学 号	0121210680206

武漢理工大學

课 程 设 计

课程名称 可视化编程（VC）

学　　院 计算机科学与技术学院

专业班级 软件工程 ZY1201 班

姓　　名 廖　星

指导教师 岑　丽

2013 — 2014　学年　　第 二 学期

目　录

基于 MFC 的 WEB 浏览器实现

摘　要：随着互联网的高速发展，Web 浏览器作为互联网的入口，其重要性不言而喻。可以说，浏览器的性能影响着互联网的发展。如今已有多款性能优异的 Web 浏览器软件，如 Internet Explorer、Chrome、Firefox、Opera 和 Safari 等。本次课程设计将采用 MFC 框架，借鉴上述浏览器的优点，开发一款美观且实用的 Web 浏览器。

关键词：MFC　　WEB 浏览器　　VC

1　设计题目与要求

设计题目：基于 MFC 的 WEB 浏览器实现

设计要求：①能实现浏览器外观界面的设计

②能实现网页的浏览、后退、前进、刷新等基本功能

③实现其它附加功能（不在要求范围之内）

④界面良好，功能完善

2　系统设计

2.1　总体设计

本次课程设计所实现的 Web 浏览器首先要实现设计要求中的功能，要有友好的界面，能正常的浏览网页，能实现后退、前进、刷新等基本功能。

此外，在要求的功能之上，对 Web 浏览器的功能进行了扩充，能实现网页保存、打印网页、在网页上查找、选中全部内容、标签式多窗口浏览、关闭浏览器提示、页面缩放、网页加载进度显示、浏览器状态以及界面样式更换等功能。

2.2　详细设计

本 Web 浏览器具有较多的功能，因而采用了模块化的设计思想，将每个功能做成了相应的模块，便于开发。另外由于 Web 浏览器的自身性质，本次开发采用了 MFC 的多文档（MDI）模式，使用 App—MainFrame—ChildFrame—View 的模式开发。

功能模块的关系如下：

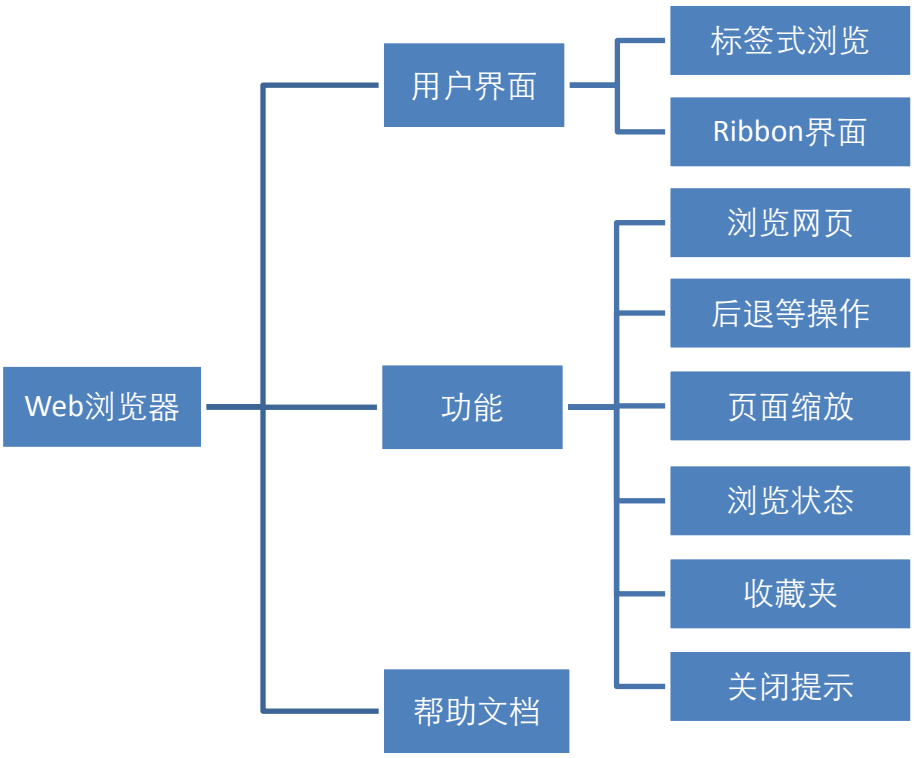

图 1　Web 浏览器功能模块

2.2.1　用户界面设计

本 Web 浏览器的界面采用类似 Microsoft Office 2007 的 Ribbon 界面，摒弃了传统的菜单栏与工具栏组合的界面，使用户界面更加友好。其中，Ribbon 界面是由微软公司设计开发，并被加入了 MFC 类库，因而在 MFC 框架下使用 Ribbon 界面较为容易，并且符合此次课程设计的要求。

图 2　本 Web 浏览器采用的 Ribbon 界面

考虑到标签式的 Web 浏览器是如今的主流，本 Web 浏览器也采用了这种设计，将每个浏览窗口放入标签中，并在标签上添加了关闭按钮，方便用户操作。另外，标签也支持拖动排序，可以拖动标签来修改页面的先后顺序。另外，Windows 7 以及更高版本操作系统推出了任务栏显示缩略图功能，本程序用到了 MFC 中的这项功能，限定了现实范围，使任务栏只显示网页的缩略图。并且对多标签页面做了适配，使一个主窗口在任务栏上能同时显示多个标签页的内容。如图 3 所示：

