Chapter 12

Windows Programming with the Microsoft Foundation Classes

The Microsoft Foundation Classes

- MFC are a set of predefined classes.
  - These classes provides an object-oriented approach to Windows programming that encapsulates the Windows API..
  - You will apply techniques you learned from the previous chapters, particularly those involving class inheritance and virtual functions.

MFC Notation

- All the classes in MFC have names beginning with C
  - CDocument
  - CView
- Data members of an MFC class are prefixed with m_
  - m_lpCmdLine
    - Explicitly showing the type of a variable in its name was important in the C environment, because of the lack of type checking
    - However, C++ has strong type checking, so this kind of notation isn't essential, and will not be used heavily in this book.
    - However, the p prefix for pointers will be retained because this helps the code more readable.

The Document/View Concept in MFC

- Document – the collection of data
  - A document is not limited to text. It could be the data for a game, or the distribution of orange trees in California.
- View – how the data is to be displayed in a window, and how the user can interact with it.
  - A document object can have as many view objects associated with it as you want.

A Document with Two Views

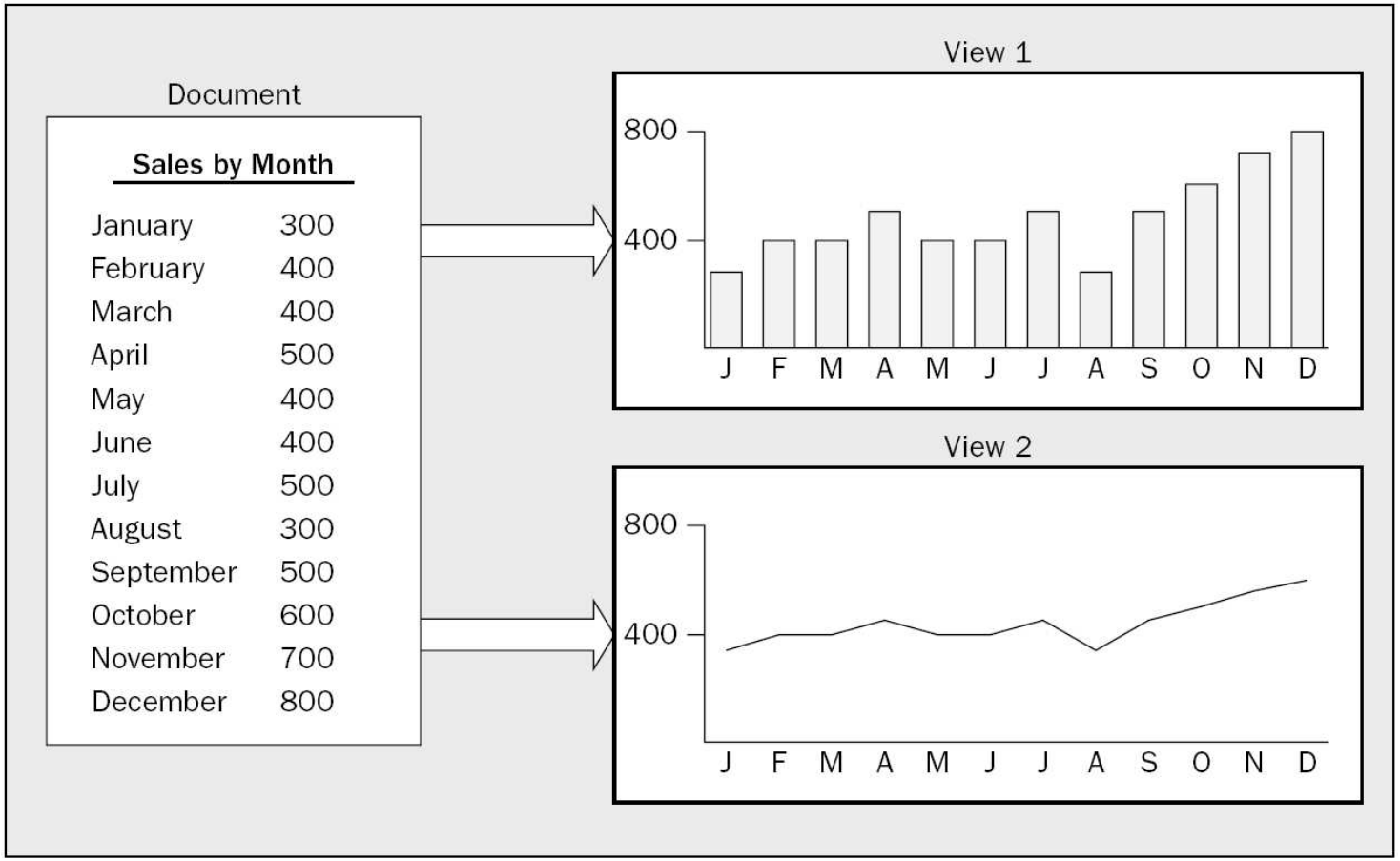


Figure 12-1

Document Interfaces

- SDI – Single Document Interface
  - Your application only open one document at a time.
- MDI – Multiple Document Interface.
  - Multiple documents can be opened in your application.
  - Each document is displayed in a child window of the application window.

Linking a Document and Its Views

- MFC incorporates a mechanism for integrating
  - a document with its views
    - a document object automatically maintains a list of pointers to its associated views
    - a view object has a pointer to the document
  - a frame window with a view
    - a frame window has a pointer to the currently active view object

