**初级软件设计实作**

**桌面放大镜 技术文档**

**目录**

1. **题目……………………………………………3**
2. **开发过程………………………………………4**
3. **设计方案………………………………………6**
4. **技术讨论………………………………………9**
5. **运行截图……………………………………10**

**（1）题目:** **一个桌面透视工具**

功能需求：

实现一个电脑桌面的放大镜。具体功能如下：

（1） 在软件启动时，显示一个“取景框”（小）和“展示框”（大），展示框的内容与取景框中的内容是一致的，但展示框内容是放大。 取景框是一个虚线框。

（2） 用户可以用鼠标来移动取景框。

（3） 当取景框在电脑桌面上移动时，软件马上把该放大镜取景框所在位置的内容给放大显示到展示框中。

相关技术：

（1） 图像展现

（2） 截屏功能的实现

（3） 鼠标消息处理

**（2）开发过程**

首先，本软件基于QT6.1.0框架开发，实现语言为C++语言，编译器为MinGW 8.1.0 x64，编译为64位动态链接可执行程序，C++语言标准为C++20。

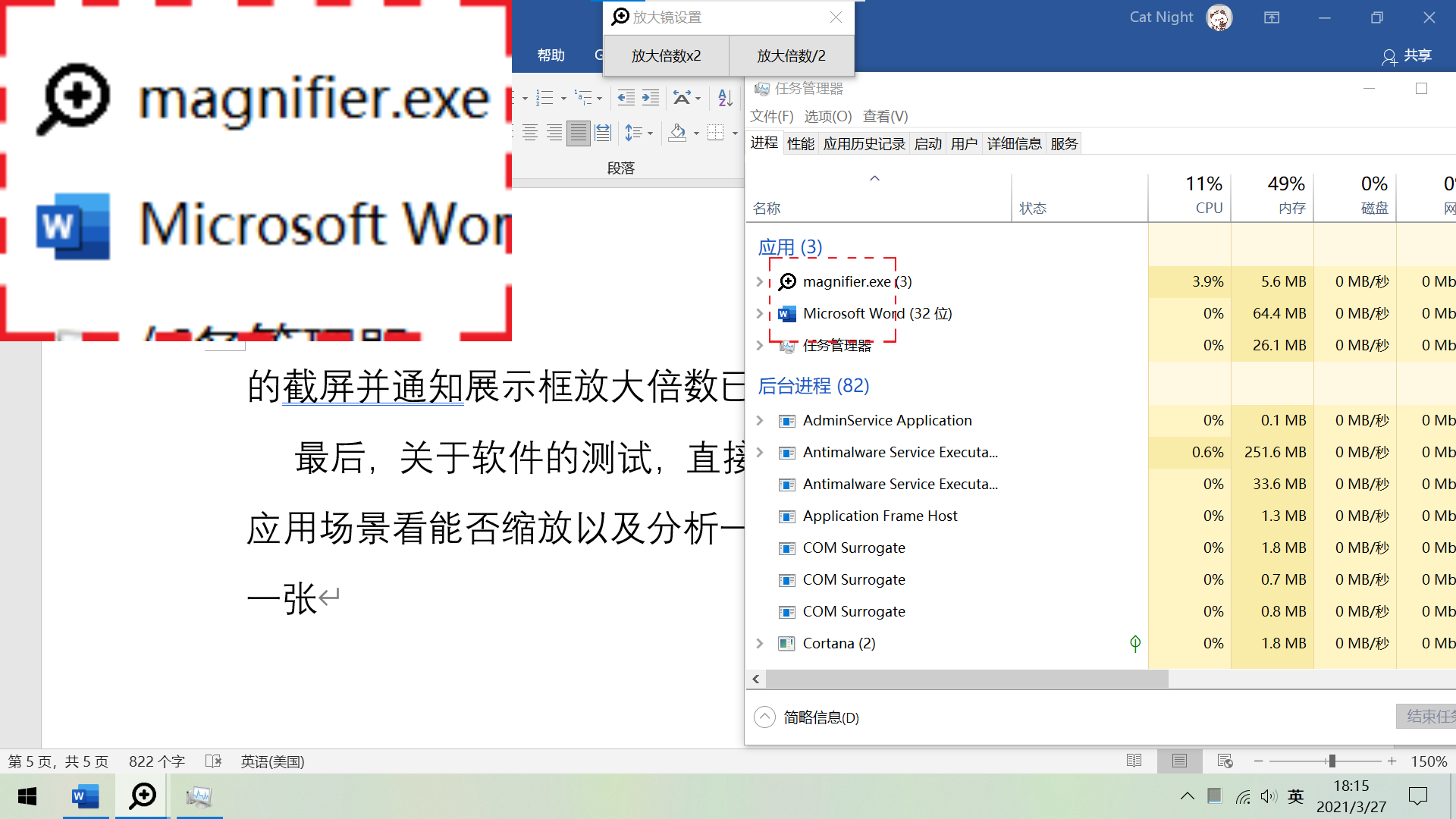
由于是桌面放大镜，所以考虑一共实现3个窗口，一个窗口用来作为取景框使用，一个窗口用来作为展示框使用，最后一个窗口用来作为设置窗口，用于设置放大镜的倍数。

在设计取景框时，采用无边框透明窗口背景，在上面放一个标签组件，标签上显示背景透明的虚线取景框图片，并对该标签做鼠标事件处理，当鼠标点击取景框并拖动时，将取景框实时的跟随用户的移动而移动到对应的位置。此外，取景框窗口还应将本取景框所在位置的内容实时的截屏并保存到一个对象中供展示框窗口读取展示。