图 3　任务栏的预览窗口

此外，为了丰富界面元素，本 Web 浏览器使用了 Basicset 图标集中的一些图标。另外支持更换界面样式，有 5 套界面皮肤可供选择。界面皮肤更换的代码如下：

```
void CMainFrame::OnApplicationLook(UINT id)
{
   CWaitCursor wait;
   theApp.m_nAppLook = id;
   switch (theApp.m_nAppLook)
   {
   case ID_VIEW_APPLOOK_WIN_2000:
   CMFCVisualManager::SetDefaultManager(RUNTIME_CLASS(CMFCVisual
Manager));
      m_wndRibbonBar.SetWindows7Look(FALSE);
      break;
   case ID_VIEW_APPLOOK_OFF_XP:
```

```
   CMFCVisualManager::SetDefaultManager(RUNTIME_CLASS(CMFCVisualMan
agerOfficeXP));
         m_wndRibbonBar.SetWindows7Look(FALSE);
         break;
      case ID_VIEW_APPLOOK_WIN_XP:
         CMFCVisualManagerWindows::m_b3DTabsXPTheme = TRUE;
        CMFCVisualManager::SetDefaultManager(RUNTIME_CLASS(CMFC
VisualManagerWindows));
         m_wndRibbonBar.SetWindows7Look(FALSE);
         break;
      case ID_VIEW_APPLOOK_OFF_2003:
        CMFCVisualManager::SetDefaultManager(RUNTIME_CLASS(
CMFCVisualManagerOffice2003));
         CDockingManager::SetDockingMode(DT_SMART);
         m_wndRibbonBar.SetWindows7Look(FALSE);
         break;
      case ID_VIEW_APPLOOK_VS_2005:
        CMFCVisualManager::SetDefaultManager(RUNTIME_CLASS(
CMFCVisualManagerVS2005));
         CDockingManager::SetDockingMode(DT_SMART);
         m_wndRibbonBar.SetWindows7Look(FALSE);
         break;
      case ID_VIEW_APPLOOK_VS_2008:
         CMFCVisualManager::SetDefaultManager(RUNTIME_CLASS(
CMFCVisualManagerVS2008));
         CDockingManager::SetDockingMode(DT_SMART);
         m_wndRibbonBar.SetWindows7Look(FALSE);
         break;
      case ID_VIEW_APPLOOK_WINDOWS_7:
```

```
            CMFCVisualManager::SetDefaultManager(RUNTIME_CLASS(
CMFCVisualManagerWindows7));
            CDockingManager::SetDockingMode(DT_SMART);
            m_wndRibbonBar.SetWindows7Look(TRUE);
            break;
        default:
            CMFCVisualManager::SetDefaultManager(RUNTIME_CLASS(CMFC
VisualManagerOffice2007));
            CDockingManager::SetDockingMode(DT_SMART);
            m_wndRibbonBar.SetWindows7Look(FALSE);
        }
        RedrawWindow(NULL, NULL, RDW_ALLCHILDREN | RDW_INVALIDATE |
RDW_UPDATENOW | RDW_FRAME | RDW_ERASE);
        theApp.WriteInt(_T("ApplicationLook"), theApp.m_nAppLook);
    }
```

2.2.2 多标签模块设计

本程序采用的是 MFC 的多文档模式，并对多文档进行了管理，统一到界面的标签上。新建功能是用来创建新的子窗口，并向父窗口注册，显示在标签上。新建窗口分两种情况，一种是用户触发 ID_FILE_NEW 产生的，无论是从 Ribbon 功能区上点击，还是下拉菜单上点击。另外一种是浏览器触发，当浏览器需要弹出新窗口时新建子窗口，这个问题后面会有解决方案的阐述。

使用 CHtmlView 访问网页的时候，经常会出现脚本错误的提示框，本程序采用了对其设置 SetSilent(TRUE)来屏蔽脚本错误提示。

2.2.3 浏览模块设计

浏览模块作为 Web 浏览器的核心部分，设计使需要考虑很多问题。首先是地址栏。地址栏可以接受用户输入网址，并将信息传给 View 模块，使其加载网页。另外，地址栏还需要能显示当前页面的网址，并且切换标签页的时候，地址栏的网址会随之改变。为了实现以上功能，在 MainFrm.cpp 中 OnAddr 用来接受用户数据并处理，ChangeAddr 用来修改地址栏内容。

OnAddr 首先获取地址栏的指针，然后获取用户输入的内容，如果内容非空，再判断输入的内容是否与当前网页地址相同，防止重复加载。不相同的话，再获取当前活动的子窗口，并将网址传递给子窗口的 View。代码如下：

```
void CMainFrame::OnAddr()
{
    CMFCRibbonEdit *pEdit=DYNAMIC_DOWNCAST(CMFCRibbonEdit,
m_wndRibbonBar.FindByID(ID_ADDR));  //获取地址输入框
    CString strUrl=pEdit->GetEditText();  //获取地址框文本
    if(strUrl=="")  //数据合法性判断
       MessageBox(L"请输入要访问的网址！",L"提示",MB_ICONWARNING);
    else
    {
      CChildFrame *pChildFrame=(CChildFrame*)GetActiveFrame();
CHtmlView *pBrowser=(CHtmlView*)pChildFrame->GetActiveView();
       if(pBrowser->GetLocationURL()!=strUrl)
          pBrowser->Navigate(strUrl);
    }
}
```