Document Templates

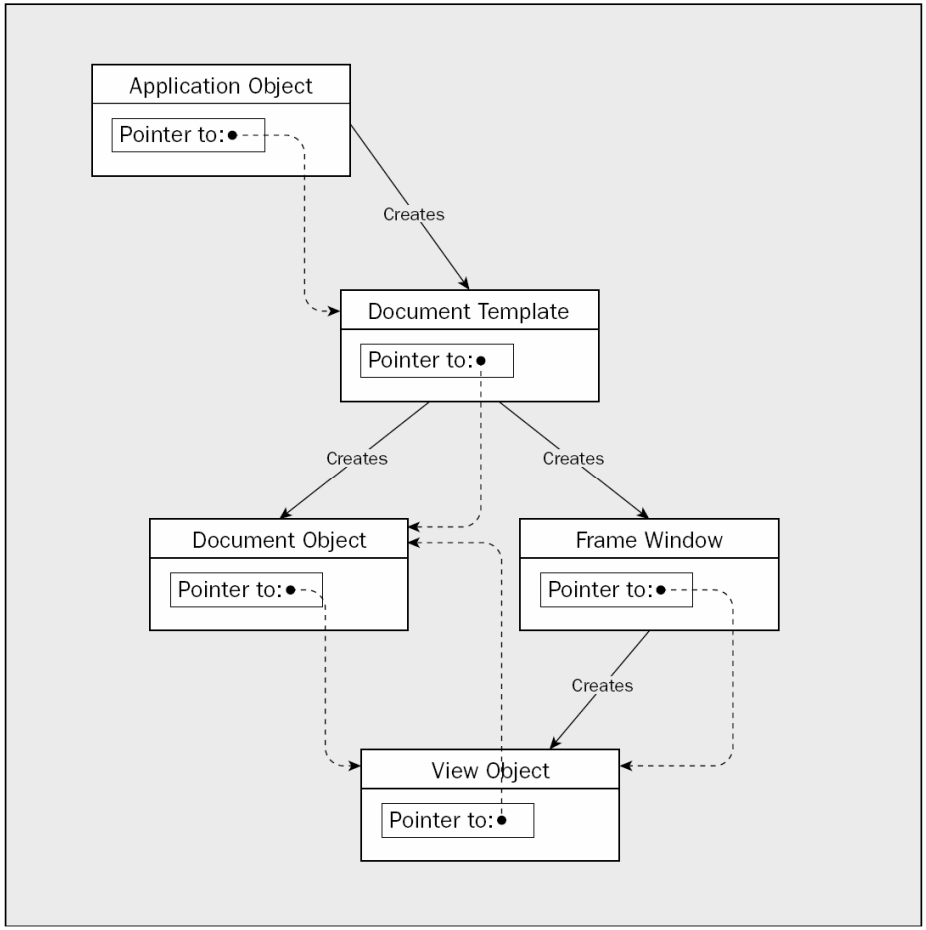


Figure 12-2

- A document template object creates document objects and frame window objects
- View of a document are created by a frame window object.
- The application object creates the document template object.

Document Template Classes

- MFC has two classes for defining document templates:
  - CSingleDocTemplate for SDI
  - CMultiDocTempalte for MDI

Document / View Classes

- Your document class is derived from the CDocument class in the MFC library
  - You will add your own data members to store items that your application requires,
  - and member functions to support processing of that data.
- Your view class is derived from the CView class.

The Application Class

- The class CWinApp is fundamental to any Windows program written using MFC.
- An object of this class includes everything necessary for starting, initializing, running and closing the application.

```
class CMyApp: public CWinApp
{
  public:
       virtual BOOL InitInstance();
};
```

The Window Class

- The CFrameWnd class provides everything for creating and managing a window for your application
  - All you need to add to the derived window class is a constructor.

```
class CMyWnd: public CFrameWnd
{
  public:
        // Constructor
        CMyWnd()
        {
              Create(0, L"Our Dumb MFC Application");
        }
};
```

Your Application and MFC

- Four basic classes that will appear in almost all your MFC-based Windows applications:
  - The application class
  - The document class
  - The view class
  - The frame window class

