们则需要的工作量大(直接抄袭代码倒是可以节省大量工作量☺)，可以先按照上面我说的绘制基本的标尺，熟悉以后再实现里面的各种各样的式样。Klayout画标尺的源代码在antService.cc文件中的draw\_ruler()函数里面实现。

另：

1、对于邓恒华提出的显示屏的横竖切换，这个只需要根据客户区的X/Y方向距离重新计算出图形的宽度和高度(DXmap,DYmap)，然后再刷新重新绘制一次，就能完全对应。

2、关于图形显示的全局转换设置，则在绘图之前，必须把整个图形数据沿着图形基准点旋转指定的角度(也就是每个点转换为屏幕坐标时，都要事先旋转指定的角度再转换为屏幕坐标)。