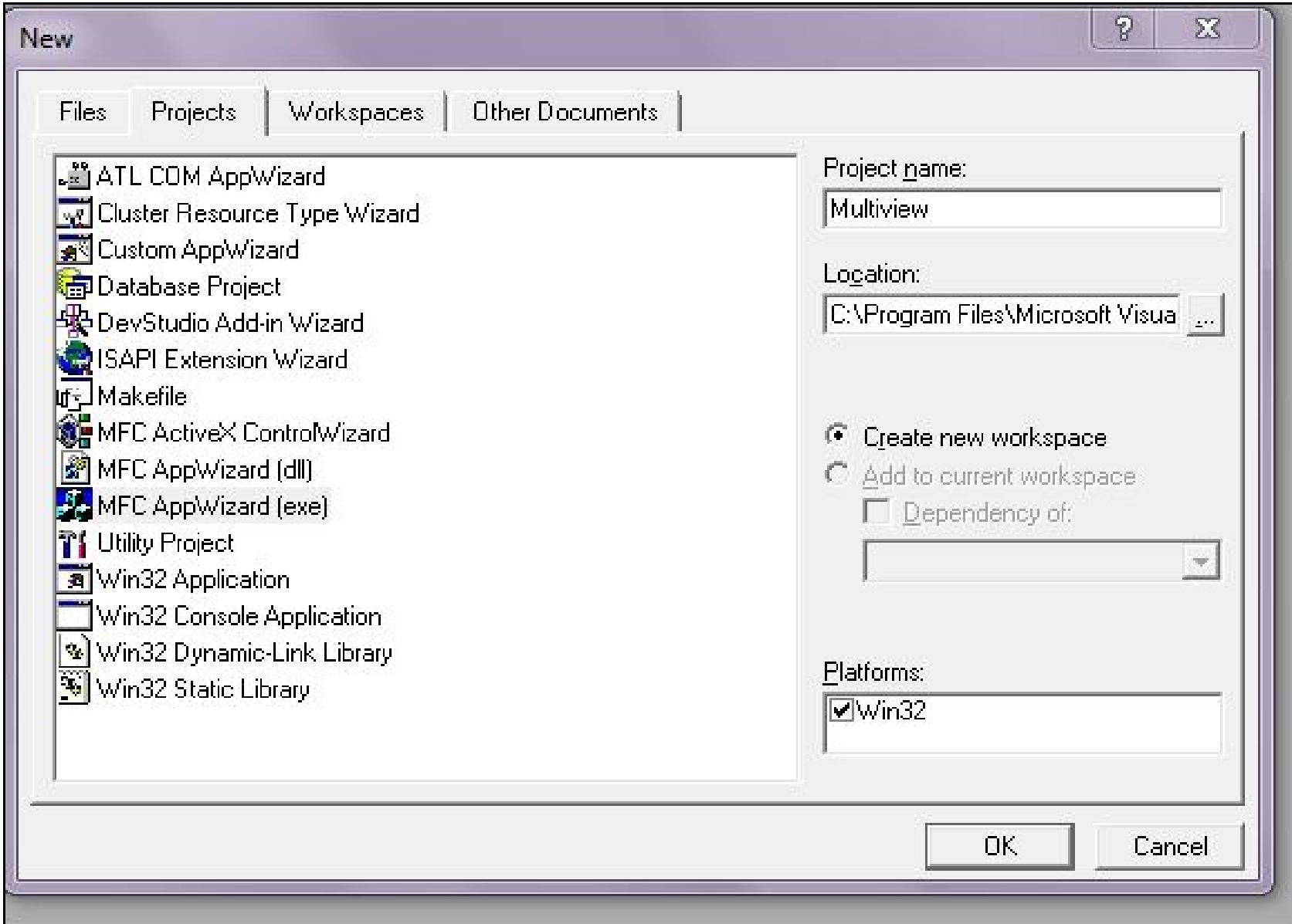


Multiple Document



MFC AppWizard - Step 1



What type of application would you like to create?

- Single document
- Multiple documents
- Dialog based

Document/View architecture support?

What language would you like your resources in?

English [United States] (APPwZENU.DLL) ▾

< Back

Next >

Finish

Cancel

MFC AppWizard - Step 6 of 6

?