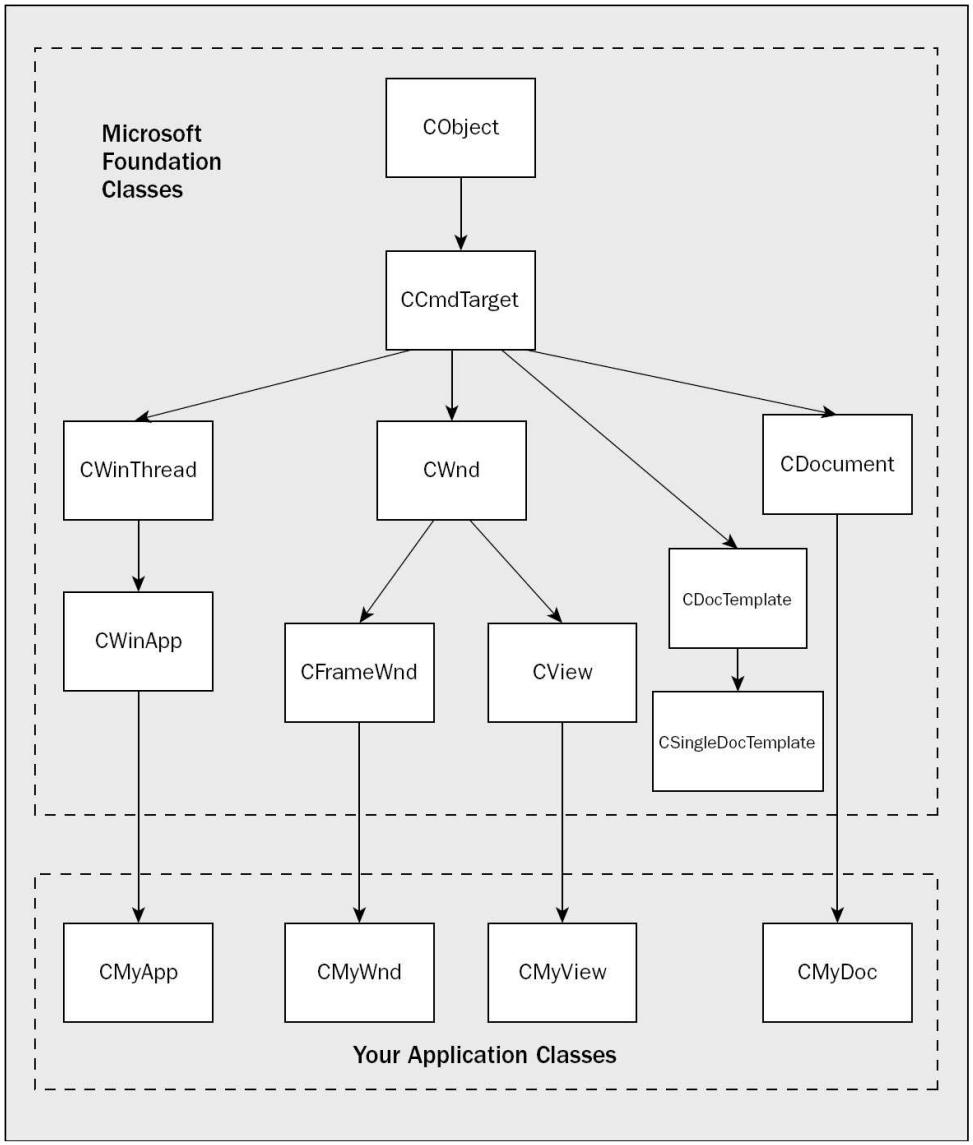


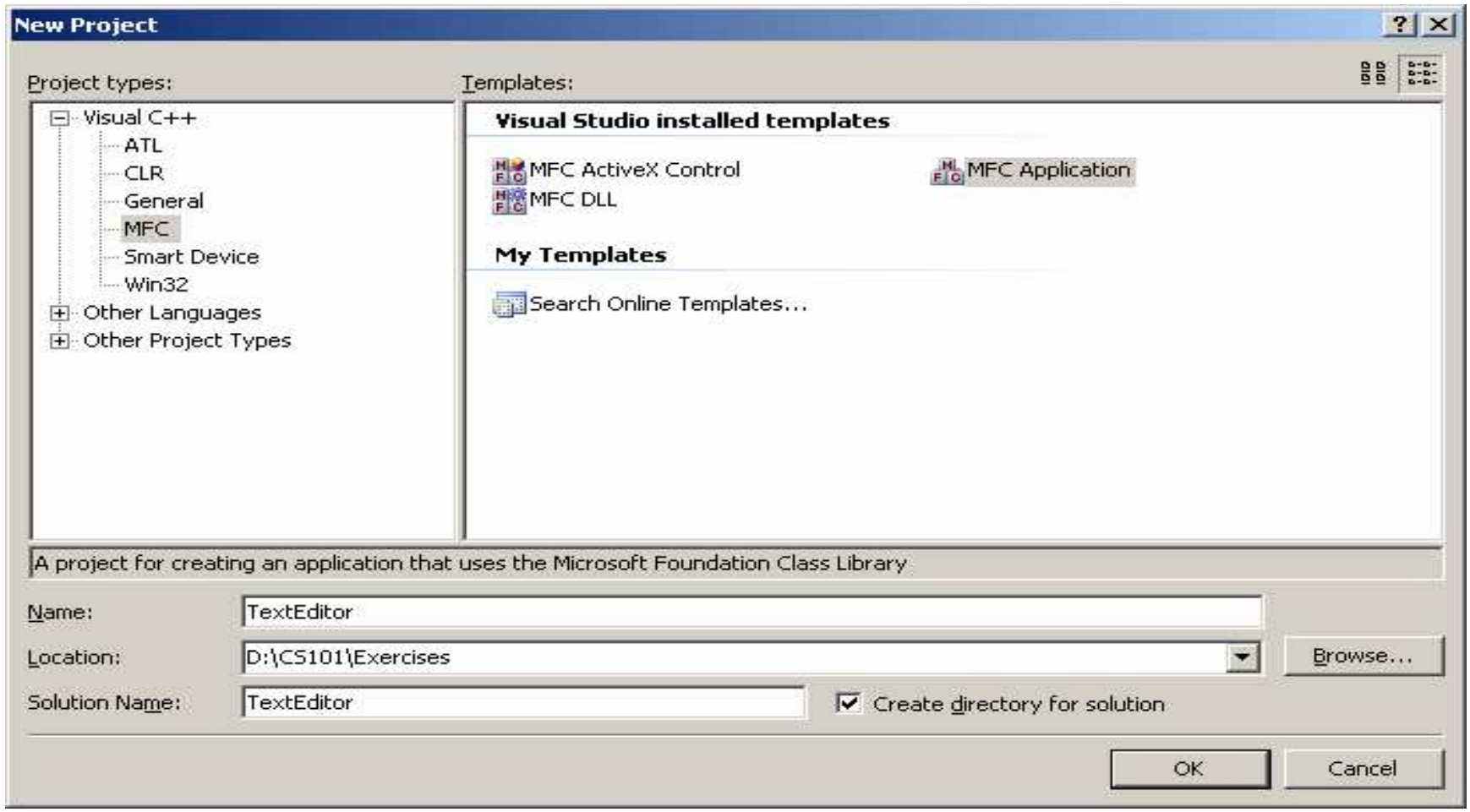
Figure 12-3

Creating MFC Applications

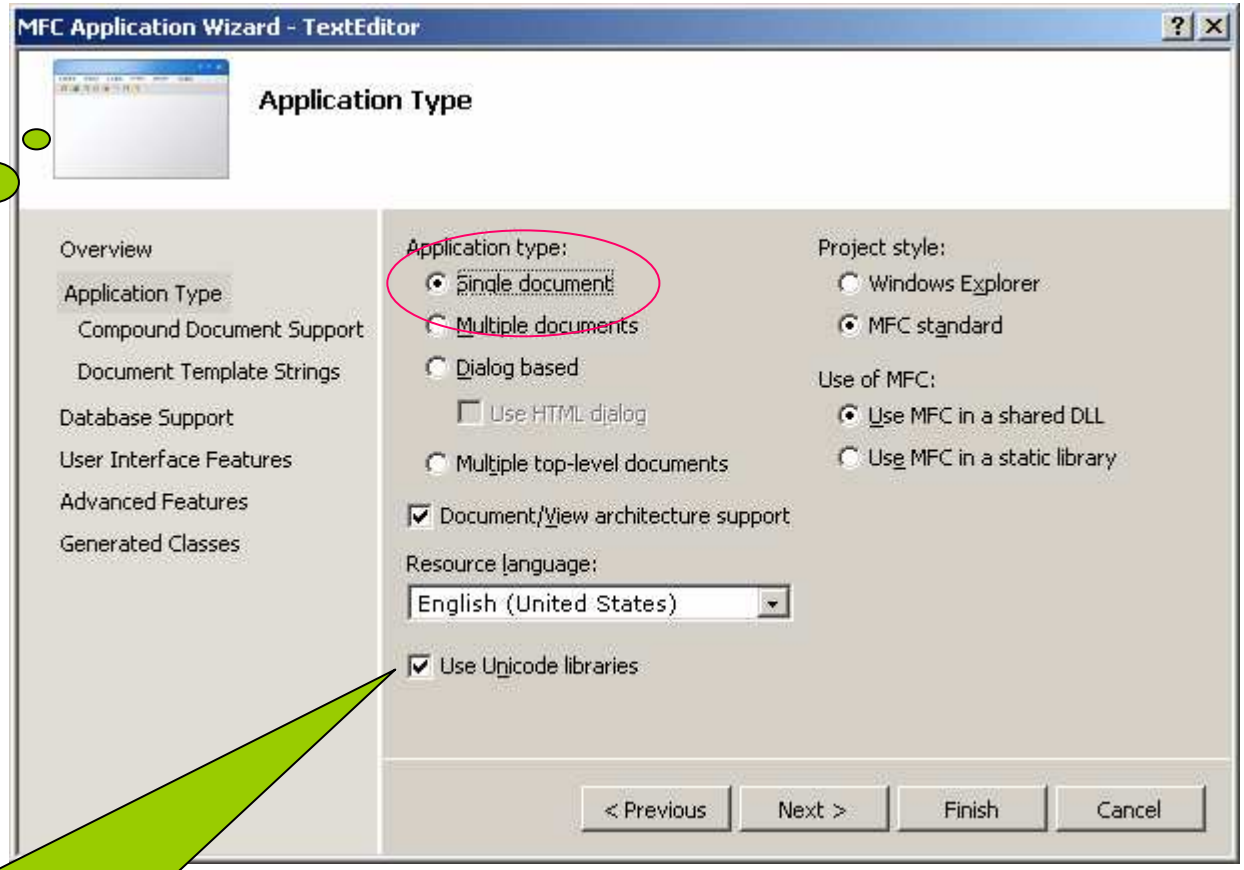
- You don't need to worry about which classes you need to have in your program.
- Visual C++ 2005 will take care of that for you.

- Create a new project
  - `File > New > Project`
  - `Crtl + Shift + N`
- Choose `MFC` as the project type and `MFC Application` as the template.