当活动窗口的网页加载完成之后，会执行 OnDocumentComplete 函数，并调用 ChangeAddr 函数，将改变传递给主框架。同时，通过 IHTMLDocument2 接口获取网页标题，并显示在标签上。因为实践测试，GetLocationName 方法获取的网页标题并不准确，因而不采用此方法。

OnDocumentComplete 的代码如下：

```
void CMyView::OnDocumentComplete(LPCTSTR lpszURL)
{  //浏览器加载完成
    CComPtr<IHTMLDocument2> pDoc = (IHTMLDocument2*)this->
GetHtmlDocument();
    BSTR *bstrTitle;
    pDoc->get_title(bstrTitle);
    CString title;
```

```
        title.Empty();
        title=*bstrTitle;
        title=title.Left(20);  //只截取前 20 个字符，防止标题过长
        GetMainFrame();  //这个函数用来获取主框架和当前子框架的指针
        pChildFrame->SetWindowText(title);
        pMainFrame->ChangeAddr(GetLocationURL());  //调用函数修改
        pMainFrame->UpdateFrameTitleForDocument(title);
        CHtmlView::OnDocumentComplete(lpszURL);
    }
```

ChangeAddr 的代码如下：

```
    void CMainFrame::ChangeAddr(CString str)
    {  //修改地址输入框文本
        CMFCRibbonEdit *pEdit=DYNAMIC_DOWNCAST(CMFCRibbonEdit,
m_wndRibbonBar.FindByID(ID_ADDR));
        pEdit->SetEditText(str);
    }
```

此外，CHtmlView 类有个需要注意的地方，当有新窗口打开时，它会默认调用 IE 打开新窗口。因而需要对其 WM_NEWURL 消息进行处理，使其在新标签页打开。代码如下：

```
    ON_MESSAGE(WM_NEWURL, &CMainFrame::OnNewUrl)  //消息映射
    afx_msg LRESULT CMainFrame::OnNewUrl(WPARAM wParam, LPARAM lParam)
    {  //CHtmlView 类打开了新窗口
        LPDISPATCH* ppDispatch=(LPDISPATCH*)wParam;
        SendMessage(WM_COMMAND, ID_FILE_NEW, 0);
        CChildFrame* pChildFrame = (CChildFrame*)GetActiveFrame();
        *ppDispatch=((CHtmlView*)pChildFrame->GetActiveView())->GetA
pplication();
        return 0;
    }
```

同时处理 CHtmlView 的 OnNewWindow2 事件。代码如下：

```
void CMyView::OnNewWindow2(LPDISPATCH* ppDisp, BOOL* Cancel)
{   //浏览器弹出新窗口（自定义消息）
    ::SendMessage(AfxGetMainWnd()->m_hWnd,WM_NEWURL,(WPARAM)ppDi
sp,NULL);
    *Cancel=TRUE;
    CHtmlView::OnNewWindow2(ppDisp, Cancel);
}
```

2.2.4 操作按钮模块设计

操作按钮即指后退、前进、刷新、停止等功能按钮，其中后退、前进需要特殊处理。因为需要判断当前页面是否可以后退、可以前进，并且对按钮发出可用或禁用消息。并且切换标签页时同步后退、前进按钮的状态。

后退、前进按钮的状态可利用 OnCommandStateChange 事件判断。代码如下：

```
void CMyView::OnCommandStateChange(long nCommand, BOOL bEnable)
{   //设置按钮状态
    if(pMainFrame)
        if(nCommand==CSC_NAVIGATEFORWARD) isForward=bEnable;
        else if(nCommand==CSC_NAVIGATEBACK) isBack=bEnable;
    GetMainFrame();
    pMainFrame->SetButtonState(isForward,isBack);
    CHtmlView::OnCommandStateChange(nCommand, bEnable);
}
```

其中 isForward、isBack 是布尔型的成员变量，用于记录状态，并传递给 MainFrame。由于 CMFCRibbonButton 类没有提供禁用的方法，我们需要自行发送消息来禁用按钮。MainFrame 的相应代码如下：

```
ON_UPDATE_COMMAND_UI(ID_EDIT_BACK,&CMainFrame::ButtonEnable)
ON_UPDATE_COMMAND_UI(ID_EDIT_FORWARD,&CMainFrame::ButtonEnable)
void CMainFrame::ButtonEnable(CCmdUI *pCmdUI)
{  //设置按钮状态
    if(pCmdUI->m_nID==ID_EDIT_FORWARD)
```

```
        pCmdUI->Enable(isForward);
    else if(pCmdUI->m_nID==ID_EDIT_BACK)
        pCmdUI->Enable(isBack);
    else pCmdUI->Enable(TRUE);
}
```

另外，后退、前进的功能实现上对按钮状态进行了判断，防止出现意外情况，增强程序的稳定性。代码如下：

```
void CMyView::OnEditBack()
{  //浏览器后退
    if(isBack) GoBack();
}
void CMyView::OnEditForward()
{  //浏览器前进
    if(isForward) GoForward();
}
```

Web 浏览器的刷新、停止功能可直接调用相应方法，主页即访问百度首页，查找、打印、另存为、全选等功能使用的是 CHtmlView 的 ExecWB 方法，具体使用方法不再赘述，可参考微软 MSDN 的文档。