X

AppWizard creates the following classes for you:

CMultiviewView
CMultiviewApp
CMainFrame
CChildFrame
CMultiviewDoc

Class name:

CMultiviewView

Header file:

Multiview.h

Base class:

CView

Implementation file:

Multiview.cpp

< Back

Next >

Finish

Cancel

MultiViewDoc.h

The screenshot shows the Microsoft Visual Studio IDE interface. On the left, the Solution Explorer displays a project named 'Multiview' with the following structure:

- Source Files:
 - ChildFrm.cpp
 - MainFrm.cpp
 - Multiview.cpp
 - Multiview.rc
 - MultiviewDoc.cpp
 - MultiviewView.cpp
 - StdAfx.cpp
- Header Files:
 - ChildFrm.h
 - MainFrm.h
 - Multiview.h
 - MultiviewDoc.h** (highlighted in blue)
 - MultiviewView.h
 - Resource.h
 - StdAfx.h
- Resource Files:
 - ReadMe.txt

The code editor on the right contains the content of the MultiviewDoc.h file:

```
#define AFX_MULTIVIEWDOC_H_C2397073_B6D8_4784_B7B8_E44D9028CFF4_INCLUDED_
#if _MSC_VER > 1000
#pragma once
#endif // _MSC_VER > 1000

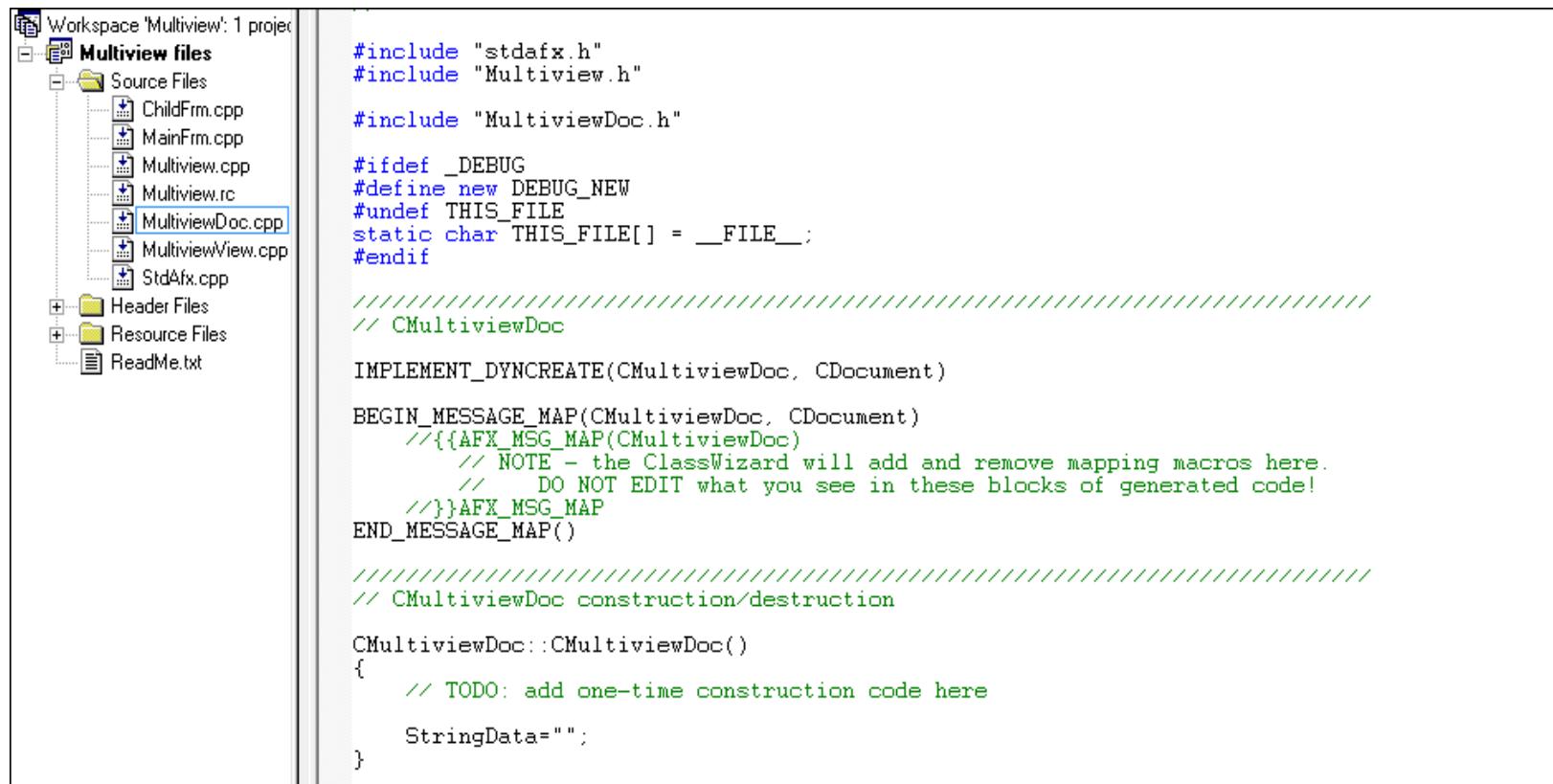
class CMultiviewDoc : public CDocument
{
protected: // create from serialization only
    CMultiviewDoc();
    DECLARE_DYNCREATE(CMultiviewDoc)

    // Attributes
public:
    CString StringData;

    // Operations
public:
    CString StringData;

    // Overrides
    // ClassWizard generated virtual function overrides
    //{{AFX_VIRTUAL(CMultiviewDoc)
public:
    virtual BOOL OnNewDocument();
    virtual void Serialize(CArchive& ar);
//}}AFX_VIRTUAL
```