Create a New MFC Project



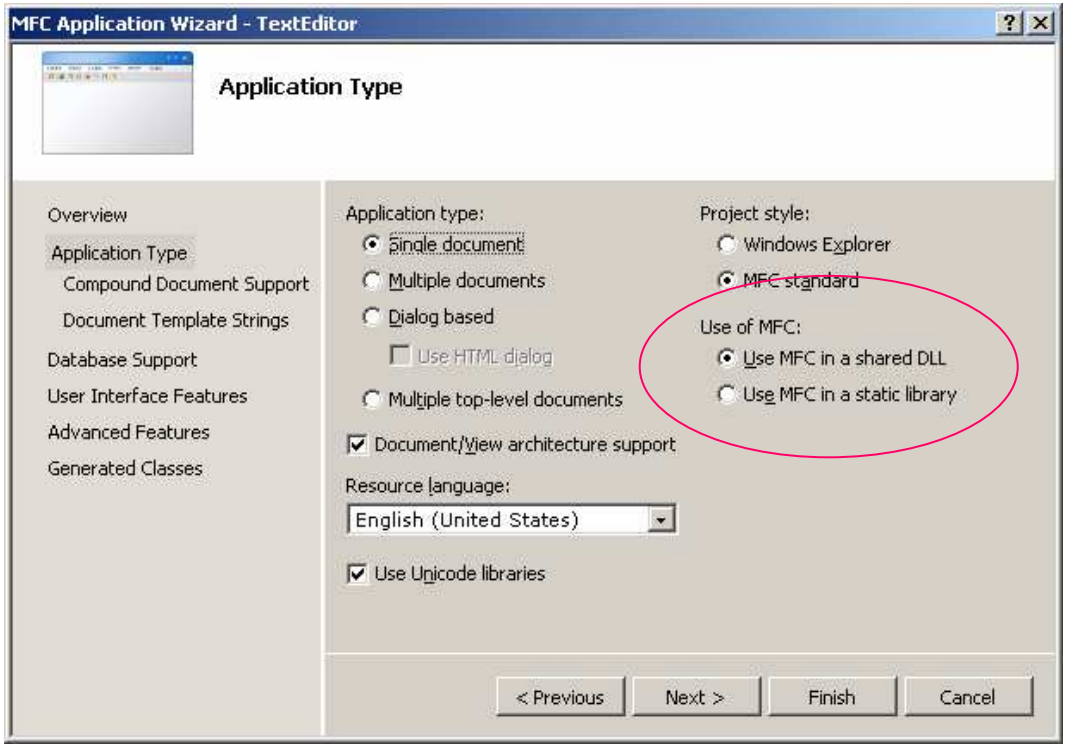
Creating an SDI Application



The appearance of an SDI application

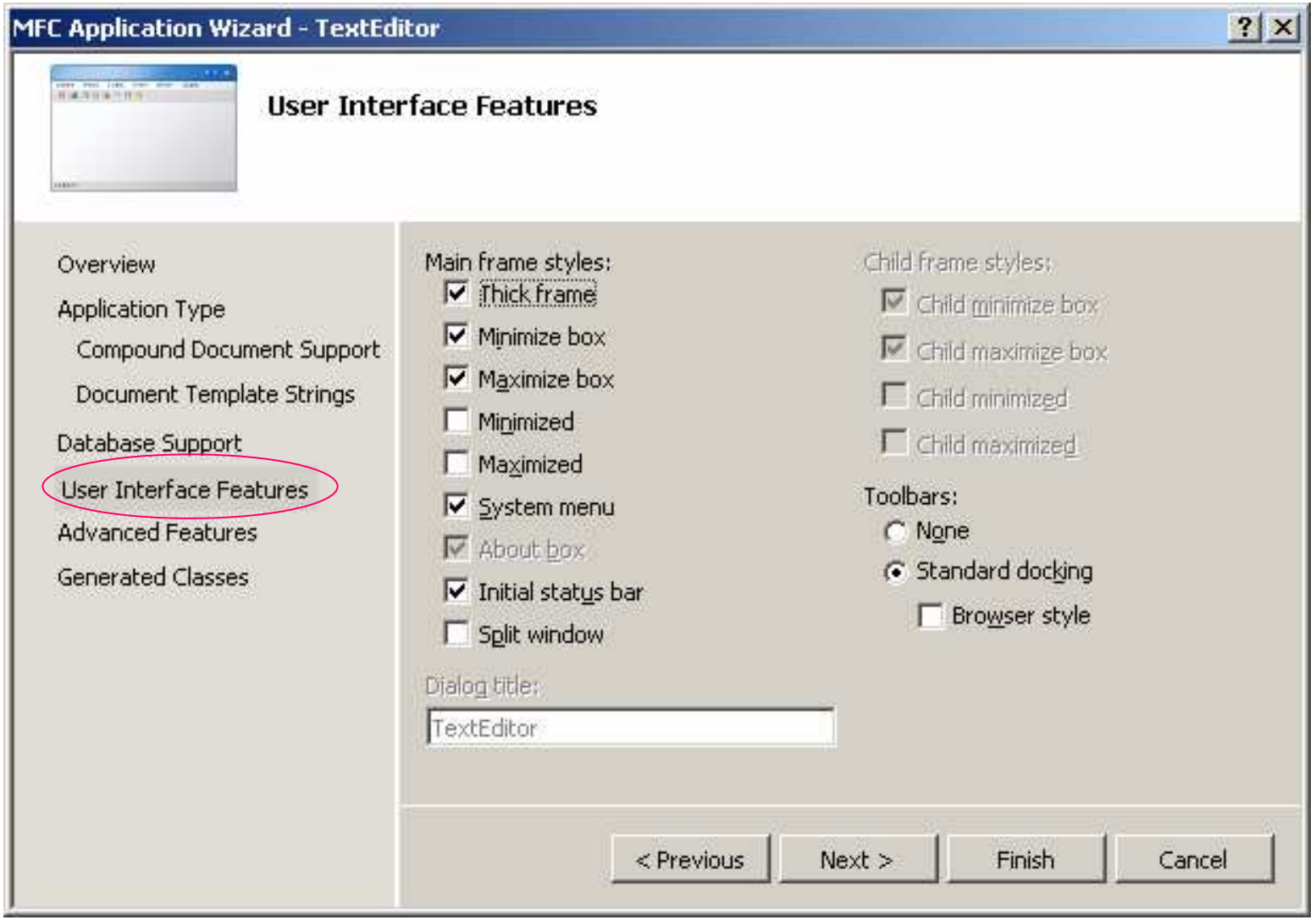
Uncheck this option if your program expect ASCII text.