2.2.5 页面缩放模块设计

页面缩放用到了 CMFCRibbonSlider 类作为缩放控件，通过 IHTMLDocument2 接口修改网页 Document 的缩放级别。

代码如下：

```
void CMyView::OnSlider()
{  //页面缩放
    GetMainFrame();
    double fZoom=pMainFrame->GetZoom();
    CComPtr<IHTMLDocument2> pDoc = (IHTMLDocument2*)this->
GetHtmlDocument();
    ASSERT(pDoc);
```

```
        CComPtr<IHTMLElement> pElem;
        pDoc->get_body(&pElem);
        ASSERT(pElem);
        CComPtr<IHTMLStyle> pStyle;
        pElem->get_style(&pStyle);
        CString str;
        str.Format(L"zoom:%f;",fZoom);
        pStyle->put_cssText(str.AllocSysString());
    }
```

2.2.6 状态栏模块设计

浏览器的状态栏使用了 CMFCRibbonStatusBar 类，并分为三个 Panel，第一个部分使用 CMFCRibbonStatic 类显示浏览器的状态，比如加载图片、完成等。第二个部分使用 CMFCRibbonProgressBar 类以进度条的形式显示浏览器加载进度。第三个部分是静态文本，始终显示”Crazy Urus 浏览器”。如图 4 所示：

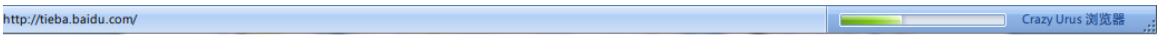

图 4 状态栏的三个部分

创建 Panel 对象的代码如下：

```
    m_wndStatusBar.AddDynamicElement(new CMFCRibbonLabel(
strTitlePane1,FALSE););
    m_wndStatusBar.AddExtendedElement(new CMFCRibbonProgressBar(
ID_STATUSBAR_PROGRESS,160),L"进度");
    m_wndStatusBar.AddSeparator();
    m_wndStatusBar.AddExtendedElement(new CMFCRibbonStatusBarPane
(ID_STATUSBAR_PANE2, strTitlePane2, TRUE), strTitlePane2);
```

当 Web 浏览器执行加载过程时，会触发 OnProgressChange 事件，同时将当前进度和进度最大值返回。程序将返回的数据传递给 MainFrame 的进度条即可。代码如下：

```
    void CMyView::OnProgressChange(long nProgress,long nProgressMax)
    {  //浏览器进度改变，修改程序的进度条
        GetMainFrame();
```

```
    if(pMainFrame)
        pMainFrame->SetProgress(nProgress,nProgressMax);
    CHtmlView::OnProgressChange(nProgress, nProgressMax);
}
```

修改进度条的代码如下：

```
void CMainFrame::SetProgress(long value,long max)
{  //设置进度条
    if(ProgressBar)
    {
        ProgressBar->SetRange(0,max);
        ProgressBar->SetPos(value);
    }
}
```

浏览器状态修改原理与进度条类似，当状态改变时会触发 OnStatusTextChange 事件，返回状态内容。代码如下：

```
void CMyView::OnStatusTextChange(LPCTSTR lpszText)
{  //浏览器状态改变，修改程序的状态栏
    GetMainFrame();
    if(pMainFrame) pMainFrame->SetStatusText(lpszText);
    CHtmlView::OnStatusTextChange(lpszText);
}
```

修改状态文本的代码如下：

```
void CMainFrame::SetStatusText(LPCTSTR str)
{  //设置状态栏文本
    CString tmp=str;
    static BOOL isFinish=TRUE;
    if(StatusBar&&!tmp.IsEmpty()&&isFinish)
        StatusBar->SetText(str);
    isFinish=tmp.Find(L"完成",0);
```

```
}
```

2.2.7 收藏夹模块设计

收藏夹模块是一个较为简单的模块，因为本浏览器采用了预设的机制，为使用者提供了一些常用的网站，使用者并不能修改收藏夹。目前收藏夹对网站分成了导航、搜索、门户、电子商务、社交和其它几类。每个收藏的网页都对应一个按钮，点击之后会执行相应操作，Web 浏览器执行 Navigate 打开相应网页。

图 5　收藏夹

2.2.8 窗体关闭模块设计

窗体关闭模块是用来提醒用户，防止用户错误操作的。由于本 Web 浏览器采用了标签式的浏览方式，因而容易出现关闭主窗口导致所有标签页被关闭的情况，所以需要在关闭主窗口时添加提示，告知用户是否关闭所有标签页，或者只关闭当前的标签页。本程序添加了一个对话框窗口 IDD_CLOSE，如图 6 所示：