MultiviewDoc.cpp



The screenshot shows the Microsoft Visual Studio IDE interface. On the left is the Solution Explorer pane, which displays a project named 'Multiview' containing one source file: 'MultiviewDoc.cpp'. This file is currently selected and highlighted in blue. The main window on the right is the Code Editor, showing the content of 'MultiviewDoc.cpp'. The code is written in C++ and includes standard headers like stdafx.h and Multiview.h, defines _DEBUG, and implements message mapping for CMultiviewDoc. It also contains a constructor for CMultiviewDoc.

```
#include "stdafx.h"
#include "Multiview.h"

#include "MultiviewDoc.h"

#ifndef _DEBUG
#define new DEBUG_NEW
#undef THIS_FILE
static char THIS_FILE[] = __FILE__;
#endif

/////////// CMultiviewDoc

IMPLEMENT_DYNCREATE(CMultiviewDoc, CDocument)

BEGIN_MESSAGE_MAP(CMultiviewDoc, CDocument)
    //{{AFX_MSG_MAP(CMultiviewDoc)
        // NOTE - the ClassWizard will add and remove mapping macros here.
        // DO NOT EDIT what you see in these blocks of generated code!
    //}}AFX_MSG_MAP
END_MESSAGE_MAP()

/////////// CMultiviewDoc construction/destruction

CMultiviewDoc::CMultiviewDoc()
{
    // TODO: add one-time construction code here

    StringData="";
}
```

MultiviewDoc.cpp

The screenshot shows the Microsoft Visual Studio IDE interface. The title bar reads "CMultiviewDoc" and "Serialize". The toolbar includes standard icons for file operations, search, and project navigation. The left pane displays the "Solution Explorer" with a tree view of the project structure:

- Workspace 'Multiview': 1 project
- Multiview files
 - Source Files
 - ChildFrm.cpp
 - MainFrm.cpp
 - Multiview.cpp
 - Multiview.rc
 - MultiviewDoc.cpp
 - MultiviewView.cpp
 - StdAfx.cpp
 - Header Files
 - Resource Files
 - ReadMe.txt

