C 技巧之一（MFC） PDF转换可能丢失图片或格式，建议阅读原文
https://www.100test.com/kao_ti2020/465/2021_2022_C___E6_8A_80_E5_B7_A7_E4_c67_465571.htm 在冗长和粒状的操作中更新窗口时的一个典型难题就是窗口闪烁，改变控件内容导致反复重画控件可见区域的部分或全部。当短时间内这样的更新大量发生时，这就成问题了，而且这种反复重画是以一种无任何吸引力的可视干扰形式出现的。解决这一难题的两种常用方法是使用API函数LockWindowUpdate和WM-SETREDRAW消息。 LockWindowUpdate通过换出此窗口的正常设备场景(device context)来实现，以可见区域为空的窗口来替代此窗口。当LockWindowUpdate被以NULL呼叫时(解除窗口锁定)，原来的设备场景被替换，系统使在它之内的与临时性的窗口大小和位置相同的一个区域失效，这样窗口将收到一要求以重画修改的区域。因此所有的窗口绘制在单一操作中有效完成，而且视觉效果更加无缝。使用LockWindowUpdate的缺点是它一次仅能用于一个窗口，因此它不能被用来同时锁定一批相关的控件集。呼叫此函数的第一个窗口被锁定，而解除锁定之前的所有其它呼叫均失败。(这一点很容易地显示地被显示出来，通过测试程序执行文件wndscope.cpp的第93行上的运行按钮的未注释的人工锁定，wndscope.cpp包括在这个月的代码文档中)。有趣的是，使用LockWindowUpdate的应用程序的两个版本同时运行，看起来，任何一个呼叫此函数的过程“拥有”锁定功能，移去以前成功呼叫的所有权，这还原成为串的闪烁的解除锁定的行为插入。 WM-SETREDRAW是被各种标准和自定义控件实现

的消息，这些控件包括列表框(listbox)，组合框 (combobox)，列表视图控件(list view)，按钮(button)，和tab控件(tab control)。它通过清除和设置窗口重画标记(flag)工作。唯一的微小缺点是应用程序必须在发送恢复活动的消息之后使窗口矩形失效，因此全部可视窗口矩形将被重画。WM-SETREDRAW的使用一般取代LockWindowUpdate的使用。当然，对于不支持这一消息的窗口，LockWindowUpdate仍然是可选用的工具。这里给出WinSTL库(http://winstl.org/)中的两个类window_0update_scope 和 window_redraw_scope提供锁定locking的两种形式的自动化作用域。(有所删节的实现部分内容见列表1和列表2。 完整的执行部分在档案文件中提供，而在WinSTL站点也可联机获取)。锁定在构造器constructor中设定，然后在析构器destructor中重置：window_0update_scope呼叫其constructor中的LockWindowUpdate，如果成功在其析构器destructor中呼叫LockWindowUpdate(NULL)；window_redraw_scope发送其constructor中的WM-SETREDRAW (传递False)并且发送WM-SETREDRAW (传递True)。它们二者都取得窗口的句柄以在它们的constructors中的锁定；window_redraw_scope取得第二个参数(缺省值为True)以确定当未锁定时窗口是否失效(通过呼叫InvalidateRect)。 这两种技术的用法在附随的的应用程序(图表1 )中作了演示，把许多项目发送到一个列表框listbox，使用或没有使用两种刚刚描述的锁定技术。为了显示技术，简单地执行程序并且选择“ Run ”，演示闪烁。然后依次选择“ Use LockWindowUpdate ”和“ Use WM-SETREDRAW ”并运行，演示锁定。 100Test 下载频道开