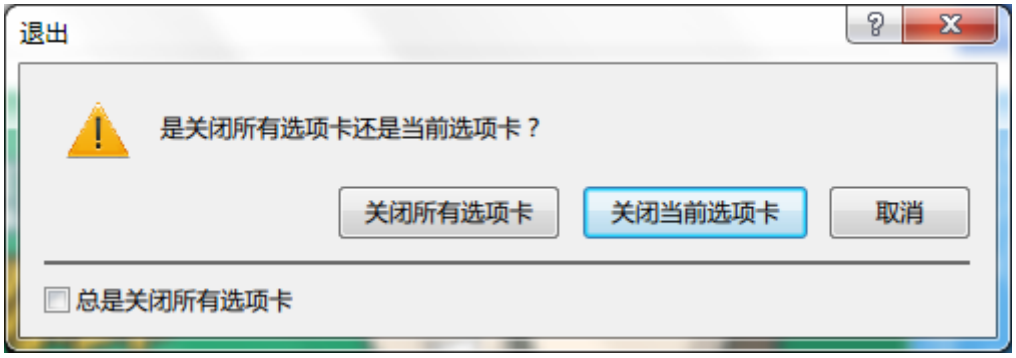

图 6　关闭提示对话框

MainFrame 里的 OnClose 函数映射到消息 WM_CLOSE，代码如下：

```
void CMainFrame::OnClose()
{
    CCloseDlg closeDlg;
    UINT r=closeDlg.DoModal();  //打开关闭对话框
    if(r==IDALL) CMDIFrameWndEx::OnClose();
```

```
    else if(r==IDCUR) GetActiveFrame()->DestroyWindow();
}
```

关闭对话框将以模态打开，并利用 EndDialog 函数实现点击不同的按钮返回不同的值。MainFrame 根据返回值执行不同的操作，即可实现不同的关闭方式。

此外，为了防止用户厌烦此对话框，本程序添加了一个 CheckBox 控件。如果用户勾选，则不再出现此对话框。

2.3 系统平台、语言和工具

本次课程设计在操作系统为 64 位 Windows 7 的个人计算机上完成，采用 C / C＋＋程序设计语言编写代码，Visual Studio 2010 开发和编译程序。

由于 MFC 的 Ribbon 界面库具有良好的兼容性，本 Web 浏览器程序可在 Windows XP、Windows Vista、Windows 7、Windows 8、Windows 8.1 以及更高版本 Windows 操作系统上使用，并且使用了 dll 静态调用技术，运行环境即便缺少 MFC 运行库本程序仍可正常使用。

3 调试过程及操作说明

3.1 启动 Web 浏览器

本 Web 的启动方式与其它 Windows 程序无任何区别，双击图标即可启动。如出现缺少 dll 等情况而无法启动，请请前往微软网站更新 MFC 运行库。本程序编译时已经将运行库编译到程序里，上述情况一般不会发生。

3.2 浏览网页

程序启动后，会显示程序的主界面，并自动打开百度首页。如果希望输入网址访问指定网页，请在程序右上角的地址栏中输入网址，并按下回车，浏览器就会在当前窗口加载此网页。如果希望访问收藏家中的网页，可直接点击收藏夹下的按钮或图标，浏览器会在当前窗口加载。如果希望在新窗口打开网页，请先点击“新建”，出现新窗口后再输入网址访问。

图 7　浏览网页

此外，本 Web 浏览器提供了一系列对网页的操作，可以使用后退、前进、刷新、停止等功能，直接点击相应按钮即可使用。另外，本 Web 浏览器还提供了查找、打印、保存、全选等功能。如果希望将当前网页保存到本地，可以点击“保存”按钮，程序会弹出对话框，选择位置即可保存。如果希望打印网页，可以直接点击“打印”按钮，在打印对话框中配置好参数，即可打印。如果在打印之前希望预览打印效果，可点击“打印”按钮下的小箭头，出现下拉菜单，并点击“打印预览”。如下图 8 所示：