The right pane contains the code editor with the following content:

```
BOOL CMultiviewDoc::OnNewDocument()
{
    if (!CDocument::OnNewDocument())
        return FALSE;

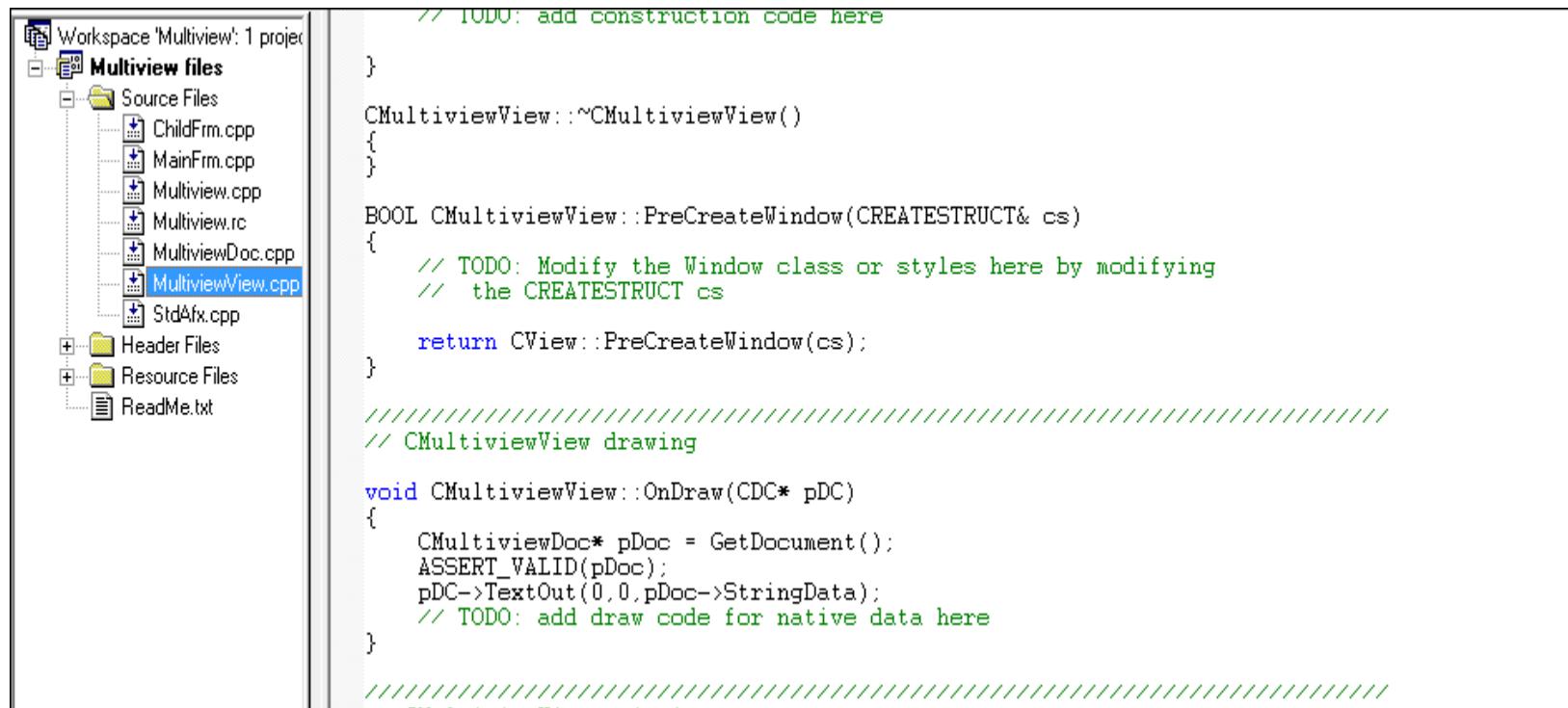
    // TODO: add reinitialization code here
    // (SDI documents will reuse this document)

    return TRUE;
}

/////////////////////////////////////////////////////////////////////////////
// CMultiviewDoc serialization

void CMultiviewDoc::Serialize(CArchive& ar)
{
    if (ar.IsStoring())
    {
        ar<< StringData;
        // TODO: add storing code here
    }
    else
    {
        ar >> StringData;
        // TODO: add loading code here
    }
}
```