Share DLL (Dynamic Link Library)

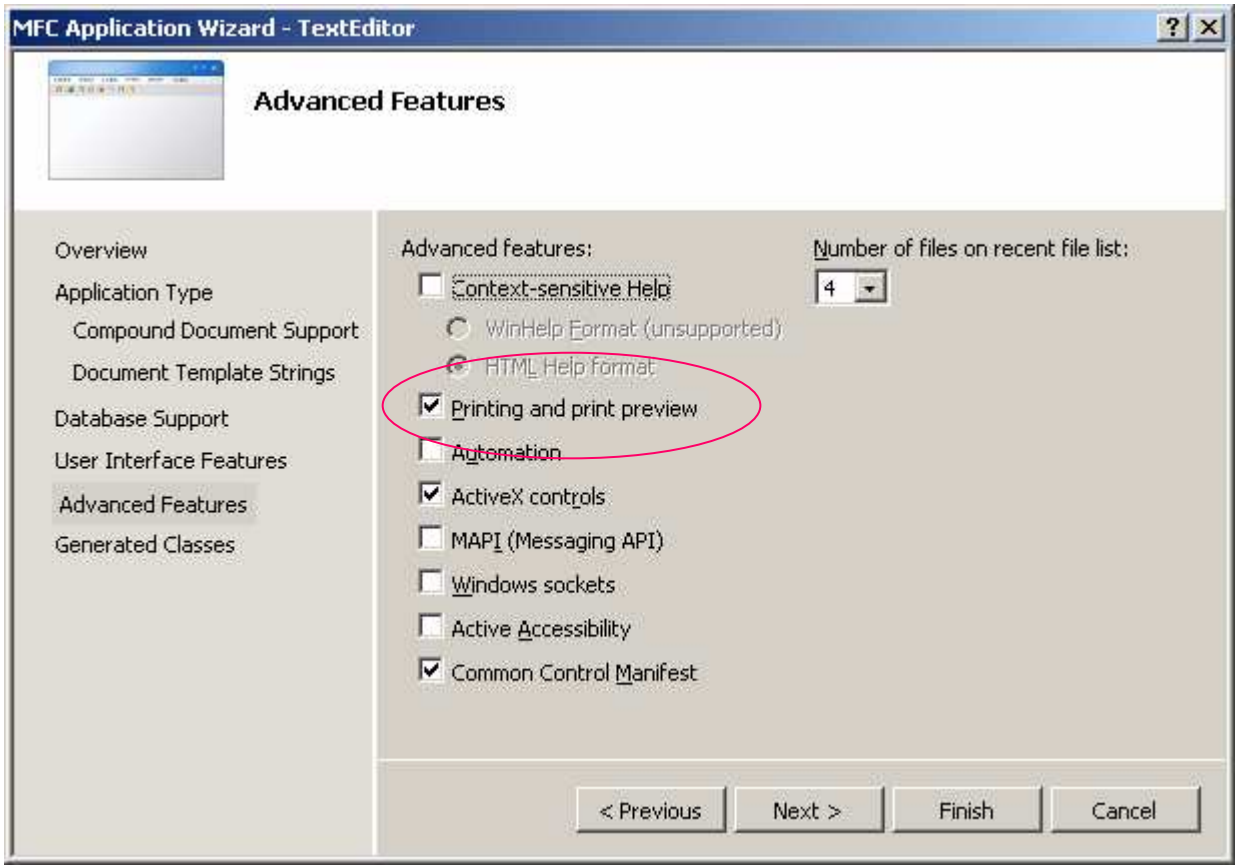


- Share DLL
  - Your program links to MFC library routines at run-time.
  - This reduces the size of the executable file.
  - When several programs are running simultaneously, they share a single copy of the library in memory.
- Statically linked
  - The library is included in the executable program you built.
  - This runs slightly faster, with the cost that the file size is larger.