在设计展示框时，采用无边框窗口样式，上面不放置任何的控件，当取景框已经将桌面截屏保存下来并通知展示框时，展示框要及时地读取取景框保存下来的图片并用画笔画到展示框的窗口上去，同时要触发窗口重绘事件刷新窗口以让用户能及时感知到变化。

在设置窗口上一共设计了一个功能，即调整放大镜的放大倍数，上面提供两个按钮用于调整倍数，一个用来将放大镜的放大倍数X2，一个用来将放大镜的放大倍数/2，在用户按下按钮时应及时通知取景框改变放大倍数，取景框也应及时的截屏并通知展示框放大倍数已经变化了。

最后，关于软件的测试，直接打开应用，然后尝试在几个应用场景看能否缩放以及分析一下缩放的性能即可。这里放一张测试成功的图片，内含正常使用时CPU的占用率。



**（3）设计方案**

如上图，整个软件一共有三个窗口。

展示框只是一个纯净的窗口，没有任何的控件，相当于提供一个画布来画画。该窗口开启了无边框窗口模式，看上去比较美观，并且在设置为无边框模式后该窗口占用的位置更少。这个界面的实现类中有一个QPixmap类的成员变量叫mScreenShot，里面存放着截屏的图片。在这个窗口的paintevent重绘事件中，将该截屏用QPainter画到窗口上，铺满窗口，并开启pixmap抗锯齿算法，使得显示的效果比较完美。该实现类还提供了一个成员函数用于设置mScreenShot，该函数主要提供给取景框类在截完图后设置展示框的mScreenShot使用。至于这个窗口的拖动算法，具体的实现原理与取景框类似，这部分算法将在取景框部分详细介绍。

取景框是一个窗口加上一个QLabel控件，QLabel控件用于显示虚线框，告诉用户这个是取景框。该窗口也开启了背景透明化，也开启了无边框窗口模式。为了实现用户拖动取景框的效果，该窗口重写了mouseMoveEvent，mousePressEvent,mouseReleaseEvent三大方法，在用户按下时先判断是否按下的是鼠标左键，如果是的话就记录按下时的窗口的位置和鼠标的位置并将“在窗口内按下“这个布尔变量的值设置为真，当鼠标移动时先判断”在窗口内按下”这个布尔变量的值是否为真并且鼠标左键是否被按着，如果是真的话就根据之前记录的信息和现在鼠标所在的位置将窗口移动到相应的位置，具体的算法是:窗口的新位置=窗口在鼠标左键按下时的位置+现在的鼠标位置-在按下鼠标左键时鼠标所在的位置。在鼠标左键松开时将” 在窗口内按下”这一布尔变量设置为假。判断是否左键按下的具体作用是防止鼠标只有右键按下或者没有按下任何键只是单纯的移动时窗口也跟着移动。而设置” 在窗口内按下”这一布尔变量的意义是防止鼠标在窗口外按下鼠标左键并移动到窗口内时触发窗口移动而导致产生诡异的效果。该类还提供了一个成员函数用于改变放大的倍数，其实也就是改变取景框的大小，只要取景框宽高改变了而展示框的宽高没有动，那么就相当于增加或者减少了放大倍数。至于截屏功能，是使用的Qt的API接口QtGuiApplication::primaryScreen()->grabWindow()方法。传入所需的截图的开始位置，结束位置即可获得格式为QPixmap的截图，然后将其赋给展示框的mScreenShot成员变量即可。探究Qt源码，这个方法底层也是调用的WindowsAPI，先CreateCompatibleDC创建一个兼容的DC，接着调用CreateCompatibleBitmap创建一个兼容的Bitmap,然后用SelectObject将Bitmap对象放入DC中准备开始拷贝位图，接着就是调用BitBlt拷贝位图数据完成了屏幕截取的操作，最后再用SelectObject还原Bitmap。由于已经掌握了原理，所以这里直接调用Qt的API接口。对于截屏这一操作，刚开始尝试使用QTimer每隔一段事件截一次屏，但是后来发现展示框的刷新率非常低，估计是QTimer的timeout信号与窗口本身的消息一起发送到窗口的消息队列导致消息队列消息过多发生拥堵所致，所以考虑开启另外一个子线程去处理这一事件，让截屏消息占满子线程的消息队列，并且在类中用mExit原子布尔变量类控制子线程的生命周期。

最后是设置窗口，该窗口设置了两个按钮，这两个按钮是用来设置放大倍数的，一个用于将放大倍数X2，一个用于将放大倍数/2，具体的实现是发送信号给取景框，通知其改变大小，让其宽高X2或者/2。

**（4）技术讨论**

这里讨论一下软件的不足之处：

（1）、不能直接使用像调整窗口一样去调整取景框的大小从而调整放大倍数。

（2）、展示框大小不能调整

（3）、性能不太好，cpu占用率有点高，但是无法优化。如果兼顾性能，实现高性能，只在取景框移动时刷新展示框，那么性能就会得到非常好的提升，但是带来的问题是展示框内的内容不是实时的，需要移动取景框才能同步过去，不移动就保持一个画面不动了，在取景框静止或者鼠标拖动过程中不动时无论取景框下发生了什么变化都不会实时显示上去。如果兼顾数据一致性，实现高可用，那么就需要牺牲一些cpu性能来换取最佳的软件表现，画面更新速度十分快，取景框下发生了什么变化都会实时显示到展示框上去。本软件采用的是高可用解决方案。

**（5）运行截图**