MultiviewView.cpp



The screenshot shows a Windows-based IDE interface. On the left, the Solution Explorer displays a project named 'Multiview' with one item: 'Multiview files'. This folder contains several source files: ChildFrm.cpp, MainFrm.cpp, Multiview.cpp, Multiview.rc, MultiviewDoc.cpp, MultiviewView.cpp (which is currently selected), and StdAfx.cpp. It also includes Header Files, Resource Files, and a ReadMe.txt file. On the right, the main code editor window shows the content of the MultiviewView.cpp file. The code is mostly placeholder comments and includes sections for window creation and drawing.

```
// TODO: add construction code here
}

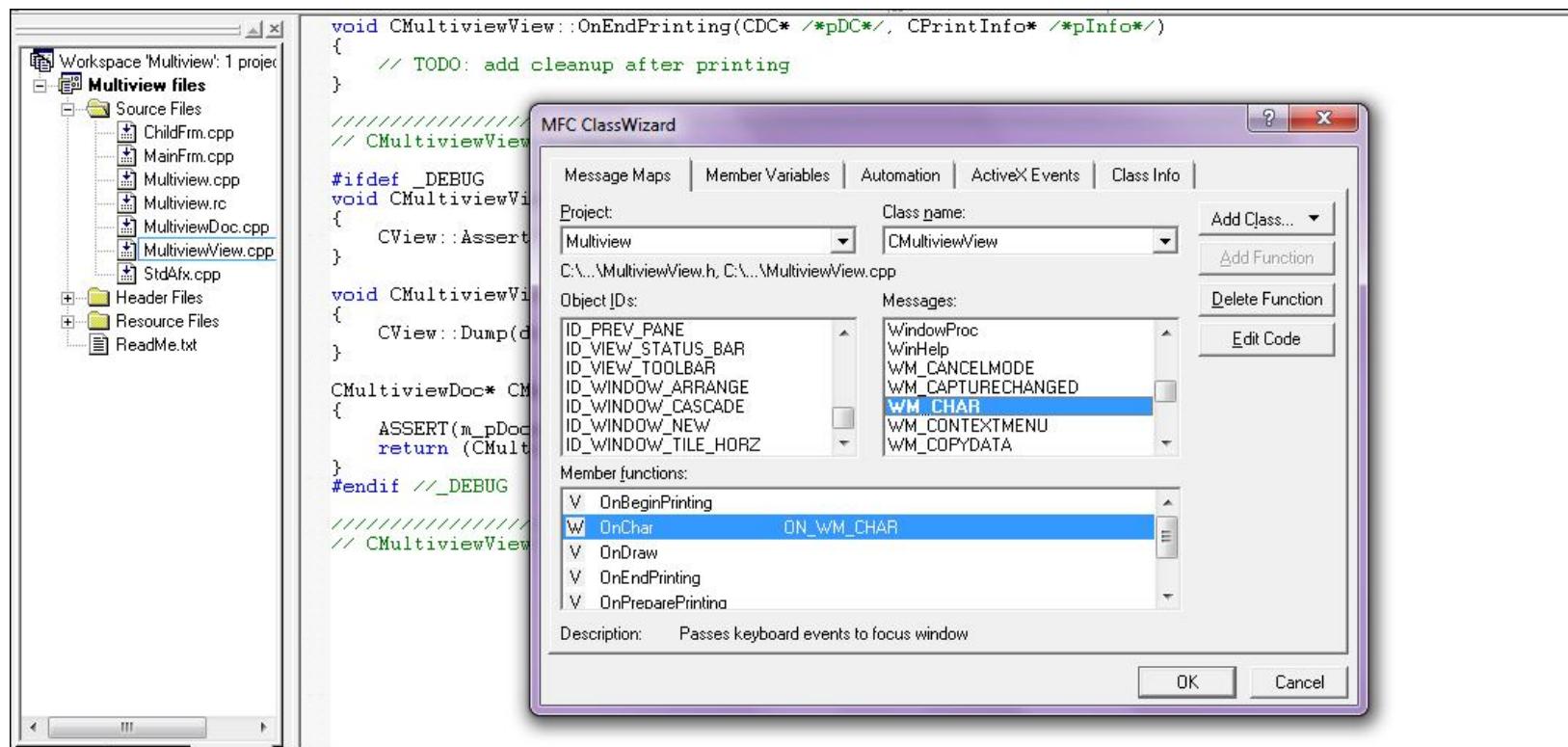
CMultiviewView::~CMultiviewView()
{
}

BOOL CMultiviewView::PreCreateWindow(CREATESTRUCT& cs)
{
    // TODO: Modify the Window class or styles here by modifying
    // the CREATESTRUCT cs
    return CView::PreCreateWindow(cs);
}

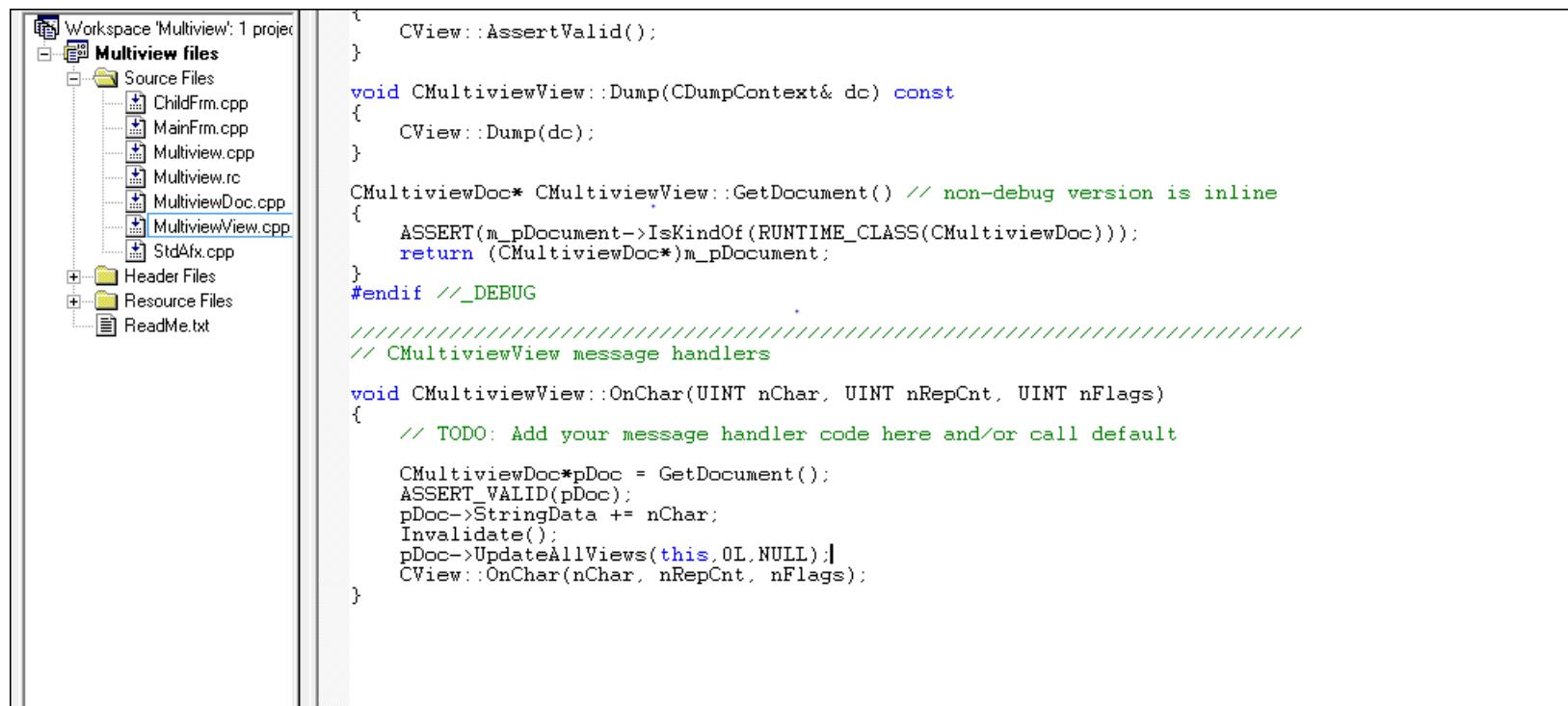
// CMultiviewView drawing

void CMultiviewView::OnDraw(CDC* pDC)
{
    CMultiviewDoc* pDoc = GetDocument();
    ASSERT_VALID(pDoc);
    pDC->TextOut(0,0,pDoc->StringData);
    // TODO: add draw code for native data here
}
```