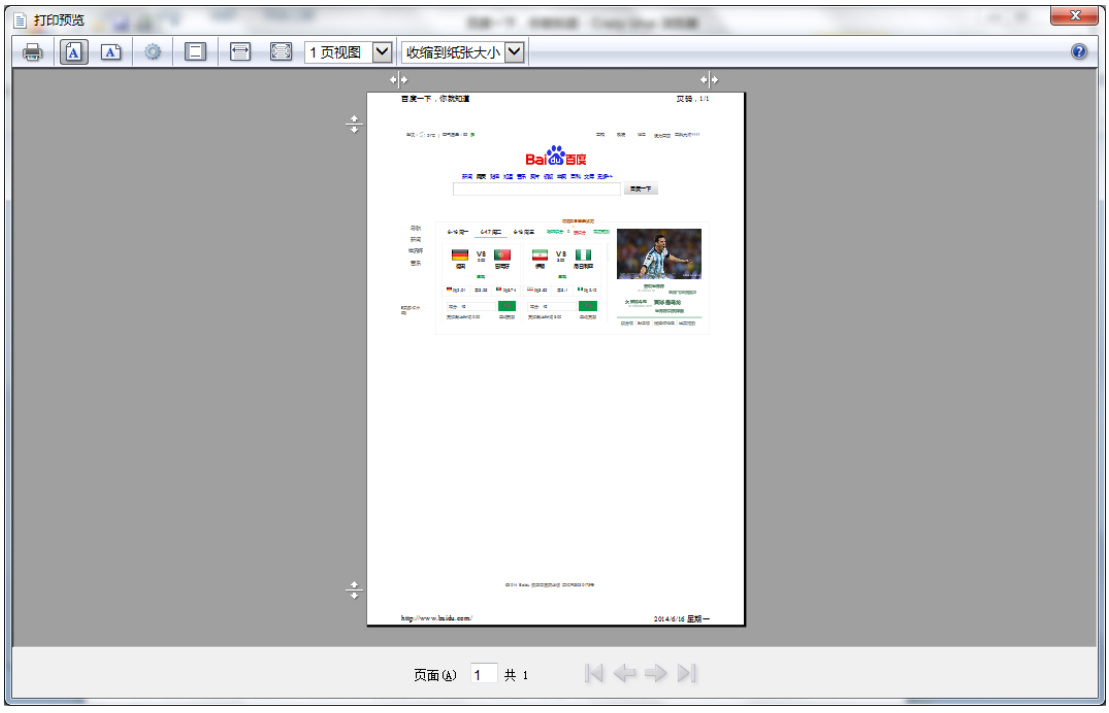

图 8　打印预览

如果在打印之前需要对页面大小进行设置，可在下拉菜单中点击“页面设置”。

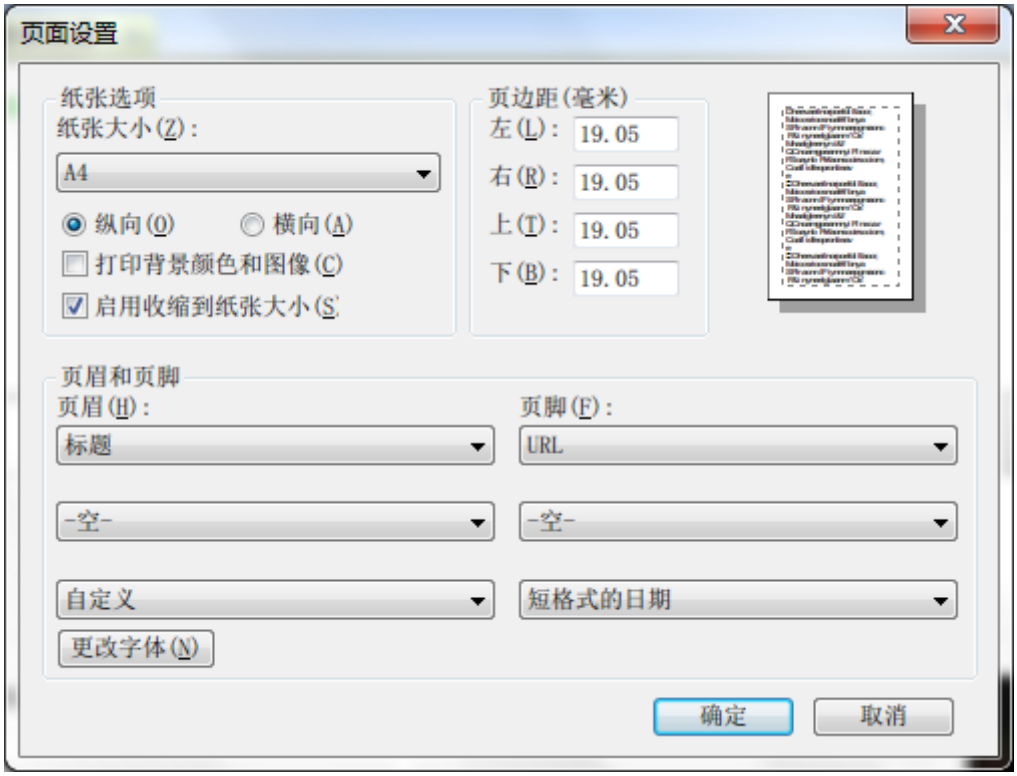

图 9　页面设置

如果需要在当前网页查找内容，可点击“查找”，并在对话框中输入内容。

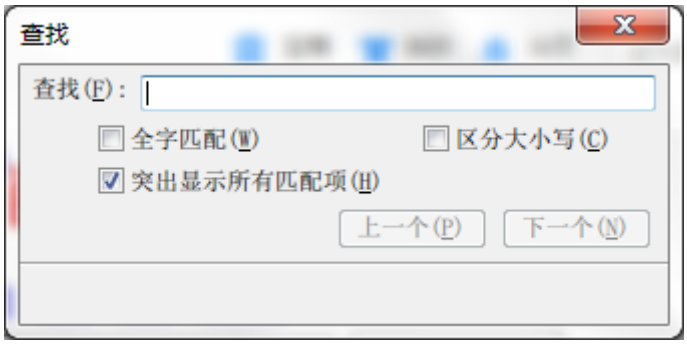

图 10　查找对话框

如果感觉页面字体较大或较小，可通过“视图”中的页面缩放功能调整字体大小。拖动滑动条，页面字体大小会随之变化。

3.3　修改界面样式及查看帮助

本 Web 浏览器提供了 5 种界面样式，其中 4 种是类似 Office 2007 的样式，另外 1 种是类似 Windows 7 画图程序的样式。如需更换程序界面的样式，请点击程序右上角的“样式”按钮，会出现下拉菜单，在下拉菜单中可以选择喜欢的样式。

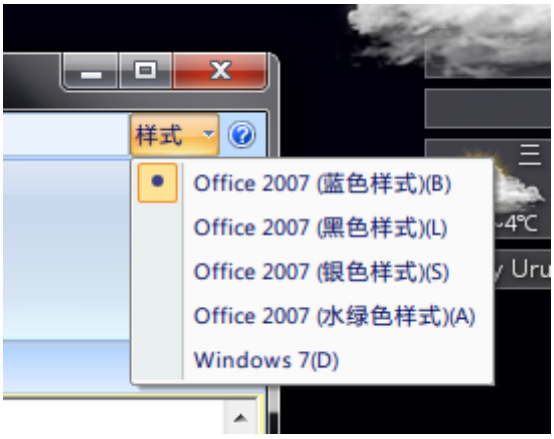

图 11　样式菜单

如果需要帮助，可点击右上角的“帮助”按钮，或者功能区中的“帮助”按钮，即会打开帮助文件。由于此次课程设计时间有限，帮助文件并未编写，打开之后里面并没有实质内容，因而使用过程中需要帮助还请参考此设计报告。