User Interface Features

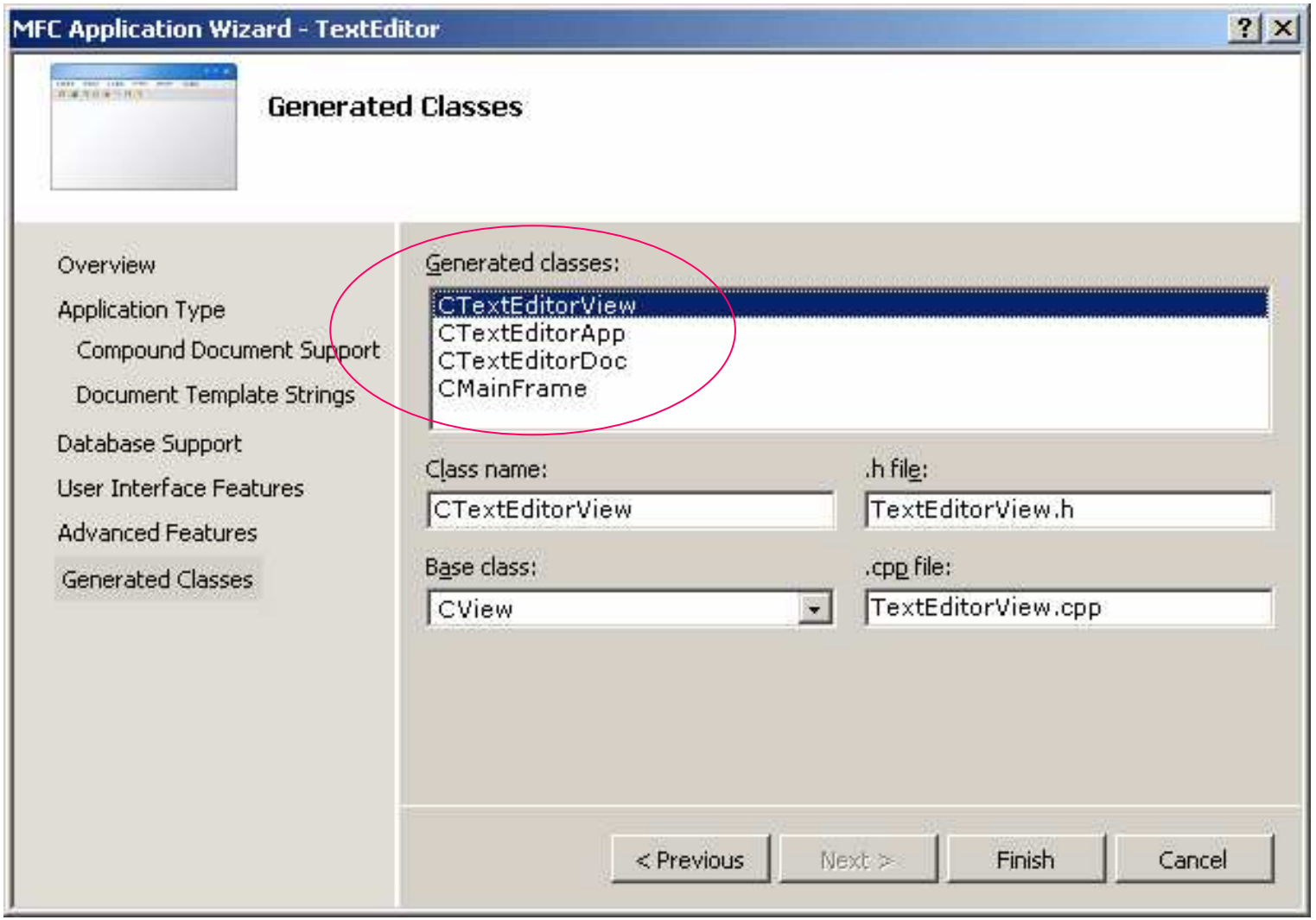


Advanced Features

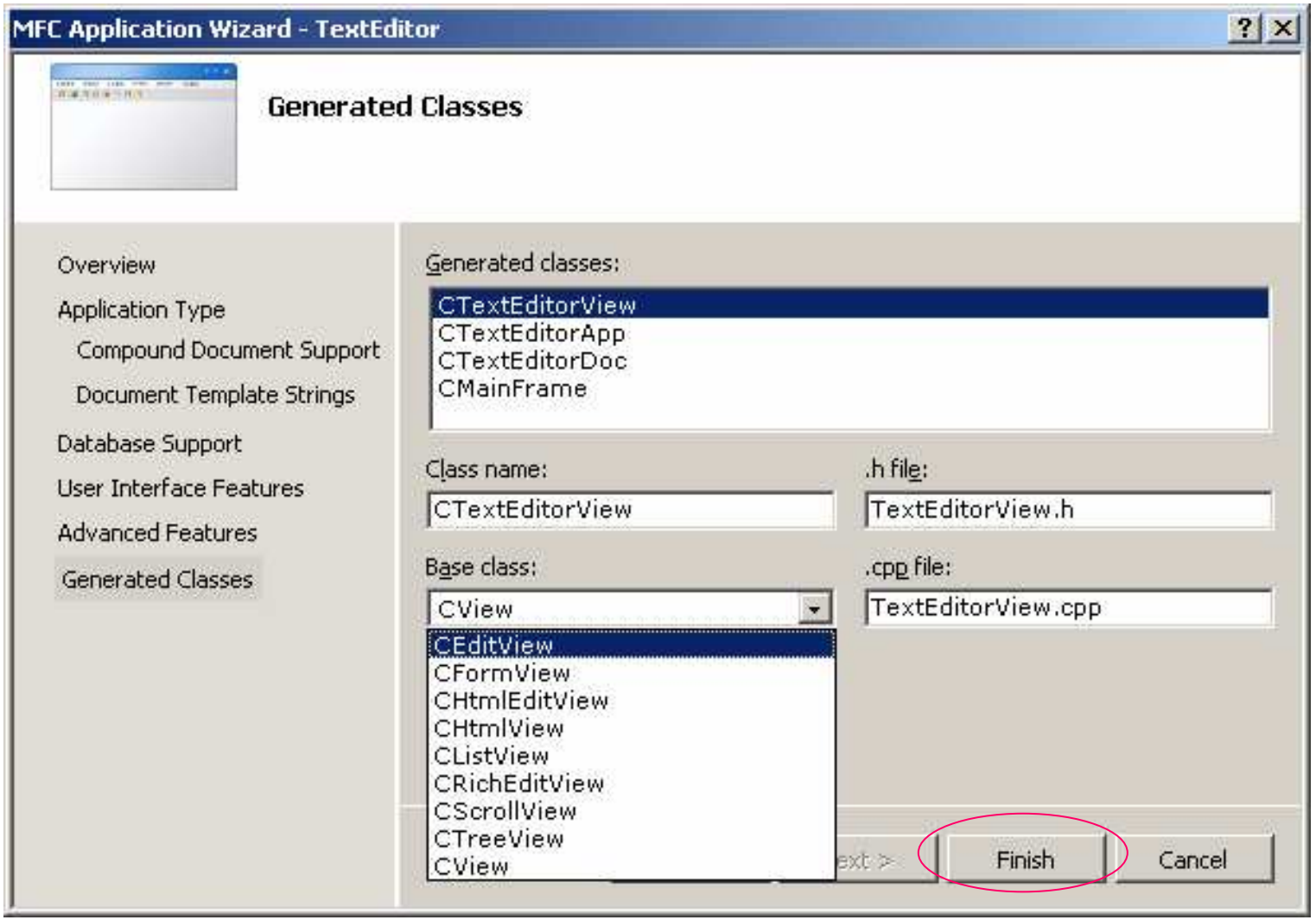


- The File menu will has the following items
  - Page Setup
  - Print Preview
  - Print
- The Application wizard also provides code to support these functions.