Event Handler



Event Handler



The screenshot shows a Windows-based IDE interface with a project tree on the left and a code editor on the right.

Project Tree (Workspace 'Multiview': 1 project)

- Multiview files
 - Source Files
 - ChildFrm.cpp
 - MainFrm.cpp
 - Multiview.cpp
 - Multiview.rc
 - MultiviewDoc.cpp
 - MultiviewView.cpp** (selected)
 - StdAfx.cpp
 - Header Files
 - Resource Files
 - ReadMe.txt

Code Editor (MultiviewView.cpp)

```
    CView::AssertValid();
}

void CMultiviewView::Dump(CDumpContext& dc) const
{
    CView::Dump(dc);
}

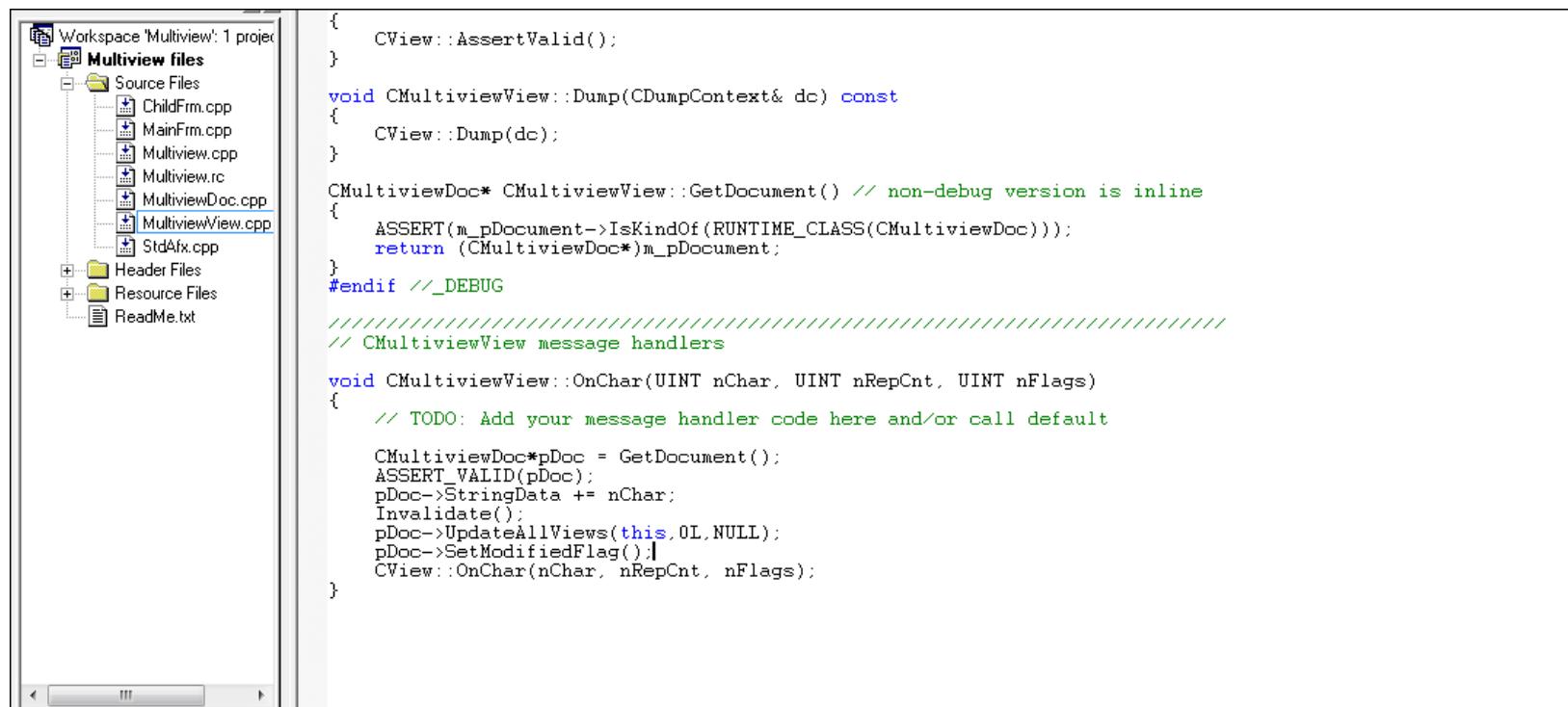
CMultiviewDoc* CMultiviewView::GetDocument() // non-debug version is inline
{
    ASSERT(m_pDocument->IsKindOf(RUNTIME_CLASS(CMultiviewDoc)));
    return (CMultiviewDoc*)m_pDocument;
}
#endif // _DEBUG

// CMultiview message handlers

void CMultiviewView::OnChar(UINT nChar, UINT nRepCnt, UINT nFlags)
{
    // TODO: Add your message handler code here and/or call default

    CMultiviewDoc*pDoc = GetDocument();
    ASSERT_VALID(pDoc);
    pDoc->StringData += nChar;
    Invalidate();
    pDoc->UpdateAllViews(this,0L,NULL);
    CView::OnChar(nChar, nRepCnt, nFlags);
}
```

Document Modified Flag



The screenshot shows a Windows-based IDE interface with a workspace titled 'Multiview' containing one project. The 'Source Files' folder under 'Multiview files' contains several files: ChildFrm.cpp, MainFrm.cpp, Multiview.cpp, Multiview.rc, MultiviewDoc.cpp, MultiviewView.cpp (which is currently selected), and StdAfx.cpp. There are also 'Header Files' and 'Resource Files' folders, along with a 'ReadMe.txt' file.

```
{  
    CView::AssertValid();  
}  
  
void CMultiviewView::Dump(CDumpContext& dc) const  
{  
    CView::Dump(dc);  
}  
  
CMultiviewDoc* CMultiviewView::GetDocument() // non-debug version is inline  
{  
    ASSERT(m_pDocument->IsKindOf(RUNTIME_CLASS(CMultiviewDoc)));  
    return (CMultiviewDoc*)m_pDocument;  
}  
#endif //_DEBUG  
  
// CMultiview message handlers  
  
void CMultiviewView::OnChar(UINT nChar, UINT nRepCnt, UINT nFlags)  
{  
    // TODO: Add your message handler code here and/or call default  
  
    CMultiviewDoc*pDoc = GetDocument();  
    ASSERT_VALID(pDoc);  
    pDoc->StringData += nChar;  
    Invalidate();  
    pDoc->UpdateAllViews(this,0L,NULL);  
    pDoc->SetModifiedFlag();  
    CView::OnChar(nChar, nRepCnt, nFlags);  
}
```