如果需要查看此程序的开发信息，请点击功能区的“关于”按钮，会弹出关于对话框。

图 12　关于对话框

关于对话框中较为详细的说明了程序的一些信息，并且提供了链接访问作者的个人主页，可以查看作者的其它一些作品。

3.4　退出程序

退出程序的方式有三种，一是直接在主窗口点击右上角的关闭按钮，程序会弹出关闭对话框（参见图 6），点击“关闭所有选项卡”即退出程序。二是点击左上角的圆形图标（Windows 7 样式下是下拉箭头），会出现下拉菜单，在下拉菜单中点击“退出”，会直接退出程序。如

图 13 所示：

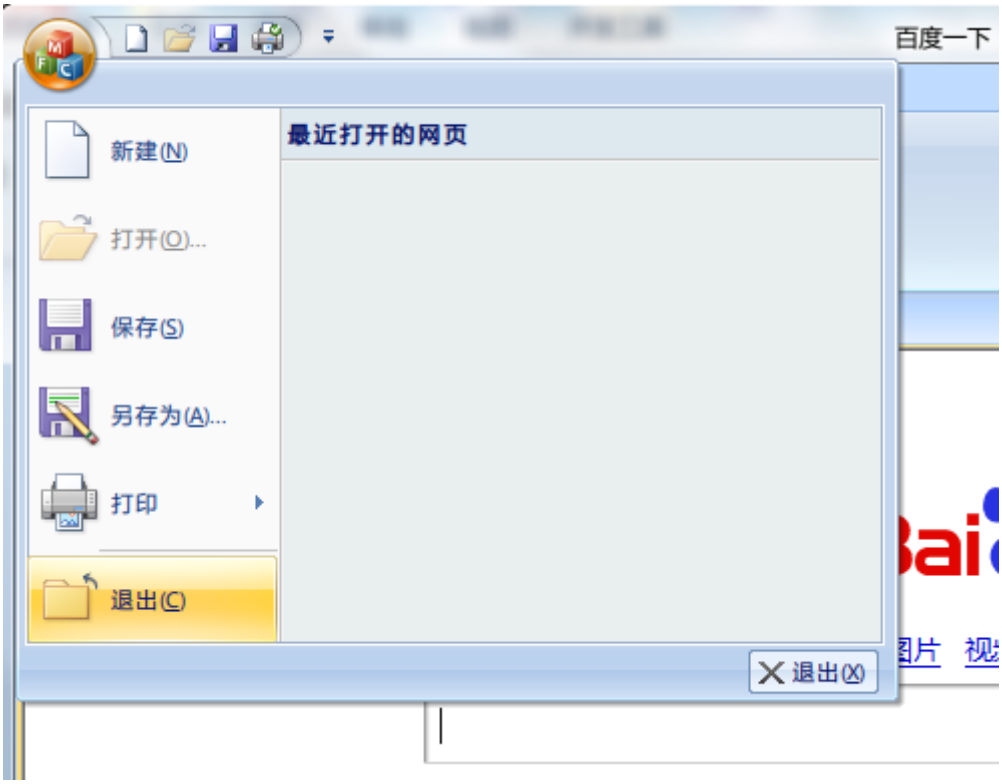

图 13　下拉菜单中退出

第三种方法是在任务栏上的程序图标，点击右键弹出菜单，在菜单中点击“关闭”，便会弹出图 3 的关闭对话框，然后按第一种方法即可退出。

4　开发中遇到的问题及解决方案

4.1　无法获取到正确的网页标题

最初开发时利用的是 GetLocationName 函数获取当前网页的标题，但经实际测试发现，部分网页的标题无法正常获取。因而改用 IHTMLDocument2 接口获取网页标题，并显示在标签上。这种方法远优于 CHtmlView 提供的函数。具体代码可参见 2.2.3 节。

4.2　多文档模式开发中获取活动窗口的问题

由于本程序是基于多文档模式的，因而有父窗口和子窗口之分。Ribbon 功能区的按钮均在父窗口上，而按钮交互的对象基本都在子窗口上，且需要对活动子窗口进行操作。参考了微软 MSDN 文档和其他开发者的一些博客，本程序采用了函数交互的方法，将需要的操作封装成函数，如 OnAddr、SetProgress 等，这样既可增加代码重用率，也便于代码的维护。另外

在 View 类中声明了 CMainFrame*和 CChildFrame*类型的成员指针，用来记录当前 View 的子框架和主框架，并通过 GetMainFrame()为两个指针获取相应的对象。这样可以用来记录，减少指针对象的获取次数。具体代码如下：

```
CMainFrame *pMainFrame;
CChildFrame *pChildFrame;
void CMyView::GetMainFrame()
{   //获取主框架和子框架并记录
    pChildFrame=(CChildFrame*)GetParentFrame();
    if(pChildFrame)
        pMainFrame=(CMainFrame*)pChildFrame->GetParentFrame();
}
```

