每个窗口保存一份

HDC m\_hMemDC;

HBITMAP m\_hMemBitmap; // 大小为窗口大小

对于正常窗口，可用于实现双缓冲绘制，对于分层窗口，可用于传递给UpdateLayeredWindow

一. Gdi 渲染模式

1. 刷新窗口

|  |
| --- |
| void OnPaint(WPARAM wParam)  {  HDC hDC = NULL;  if (NULL == wParam)  hDC = ::BeginPaint(this->m\_hWnd ,&ps);  IRenderDC\* pRenderTarget = CreateRenderTarget(Graphics\_type\_gid);  if (pRenderTarget->BeginDraw(m\_hMemDC, (RECT\*)NULL))  {  this->DrawObject(pRenderTarget);  pRenderTarget->EndDraw();  }  SAFE\_RELEASE(pRenderTarget);  this->Commit2Window(hDC, NULL);  if (NULL == wParam)  EndPaint(m\_hWnd,&ps);  } |
| void GdiRenderTarget::BeginDraw(HDC hDC, RECT\* prc)  {  m\_hDC = hDC;  if (NULL != prc)  {  SelectClip(m\_hDC, prc);  }  }  void GdiRenderTarget::EndDraw()  {  SelectClip(m\_hDC,NULL);  m\_hDC = NULL;  } |
| void WindowBase::Commit2Window(HDC hDC, RECT\* prc)  {  ::Bitble(hDC,0,0,w,h,m\_hDC,0,0,SRCCOPY);  } |

CreateRenderTarget 只需要负责new一个对象即可

BeginDraw(hDC, NULL) 负责保存这个m\_hDC

EndDraw() 空实现

Release() 释放 delete

2. 刷新控件

|  |
| --- |
| void RedrawObject(Object\* pObj)  {  CRect rcWindowObj;  pObj->GetWindowRect(&rcWindowObj);  IRenderDC\* pRenderTarget = CreateRenderTarget(Graphics\_type\_gid);  if (pRenderTarget->BeginDraw(m\_hMemDC, (RECT\*) rcWindowObj))  {  pObj->DrawObjectTransparentBkgnd(pRenderTarget, pObj->istransparent());  pObj->DrawObject(pRenderTarget);  pRenderTarget->EndDraw();  }  SAFE\_RELEASE(pRenderTarget);  this->Commit2Window(hDC, &rcWindowObj);  } |

3. 刷新列表项

二. Gdiplus 渲染模式

|  |
| --- |
| Graphics\* m\_pGraphics;  void GdiRenderTarget::BeginDraw(HDC hDC, RECT\* prc)  {  if (NULL != prc)  {  SelectClip(m\_hDC, prc);  }  m\_pGraphics = FromHDC(hDC);  m\_hDC = hDC;  }  void GdiRenderTarget::EndDraw()  {  SelectClip(m\_hDC,NULL);  m\_hDC = NULL;  delete m\_pGraphics;  } |

1. 分层窗口

|  |
| --- |
| 刷新时发送PAINT消息统一流程::SendMessage(m\_hWnd, WM\_PAINT, m\_hMemDC, 0);  void OnPaint(WPARAM wParam)  {  HDC hDC = NULL;  if (NULL == wParam)  hDC = ::BeginPaint(this->m\_hWnd ,&ps);    IRenderDC\* pRenderTarget = CreateRenderTarget(Graphics\_type\_gid+);  if (pRenderTarget->BeginDraw(m\_hMemDC, (RECT\*)NULL))  {  this->DrawObject(pRenderTarget);  pRenderTarget->EndDraw();  }  SAFE\_RELEASE(pRenderTarget);  this->Commit2Window(hDC);  if (NULL == wParam)  EndPaint(m\_hWnd,&ps);  } |
| void CustomWindow::Commit2Window()  {  if (is layered())  {  xxxx  UpdateLayeredWindow(...,m\_hMemDC...);  }  else  {  WindowBase::Commit2Window();  }  } |
| void CustomWin |

2. 普通窗口

三. Direct2D 渲染模式