Generated Classes

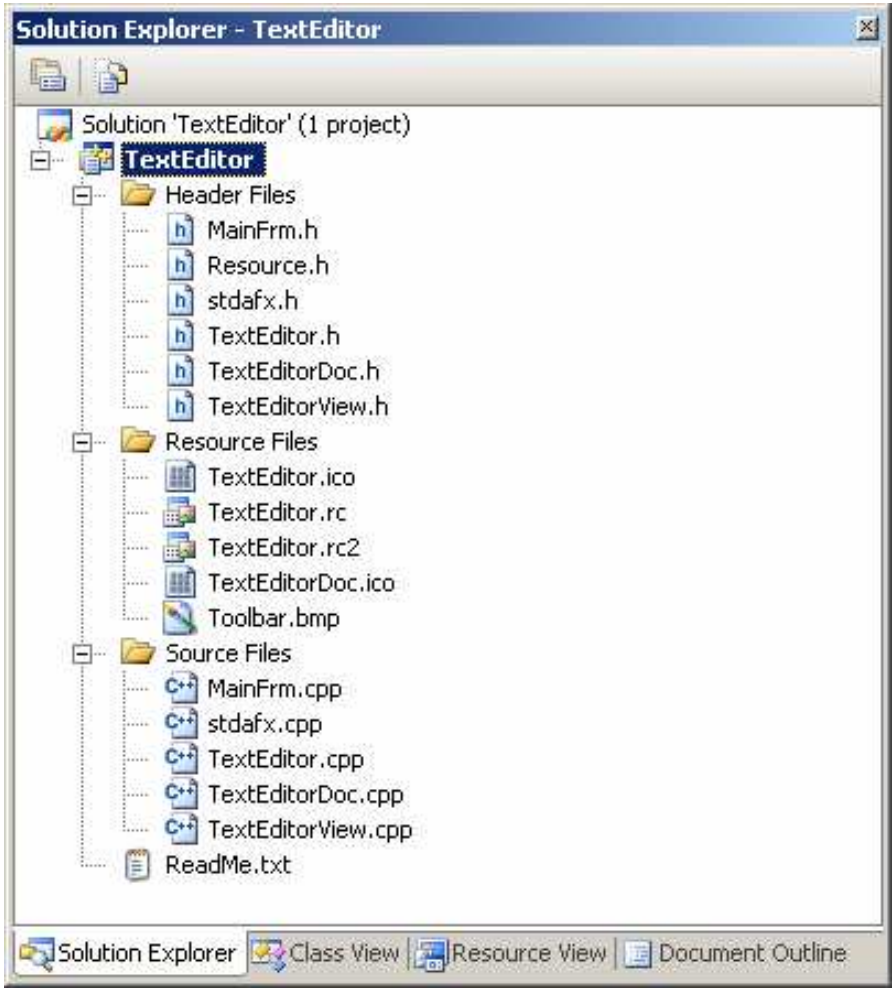


Choose CEditView as the Base class



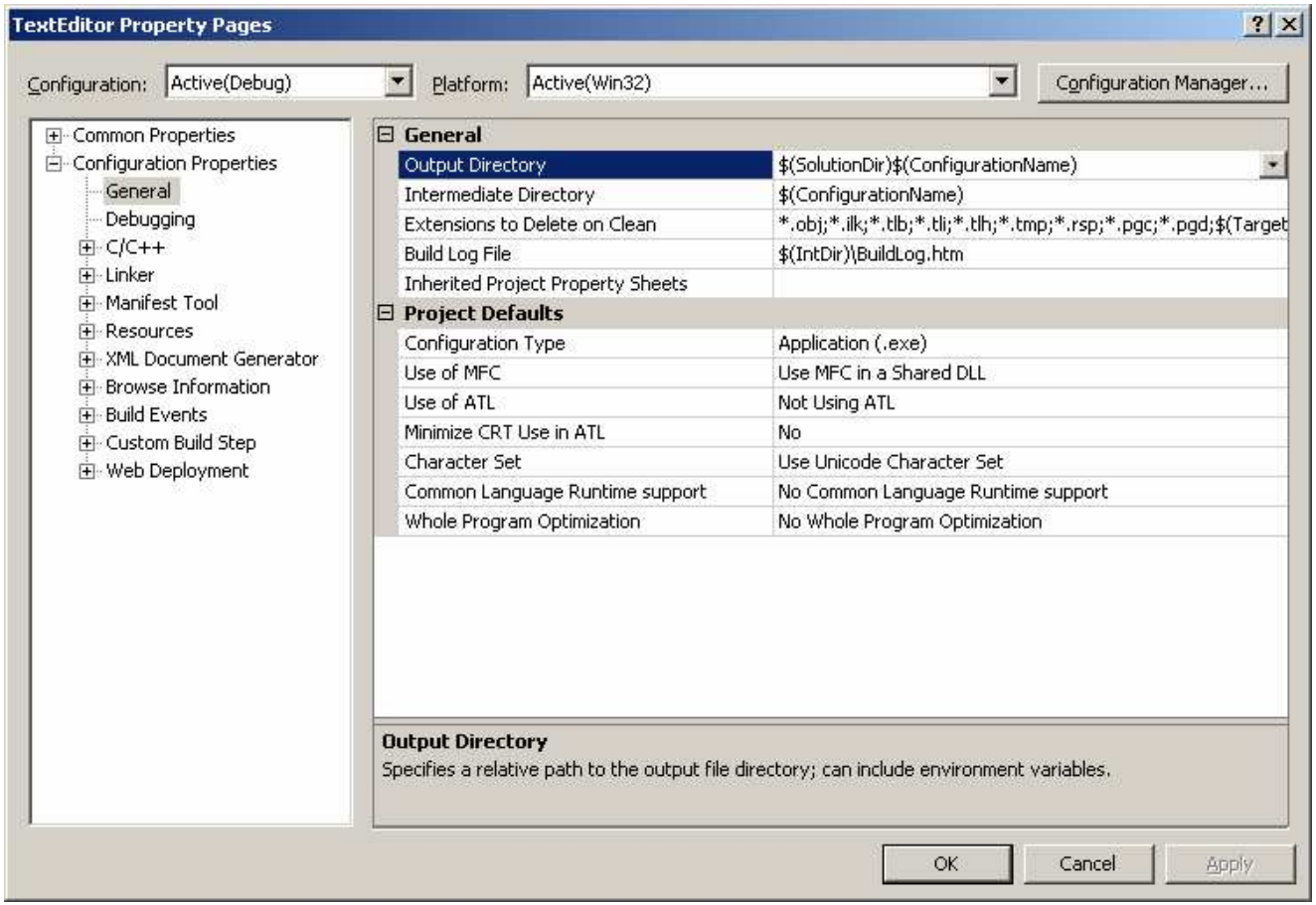
Code Generated by the MFC Application Wizard

- All the files are stored in the `TextEditor` project folder
  - which is a sub-folder to the solution folder with the same name.
- Class definitions are in `.h` files.
- Resource files are in the `res` sub-folder to the project folder.
- Member functions are defined in `.cpp` files.

