Microsoft Visual Studio .NET: An Integrated Windows Program Development Environment

Using Microsoft Visual Studio .NET

- Self-contained environment for Windows program development:
  - creating
  - compiling
  - linking
  - testing/debugging
- IDE that accompanies Visual C++, Visual Basic, Visual C#, and other Microsoft Windows programming languages
- See Chapter 2 and Appendix D of Deitel text
- Also Appendix C of Gregory text

Visual Studio Capabilities

- Generate starter applications without writing code
- View a programming project in many different ways
- Edit source and include files
- Build the application's user interface visually
- Compile and link
- Debug an application while it runs
- Obtain online help
- Lots of others

Some Visual Studio Components

- **The Editors**

  C, C++, C#, VB source program text editors
  - cut/paste, color cues, indentation,
  - generates text files

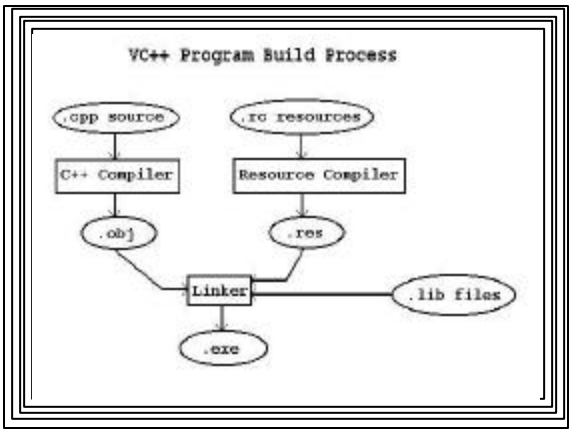
  Resource Editors
  - icons, bitmaps, cursors, menus, dialog boxes, etc.
  - graphical, WYSIWYG, Integrated
  - generates resource script (.rc) files
  - integrated with text editor
  - Done visually

VC++ Unmanaged Code Compiler

- **C/C++ Compiler**
  - translates source programs to machine language
  - detects and reports errors
  - generates object (.obj) files for linker
- **Resource Compiler**
  - Reads .rc file
  - Generates binary resource (.res) file for linker

The Linker

- Reads compiler .obj/.res files
- Accesses C/C++/Windows libraries
- Generates executable (.exe or .dll)



The Debugger

- powerful source code debugger
- integrated with all parts of Visual Studio
- Features
  - breakpoints
  - tracing through/over functions
  - variable watch windows
  - much more

The Wizards

- **AppWizard**
  - Windows code generator for MFC apps
  - automatically creates working program templates & skeleton code
- **ClassWizard**
  - facilitates easy extension of AppWizard-generated classes
  - creation of new classes and response functions
  - used to tailor AppWizard-generated MFC & .NET skeletons
  - Replaced by **Properties Window** in .NET

Help

- Hover over key words in edit window and a one-line help message appears
- More detailed help can be obtained by:
  - 'Start Page' – 'Search Online'
    - To access the MSDN Online Library
- Easier to use the 'Help' Menu Item
  - 'Dynamic Help' – context sensitive
    - Click on text in edit window and corresponding topic appears in help window
    - Click on topic in help window to get help
  - 'Contents': Select a topic
  - 'Search': Enter a topic
  - 'Index': Enter a topic

Win32 API Online Help

- 'Help' – 'Contents'
  - Filtered by: (no filter)

MSDN Library
- Windows Development
  - Win32 API
    - SDK Documentation
      - Reference
        - Functions by category
        - Functions in alphabetical order

MFC Online Help

- 'Help' – 'Contents'
  - Filtered by: (Visual C++)

Visual Studio .NET
- Visual C++
  - Visual C++ Reference
    - Visual C++ Libraries
      - MFC Reference
        - Hierarchy Chart
        - MFC Classes
        - Class Library Overview
        - MFC Classes (!!)
        - etc.

MSDN Library (on Web)

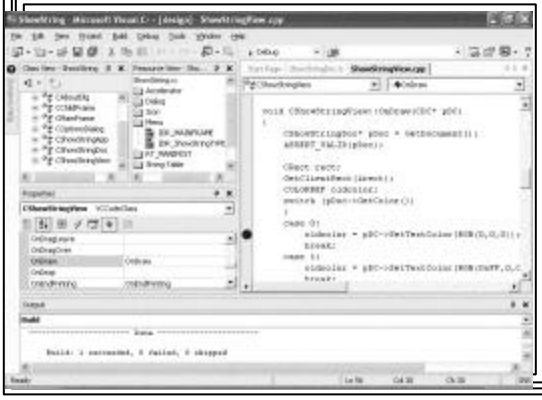
- **Go to:** http://msdn.microsoft.com
  - **Search MSDN for desired topic**
  - Good URL for MFC:
    http://msdn.microsoft.com/library/default.asp?url=/library/en-us/vcmfc98/html/_mfc_class_library_reference_introduction.asp

Using Visual Studio .NET

- To prepare many kinds of applications
  - Win32 Console Applications (DOS programs)
  - Win32 API Apps in C or VC++
  - MFC Apps in VC++
  - DLLs
  - .NET Windows Forms Apps in Managed C#, VB, C++, and other languages
  - ASP.NET Web Apps and Services
  - ADO.NET Data Base Apps
  - Others

Visual Studio Layout

- Menu bar
- Several tool bars
- View Windows (to the side)
  - Solution Explorer
  - Class View
  - Resource View
  - Properties Window
- Working Area (main window)
  - Text Editor to enter/modify source code
  - Resource Editors
- Output Window & Status Bar (bottom).
  - System messages (errors)
- Windows can be moved around, docked and undocked



