© 2014 IJIRT | Volume 1 Issue 5 | ISSN : 2349-6002

MICROSOFT FOUNDATION CLASSES

Priya Kumari, Neha Agrawal
Student, Department Of Information Technology
Dronacharya College Of Engg.

***Abstract-* In this paper, the Windows OS, its fundamental concepts,such as windows, user inputs, and messages, and its interaction with an MFC application are described.The *Microsoft foundation classes*(*MFC*) *Library* is a collection of C++ classes.MFC is used with the *Microsoft Windows operating system* (Windows OS,also Windows). Microsoft foundation classes are highly integrated with Visual Studio to speed development. The MFC base classes are contained in a C++ class library developed by Microsoft, which is now supplied with many C++ compilers; it is typically stored in the directory C:\MSVC\MFC. It is provided as a *Dynamic Link Library* (DLL) so your application has access to the classes in MFC. MFC provides easier-to-use functionality that incorporates the API functions. The use of Microsoft foundation classes means that much of the program is done and we need only to add some special features to MFC code to create our application.**

I. INTRODUCTION

MFC is designed to work with all the available Windows OSs. Today the windows OSs available in the marketplace.The relatively windows 95 (Win95) and Windows NT (NT), and their predecessor of long standing, Windows 3.1*x* (Win3.1x). MFC applications can be built and run on any of these operating systems.An MFC application built on Win3.1*x* can be run on Win3.1*x*, Win95, and NT.An MFC application built on Win95 or NT can be run on either Win95 or NT,but not on Win3.1*x* unless it can be recompiled on Win3.1*x*.The MFC classes were designed for compatibility between Win95, Win3.1*x*,and NT. Therefore, the classes do not support the unique programming featuresof NT. The Microsoft Foundation Class Library defines the **application framework**. MFC provides a variety of classes designed to serve a wide range of needs.

The majority of MFC classes are derived, either directly or indirectly, from CObject. CObject provides other useful benefits to its derived classes as well.

II. WINDOWS OPERATING SYSTEM AND MFC

MFC is designed to work with all the available Windows OSs. There are three Windows OSs available in the marketplace today: the relatively new Windows95 (Win95) and Windows NT (NT), and their predecessor of long standing,Windows 3.1*x* (Win3.1x). MFC applications can be built and run on any of these os.

III. C++ COMPILERS AND MFC

C++ compilers can build MFCapplications which are Microsoft, Symantec, and Borland. The Microsoft compiler, known as Visual C++, is the most prevalent of the compilers that host MFC. Visual C++ 1.5 is for the Win3.1x operating system.Visual C++4 is for Win32 systems. Each of these compilers provides tools to help the programmer through difficult chores. The tools are known as AppStudio, ClassWizard, and AppWizard. The Symantec C++7 compiler also provides tools to assist the programmer. The Borland C++ 5 compiler provides MFC compilation support and the Resource Workshop can be used.

IV. MFC AND WINDOWS OS INTERACTION

Windows OS has three major components: *User, Graphics Device Interface*
(GDI), and *Kernel*. User is a module of code that services input devices, such
as keyboards. GDI is a module that services output to graphic devices—screens, printers, etc. Kernel services file management and internal memory management. These three components are called the API. These components interact with the MFC application. An MFC application calls functions in the API. The base classes in the MFC library incorporate API functions, so the MFC classes also

call functions in the API. The three API components are each provided as DLLs.

V. THE STRUCTURE OF AN MFC APPLICATION

An MFC application does two basic things.The first is that creates the application’s own main window and performs any initial activities, such as allocating memory and second is processes messages to that window, or application.

When writing a program, the first step is to define and create the application’s main window, which is the “real estate” or piece of the screen that the application will control. Programs only write inside their own windows and

not in other program windows. If a program tries to write outside of its area,nothing will show up on the screen. This is how the operating system keeps programs from interfering with each other on the screen. Restricting output to the program’s window is the key to having several programs co-exist using the same screen, or display area.A program’s main window is also referred to as the mainframe. Main windows display the menu bar, if there is one. The main window, or mainframe, can be the parent to child windows, which include ordinary windows, dialog boxes, and controls.

VI. CREATING A MAIN WINDOW USING MFC

The first program creates a main window with standard features, which are: standard non-client visual elements, standard white background color, standard arrow cursor shape, and standard icon. It contains the minimal amount of code required for an MFC program. It presents the entire C++ source code needed in your MFC program to create a resizable main window that has a working system menu and can be minimized, maximized, restored, and moved on the screen.

VII. CONCLUSION

MFC provides an efficient way to build programs for Windows OSs. MFC is designed to work with all the available Window OSs. Most C++ compilers on the market today can build MFC applications. The central concept of a Windows OS application is its main window which displays the output of the application. The main window has a menu and can be designed with varying features. The user interacts with the application through keyboard presses and mouse clicks. The Windows OS sends messages to the application’s windows as a result of user inputs or other events. The MFC framework intercepts these messages, reads and interprets them, and forwards the relevant data as inputs to the application’s handler functions. Using the MFC library allows you to create powerful programs with minimal coding. Our classes are derived from the MFC classes and inherit most of the functionality that you need. An application that creates a fully functional main window with standard features contains less than 30 lines of code. It contains .an application class derived from CWinApp, from which it inherits all the functionality required to set up and run the message loop. It also contains a mainframe class derived from CFrameWnd, from which it inherits standard mainframe behaviour.

REFRENCES

[1] solomon.ipv6.club.tw/**Course**/C_Programming.

[2] **research**.**microsoft**.com/en-us/**project**s/pex/pexsharepoint

[3] "Visual Studio Express Edition FAQ". Microsoft.com. Retrieved 6 January 2012.

[4] "Microsoft Buys Into Inprise, Settles Disputes". Techweb.com. Retrieved 6 January 2012.

[5] Williams, Mickey; David Bennett. "Creating Your Own Message Maps". Inform IT.

[6] "Visual C++ 2008 Feature Pack shipped". Blogs.msdn.com. Retrieved 26 April 2008.

[7] "Quick Tour of New MFC functionality". Blogs.msdn.com. Retrieved 16 November 2007.

[8] "MFC Update Powered By bcgsoft". Msdn2.microsoft.com. Retrieved 16 November 2007.

[9] "Visual C++ 2008 Feature Pack Release Download Page". Microsoft.com. Retrieved 16 May 2008.

[10] "Microsoft Security Bulletin MS11-025 - Important : Vulnerability in Microsoft Foundation Class (MFC) Library Could Allow Remote Code Execution (2500212)". Microsoft.com. Retrieved 2012-11-19.