4.3 页面缩放的问题

页面缩放功能在开发初期一直认为是很难实现的功能，当时考虑是用 GDI＋重新对网页进行渲染并缩放。但通过查阅微软 MSDN 的相关文档，发现 IHTMLDocument2 接口中提供了相关属性可供缩放，因而直接参考了相关代码实现了缩放功能。详情见 2.2.5 节。

5 目前未解决的问题

5.1 浏览器状态无法正常显示

按照总体设计，状态栏的第一个窗格是显示的 Web 浏览器状态，状态信息由 CHtmlView 发送过来，SetStatusText 函数接收。但是通过断点调试，发现实际上该函数接收到了传进来的字符串，但是 CMFCRibbonStatic 类的 SetText 方法并没有将文字修改上去，导致状态文字并没有显示在界面上。由于国内外有关 CMFCRibbon 一系列类的文档较少，MSDN 上的说明也过于简略，因而此问题一直未找到解决方案。

5.2 标签切换时地址栏内容未切换

由于地址栏改变只是响应了 OnDocumentComplete 事件，当标签页切换的时候地址栏内容并未切换，此问题将在后续版本解决。

5.3 标签切换时网页标题丢失

当切换标签时，MFC 的内部可能有一种刷新机制，导致之前通过 OnDocumentComplete 事件获取的标题全部丢失，只能显示默认的“新网页”。此问题跟 MFC 多文档框架的内部机制有关，对多文档机制进一步深入了解后会解决此问题。

6 自我评价与总结

通过本次课程设计，我对 MFC 类库，以及多文档开发框架有了更深刻的认识。经过一个星期的努力，基本达到了本次课程设计要求。

本程序由于使用了微软新推出的 Ribbon 界面库，相关开发文档较少，因而在开发过程中遇到了很多难以解决的问题。但是通过搜索引擎的大量搜索，以及在微软社区提问，解决了大部分问题。因而我意识到遇到棘手的问题并不可怕，只要用心去解决，总是可以找到解决方案的。

对于这次课程设计，我认为基本达到了我自己的要求。虽然还有一些功能不够完善。但是目前已经具有了 Web 浏览器的雏形。下一步我打算在课余时间对该浏览器继续开发，将目前存在的问题一个个解决，并且扩充浏览器目前的功能，把它打造成一款简单易用的浏览器。目前下一步的打算是为浏览器开发设置功能，并且完善内部消息机制，使其支持命令行参数、拖拽、手势等操作。有可能的话，将不拘泥于 IE 内核，引入开源的 WebKit 核心，将该浏览器设计成双核，乃至多核浏览器，并可以无缝切换核心。

这次的课程设计让我学到了很多之前没学会的东西，并使得自己一学期以来学到的东西得到实践，我感到受益匪浅。

参考文献

［1］黄维通，贾续涵．Visual C＋＋面向对象与可视化程序设计（第三版）［M］．北京：清华大学出版社，2011.6．

［2］刘雪洁，刘永纯．从零开始学 Visual C＋＋［M］．北京：清华大学出版社，2011.2．

［3］［美］Donald E. Knuth．计算机程序设计艺术 卷 1：基本算法［M］．北京：人民邮电出版社，2010.10．

［4］［美］Thomas H. Cormen，Charles E. Leiserson，Ronald L. Rivest，Clifford Stein 著，殷建平，徐云，王刚，刘晓光，苏明，邹恒明，王宏志译．算法导论［M］．北京：机械工业出版社，2013.5．

［5］陈维兴，林小茶．C＋＋面向对象程序设计教程（第三版）［M］．北京：清华大学出版社，2009.6．

［6］［美］Jeffrey Richter，［法］Christophe Nasarre 著，葛子昂，周靖，廖敏译．Windows 核心编程（第五版）［M］．北京：清华大学出版社，2008.9．

［7］［美］Stanley B.Lippman，Josée Lajoie，Barbara E.Moo 著，王刚，杨巨峰译．C＋＋Primer（第五版）［M］．北京：电子工业出版社，2013.9．

［8］微软 MSDN．CMFCRibbonBar Class［DB／OL］．http://msdn.microsoft.com/zh-cn/beginner/bb983906.aspx．

本科课程论文评分表

班级	软件工程 ZY1201 班	学号	0121210680206	姓名	廖星
论文题目	基于 MFC 的 WEB 浏览器实现				
评阅点	评分标准（细则）			分值	得分
功能及算法（40 分）	正确实现本程序所需全部功能，算法设计正确合理且有一定创意			40 分	
	实现所需功能，算法正确			30 分	
	基本实现所需功能			15 分	
	有明显重大错误			5 分	
	无法实现程序功能			0 分	
界面和操作性（20 分）	界面美观、合理，可操作性强			20 分	
	界面合理，可操作			15 分	
	界面尚可，基本可操作			10 分	
	可操作较差			5 分	
程序可读、可维护性（15 分）	程序可读性好、逻辑清晰，程序完整，可维护性好，			15 分	
	程序可读、可维护			10 分	
	基本可读可维护			5 分	
	逻辑混乱、不可读			0 分	
论文质量（25 分）	论文规范，行文流畅，层次清晰			25 分	
	论文书写基本规范，文理较通畅			20 分	
	结构较合理，层次较清楚，基本符合要求			15 分	
	结构混乱，文不对题目，或者有明显抄袭现象			5 分	
总分					

教师签名：

年　　月　　日