Toolbars

- Contain icons --instant routes to main menu functions
- Many of them
- May not be visible
- If not, right click on any visible toolbar
- Brings up following popup window
- Can activate a toolbar by clicking on its check box

Keyboard Shortcuts

- All menu/toolbar selections are available from the keyboard using key combinations
- Can be faster
- More information in Online Help
  - 'Index' | 'Keyboard Shortcuts' | 'Shortcut Keys'

Solutions and Projects

- Solution
  - A single application
  - Can contain one or more projects
    - In Managed applications, can be in different languages
  - Overall solution information stored in a .SLN file
  - Open this when you want to work on a solution
- Project
  - Collection of files:
    - Source, headers, resources, settings, configuration information

Important Visual Studio Generated Files

`.sln`	Solution
`.vcproj`	Project
`.c, .cpp, .cs`	C/C++/C# Windows App source code
`.h`	C/C++ header
`.rc`	Resource script
`.res`	Compiled resource
`.ico`	Icon
`.bmp`	Bitmap image
`.exe`	Executable program
`.dll`	Dynamic Link Library (if used)
`.aspx`	ASP.NET Web Form source code
`.asmx`	ASP.NET Web Service source code

Temporary Visual Studio generated files

- **Many are very big and can (should) be removed!**

`.obj`	Compiler machine code translation
`.ilk`	Incremental link file
`.pch`	Precompiled header (huge!!!)
`.pdb`	Precompiled debugging info
`.idb`	Incremental debug info
`.ncb`	Supports viewing classes
`.aps`	Supports viewing resources
others	

Program Configurations

- Debug
  - appends debugging information
  - produces more and larger files
- Release
  - no debugging information
  - optimized for size & efficiency

Setting the Configuration

- Click 'Build' on Main Menu
- Choose 'Configuration Manager'
- Choose desired configuration ('Debug' or 'Release') in 'Active Configuration Box'
- Default is 'Debug'

Create a Win32 Application with Visual Studio

- **Startup**
  - click 'Start' on Task Bar – 'All Programs'
  - 'Microsoft Visual Studio .NET 2003' | 'Microsoft Visual Studio .NET 2003'
- **Creating a new solution**
  - 'File' | 'New' | 'Project' from Menu Bar
  - In 'New Project' box, select 'Visual C++ Project' from 'Project Types:' & click on 'Win32 Project' in 'Templates'
  - Set the 'Location' to a convenient directory & name the project (e.g. win32app1)
  - 'OK'

- **Click 'Application Settings' in resulting 'Application Wizard' Box**
  - Select 'Windows Application' from 'Application Type' radio buttons
  - Select 'Empty Project' from 'Additional Options' check boxes
  - Click 'Finish'
- Click 'OK' in **'New Project Information' Window**

- **Inserting source files into project:**
  - Open a new C++ file & type or copy/paste the code into the program:
    - 'File' | 'New' | 'File' from menu
    - Choose 'Visual C++' from 'Categories', C++ file (.cpp) from 'Templates', & click 'Open'
    - Type or paste source code into the resulting Edit window
    - Save the file in the project's directory as a C++ source file, giving it an appropriate name (e.g., win32app1)
  - Add the source file to the project:
    - Choose 'Project' | 'Add Existing Item' from menu
    - Click on the file you saved (e.g. win32app1.cpp)
    - Confirm that it was added to the project by expanding 'Source Files' in the Solution Explorer Window
      - If Solution Explorer is not visible, select 'View – Solution Explorer' from the menu

- **Alternative Way of Adding a Source File to a Project:**
  - You can also copy an existing source code file into the project's directory
  - Then as before:
    - Choose 'Project' | 'Add Existing Item' from the menu
    - Select the .cpp file & click 'Open'
      - Should appear in Solution Explorer window
      - Open it by double clicking on it

- **Building the Solution:**
  - 'Build' | 'Build Solution' from menu
  - Project will be compiled/linked
  - Messages/errors will appear in Output Window
- **Running the Program:**
  - 'Debug' | 'Start' from menu
    - Shortcut key: F5
  - Or 'Debug' | 'Start Without Debugging' from menu
    - Shortcut key: Ctrl-F5
    - or click exclamation point

Compiling from Command Line

- Command Line Compilers:
  - C++: CL.EXE
  - C# : CSC.EXE
  - Visual Basic: VBC.EXE
- All are run from a DOS session, but directory paths must be set correctly
- Easiest to start a Visual Studio .NET Command Prompt (paths already set)
  - From Task Bar:
    - Start | All Programs | Microsoft Visual Studio 2003 | Visual Studio .NET Tools | Visual Studio .NET Command Prompt (2003 version)
    - Start | All Programs | Microsoft Visual Studio .NET | Visual Studio .NET Tools, Visual Studio .NET Command Prompt (2002 version)

Command Line Compiling, continued

- To compile our first Visual C++, Win32 API application from the command line:
  - cl kernel32.lib user32.lib gdi32.lib win32a1.cpp
  - Note that any required libraries (DLLs) must be specified
- There are many compiler options:
  - See Online Help:
  - 'Index' | 'cl.exe compiler' | 'Compiler Options (C++)'
  - For C#: 'Index' | 'csc.exe'

- **Cleanup:**
  - Copy solution, project, source, header, resource files to disk
  - Copy .exe file from project's Debug directory
  - Best: Delete all temporary files & copy entire solution (project directory) to floppy
  - Delete project directory from hard drive
- **Exiting Developer Studio:**
  - 'File' | 'Exit' from menu