Giving a Document a Size

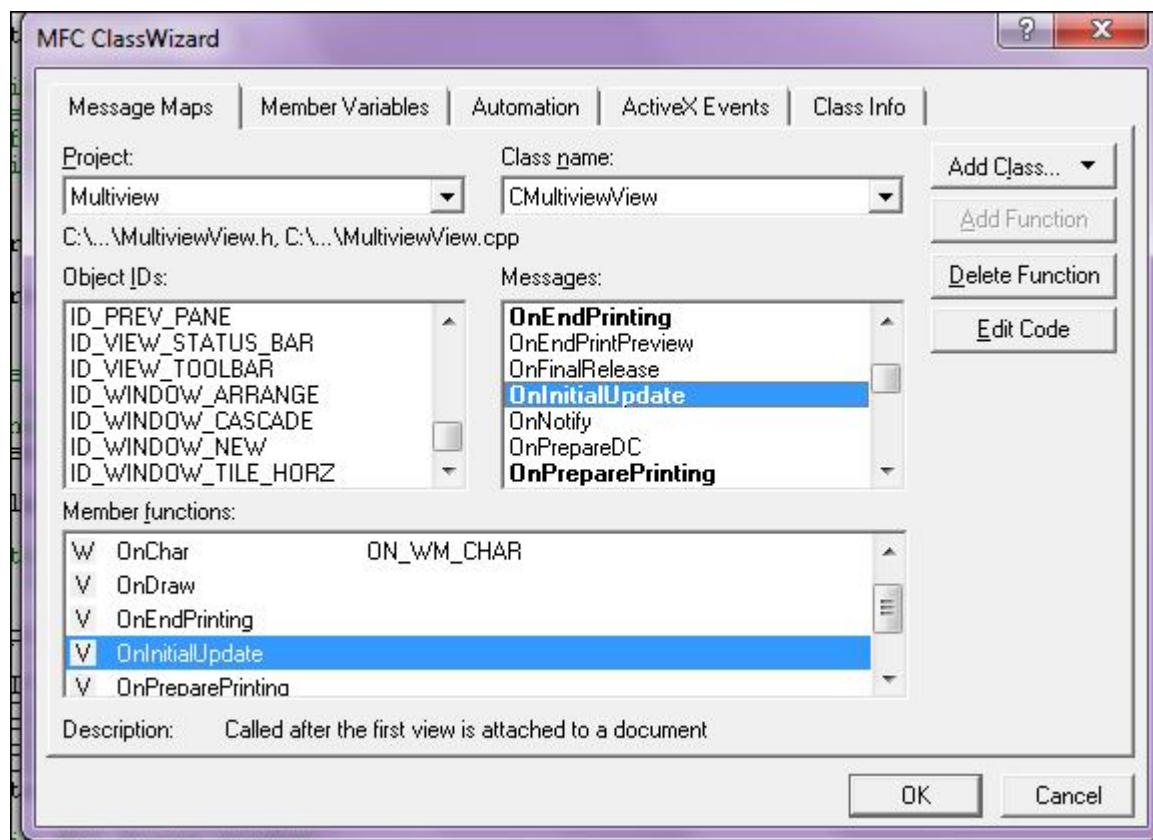
```
// MultiviewDoc.h : interface of the CMultiviewDoc class
//
///////////////////////////////////////////////////////////////////////////////
//
#ifndef AFX_MULTIVIEWDOC_H_C2397073_B6D8_4784_B7B8_E44D9028CFF4_INCLUDED_
#define AFX_MULTIVIEWDOC_H_C2397073_B6D8_4784_B7B8_E44D9028CFF4_INCLUDED_

#if _MSC_VER > 1000
#pragma once
#endif // _MSC_VER > 1000

class CMultiviewDoc : public CDocument
{
protected: // create from serialization only
    CMultiviewDoc();
    DECLARE_DYNCREATE(CMultiviewDoc)
    CSize m_size;
// Attributes
public:
    CString StringData;
// Operations
public:
    CSize GetDocSize()
    {
        return m_size;
    }
};
```

```
CMultiviewDoc::CMultiviewDoc()
{
    // TODO: add one-time construction code here
    m_size = CSize(1000 ,1000);
    StringData="";
}
```

View-> class wizard->OnInitialUpdate



```
void CMultiviewView::OnInitialUpdate()
{
    CView::OnInitialUpdate();
    CSize sizeTotal;
    CMultiviewDoc*pDoc = GetDocument();
    ASSERT_VALID(pDoc);
    sizeTotal.cx = pDoc->GetDocSize().cx;
    sizeTotal.cy = pDoc->GetDocSize().cy;
    SetScrollSizes(MM_TEXT,sizeTotal);

    // TODO: Add your specialized code here and/or call the base class
}
```