COM Programming

Course Summary Table

Duration:	5 Days
Target Audience:	Experienced windows developers, interested in understanding and writing COM components and clients in C++.
Objectives:	• Understand the why of COM • Build COM servers and clients • Use the Active Template Library (ATL) effectively • Use COM features such as in process and out of process servers, automation, events and callbacks • Understand and use COM apartments and threading models
Pre Requisites:	• At least one year of experience working with the Windows API • At least 2 years of experience working with of C++ • Basic understanding of Windows OS concepts such as processes, threads, virtual memory and DLLs • C++ 11/14/17 knowledge is not required

Instructor: **Pavel Yosifovich**

Abstract

The Component Object Model (COM) provides an abstraction and supporting runtime for creating component-based systems, leveraging loose coupling and independence of programming language and environments. Many existing Windows components are exposed through COM (as opposed to flat C interface). COM also forms the basis of the Windows Runtime, Microsoft's latest API and runtime targeting (mostly) UWP applications.

The course deals with all the foundations of COM, from interfaces and servers, to marshalling and automation. Lab exercises ground the theoretical material in practical work. Students will gain understanding of the whys of COM, not just the how. Tips and general advise are given throughout the course from years of COM experience by the instructor.

Syllabus

- Module 1: From C++ to COM
    - Component Based Development
    - Problems with C++ and traditional class libraries
    - Interfaces
    - Virtual function tables and virtual dispatch
    - Runtime discovery
    - Removing compiler dependencies
    - Lifetime management
    - Summary

- Module 2: COM Fundamentals
    - What is a COM object?
    - The IUnknown interface
    - HRESULTs
    - COM Servers
    - Activation
    - Class Factories
    - Multiple Interfaces
    - Registration
    - Implementing COM Component in C++
    - COM Clients
    - Summary
    - Lab: creating COM server and client with C++

- Module 3: Introduction to IDL
    - What is the Interface Definition Language?
    - Key Features
    - Basic Syntax
    - Attributes
    - Defining Interfaces and Classes
    - Using the MIDL Compiler
    - Type Libraries
    - Summary

- Module 4: Building COM Components with ATL
    - What is the Active Template Library?
    - Using the ATL Object Wizard
    - Implementing Interfaces
    - Smart Pointers
    - Understanding ATL Generated Code
    - Containment and Aggregation
    - Using Aggregation
    - Lab: Building a COM server and client with ATL; building a more complex component; using non-C++ clients

- Module 5: Automation
    - What is Automation?
    - The IDispatch Interface
    - Dual Interfaces
    - ATL Support for Automation
    - Automation Types
    - Strings and Variants
    - Lab: Building Automation Servers and Clients

- Module 6: Out of Process Servers

    - Why Out of Process?
    - Proxies and Stubs
    - Marshalling
    - Memory Management
    - Building EXE servers
    - COM Security model
    - Impersonation
    - Introduction to Distributed COM
    - Lab: building out of process server

- Module 7: Threads and Apartments
    - Quick recap: Processes and Threads
    - COM and Threading
    - COM Apartments
    - Process and Apartments
    - Apartments and the Registry
    - Apartments and Marshalling
    - The Global Interface Table (GIT)
    - The Free Threaded Marshalar (FTM)
    - EXE Servers and Apartments
    - Labs: Multi-threaded server; testing apartment variations

- Module 8: Events and Callbacks
    - Incoming vs. Outgoing Interfaces
    - Connectable Objects
    - Connection Points
    - ATL Support for Connection Points
    - Other ways to Manage Events and Sinks
    - Lab: exposing events from a COM component; client connection to server