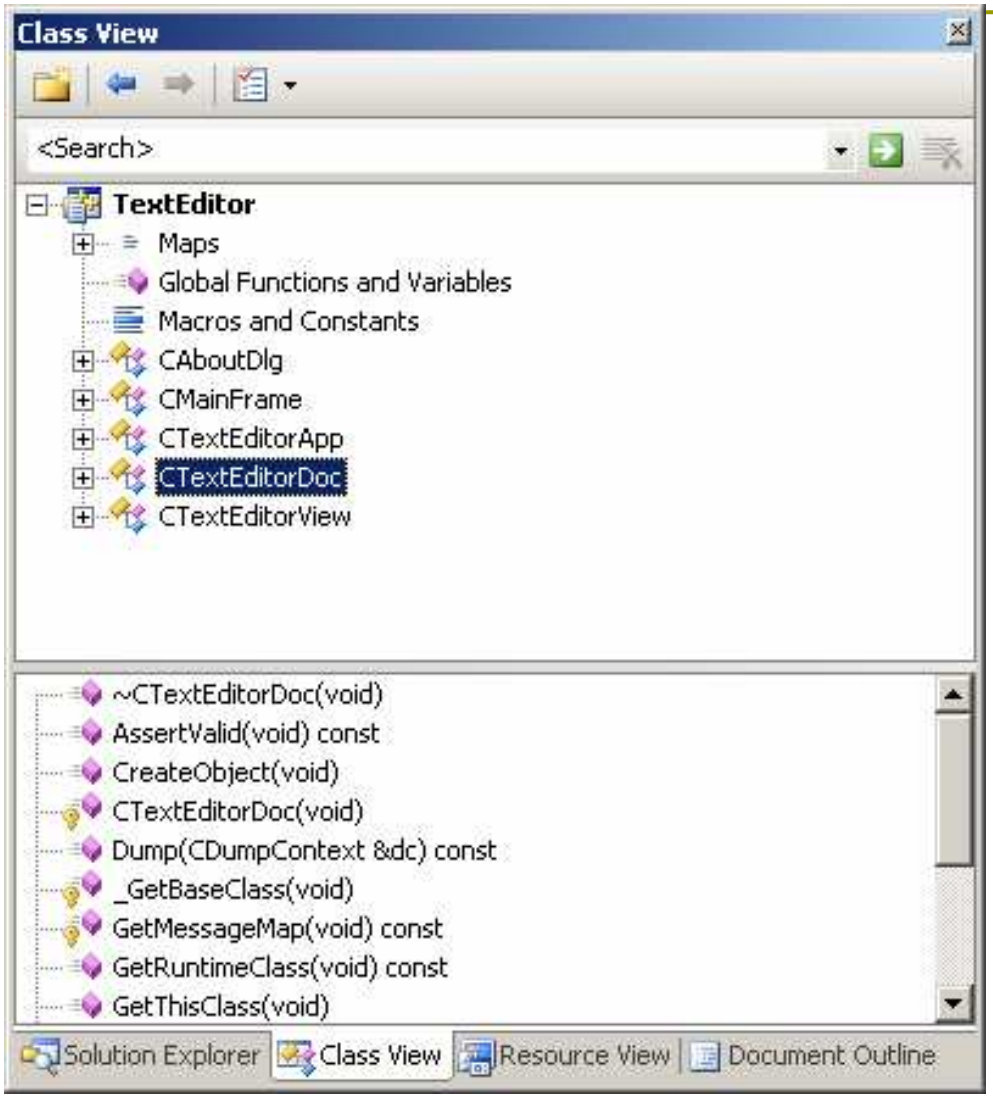


Project Property

- Right-click TextEditor project in the Solution Explorer pane, and select Property from the pop-up menu:

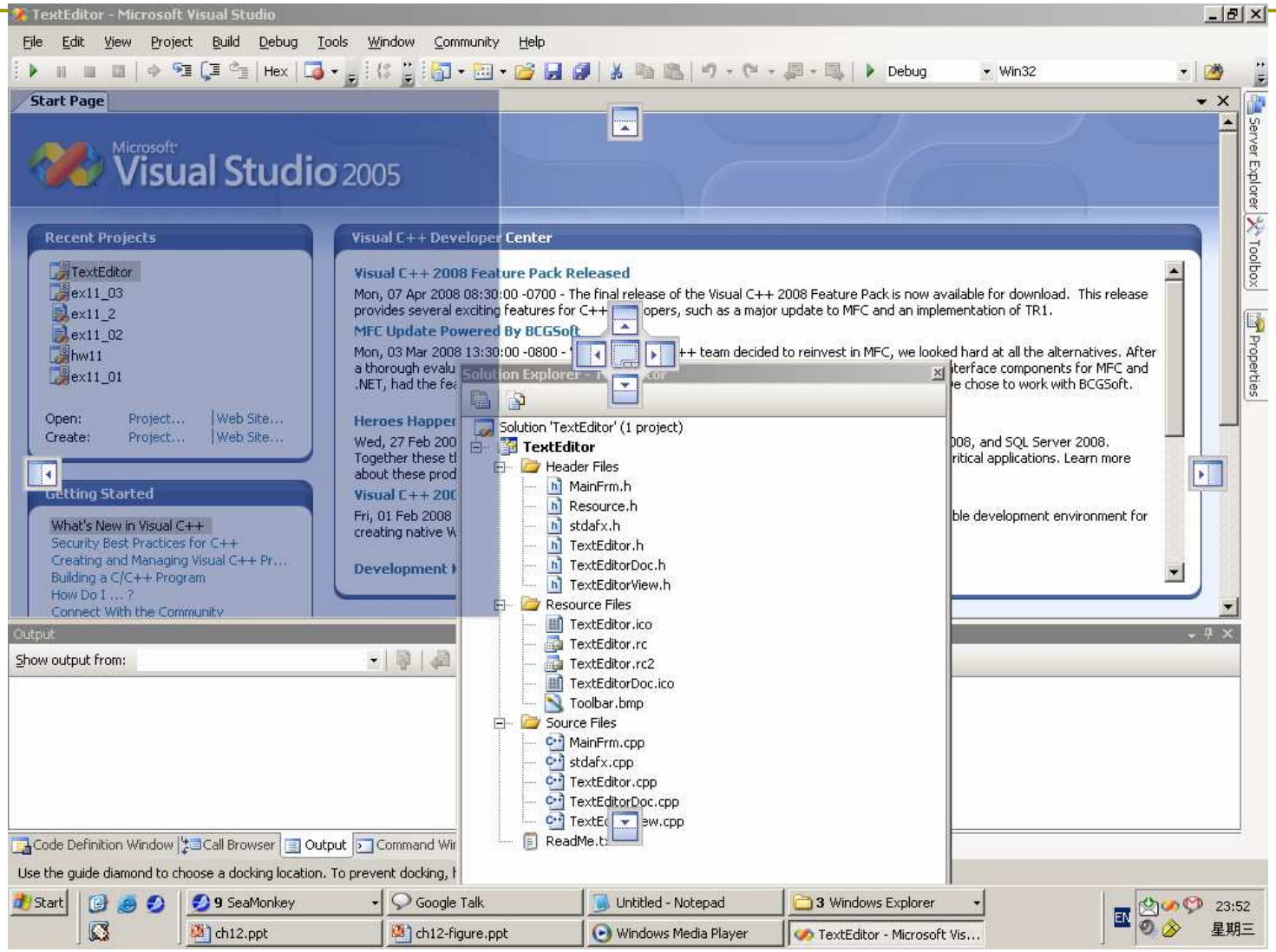


Viewing Classes



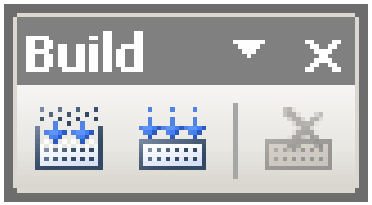
- You see four basic classes:
  - CMainFrame
  - CTextEditorApp
  - CTextEditorDoc
  - CTextEditorView
- Global Functions and Variables contains two definitions:
  - theApp – the application object
  - indicators – an array of indicators recording the status of caps lock, num lock and scroll lock.

Floating/Dockable Solution Explorer



Creating an Executable Module

- To compile and link the program
  - Build > Build Solution
  - Ctrl + Shift + B
  - <F7>
  - Click the Build icon in the boolbar



Precompiled Header Files

- The first time you compile and link a program, it will take some time.
- The second and subsequence times it should be faster.

- A feature of Visual C++ 2005 called precompiled headers will save the output from compiling header files in a special file with the extension `.pch`.
- On subsequent builds, this file is reused if the source in the headers has not changed, thus saving the compilation time for the headers.

Running the Program

- Ctrl + F5
- This is a fully functioning, simple text editor.
  - All the items under all menus are fully operational
    - Save / Open files
    - Cut / Paste text
    - Print
    - Toolbar
    - Tool tip
    - System menu

Exercise

- Generate the text editor application several times, trying different window styles from the User Interface Features in Application wizard.