

Exam 70-016: Designing and Implementing Desktop Applications with Microsoft Visual C++ 6.0

Contents:

Unit 1: Visual C++ 6.0: IDE Interface

- Identify the features of Visual C++ 6.0.
- Install Visual C++ 6.0.
- Identify the function of a Visual C++ 6.0 tool.
- Match the Visual C++ 6.0 wizard with their uses.
- Match the Visual C++ 6.0 resource editors with their uses.
- Identify the IDE interface components according to their functions.
- Identify the menu items of Visual C++ according to their functions.
- Identify the toolbar buttons of Visual C++ according to their functions.
- Create a new project by using the MFC AppWizard (exe).
- Match the different types of Visual C++ project files with their contents.
- Add a shortcut key to a command that does not have a default shortcut key.
- Edit the shortcut key for a specific command.
- Display a specific toolbar.
- Add a specific toolbar button to a toolbar.
- Group specific buttons on a toolbar.
- Add an additional tool to the IDE.
- Add a macro to the IDE.

Unit 2: SQL Server 6.5 and MSMQ

Register SQL Server 6.5 by using the SQL Enterprise Manager.

- Configure SQL Server 6.5 by using the Microsoft SQL Enterprise Manager.
- Configure a SQL Server database by using the SQL Enterprise Manager.
- Set permissions for a specific user by using the By User tab.
- Install Microsoft Message Queue Server (MSMQ).
- Create a queue by using the Microsoft Message Queue Explorer.
- Configure an enterprise by using the Microsoft Message Queue Explorer.
- Configure a site by using the Microsoft Message Queue Explorer.
- Configure a queue by using the Microsoft Message Queue Explorer.

Unit 3: MTS and Application Types

- Identify the limitations of a two-tier application.
- Identify the advantages of a three-tier application.
- Identify the features of Microsoft Transaction Server (MTS).
- Identify the ideal situation for installing MTS according to a given set of requirements.
- Install MTS 2.0 on a Windows NT server.
- Identify the uses of the MTS environment elements.
- Identify the MTS Explorer interface items that will be used in a given situation.
- Sequence the steps that are performed by a client application executable to set up a client for using MTS components.

Unit 4: VSS and Source-Code Control

- Install a Visual SourceSafe (VSS) 6.0 Server.
- Set a password for the default Admin user account by using the VSS Administrator application.
- Create a user account by using the VSS Administrator application.
- Grant specific rights to all the VSS database users by using the VSS Administrator application.
- Grant rights to specific users of a project by using the VSS Administrator application.
- Install the VSS Client On Windows NT.
- Identify the steps to establish the source code control environment.
- Place a project under source code control.
- Check a file out by using the appropriate menu item.
- Compare the changes made to a local file copy with its associated master copy.
- Check a file in by merging the changes made to the file.
- Check a file in without accepting changes made to the file.
- Remove a file from source-code control.

Unit 5: Microsoft Foundation Classes

Identify the features of MFC.

- Identify the benefits of MFC.
- Match the key classes in the MFC class library with their functions.
- Identify the platform-specific characteristics to be considered when developing MFC applications for various platforms.
- Identify the distinguishing features of MFC and SDK.
- Identify the characteristics of a DLL.
- Identify the advantages of dynamic libraries over static libraries.

Confidential

- Identify the features of different types of linking.
- Select the appropriate situation in which you would use a specific DLL.

Unit 6: Document View Architecture

- Identify the features of the Document View architecture.
- Identify the functions of the key classes in Document View architecture.
- Create a minimal MFC program by using the Appwizard.
- Identify the characteristics of CDocument class.
- Identify the characteristics of CView class.
- Create a Document View application by using the Appwizard.
- Create a scrolling view by using the Appwizard.
- Create a splitter window by using the Appwizard.
- Match the drawing classes with their functions.
- Draw a line by using the OnDraw() function.
- Identify the printer information provided by the Windows functions.
- Identify the tasks performed by the CView class to print a document.
- Match the overridable member functions of the CView class with the reasons for overriding them.
- Match the members of the CPrintinfo class with their uses.
- Sequence the steps to print a document using the MFC functions.
- Match the members that contribute to modify the print preview with their functions.
- Enhance print preview by setting the number of pages to be displayed in the preview mode.
- Sequence the steps to manage printing using the CPrintDialog class.

Unit 7: Introduction to Component Design

Identify the benefits of using components.

- Identify the interface design rules.
- Identify the interface design guidelines.
- Identify the features of COM components.
- Identify the benefits of using COM components.
- Identify the features of ActiveX controls.
- Identify the benefits of using ActiveX controls.

Unit 8: Determining Data Access Requirements

- Evaluate whether a database should be incorporated in an application.
- Identify the benefits of using data access model within an application.
- Match the library to be used for accessing data with the given situation.
- Identify the benefits of using DAO.
- Identify the benefits of using ODBC.

Miracle Corporate Solutions Pvt. Ltd., C-41,GF, Behind Nirula's Hotel, Sec-2, Noida
Call : 0120-3058446/7
Mobile : 9311305846/47

Confidential

- Identify the benefits of using OLE DB.
- Identify the benefits of using RDO.
- Identify the benefits of using ADO.
- Select the most appropriate data access model to access data from the database in a given situation.

Unit 9: Extending Functionality

- Create a new class by using the ClassWizard.
- Add a member variable to a control.
- Add a member function for a class.
- Add a message handler for a control.
- Sequence the steps to route command messages between a user interface object and a handler function.
- Identify the benefits of subclassing.
- Incorporate existing code by using Components and Controls Gallery.
- Identify the steps to add a scriptlet to an application.
- Insert DHTML controls into an MFC application.

Unit 10: Menus, Dialog Boxes & Property Sheets

- Create a static menu.
- Create a context menu.
- Create a dynamic menu at run time.
- Create an accelerator for a given menu item by using the Resource Editor.
- Identify the characteristics of modal and modeless dialog boxes.
- Create a dialog box template by using the Dialog Editor.
- Associate a dialog box template with a class.
- Add member variables to the dialog box controls by using the ClassWizard.
- Write the code to display a dialog box at run time.
- Sequence the steps to exchange and validate the user input from a dialog box.
- Identify the features of the CFormView class.
- Create a form view with controls by using the CFormView class.
- Identify the characteristics of property sheets and property pages.
- Create a property page template by using the Dialog Editor.
- Associate the resources of the property sheet with classes.
- Create a property sheet class by using the ClassWizard.
- Write the code to display the property sheet at run time.

Unit 11: Adding & Editing User Interface Controls

Match the user interface controls with their descriptions.

- Create user interface controls by using the Resource Editor.
- Create a toolbar button in an MFC application by using the Resource Editor.

Miracle Corporate Solutions Pvt. Ltd., C-41,GF, Behind Nirula's Hotel, Sec-2, Noida
Call : 0120-3058446/7
Mobile : 9311305846/47

Confidential

- Add tooltips to a toolbar by using the Resource Editor.
- Identify the steps to create a status bar in an MFC application.
- Identify the steps for creating a rebar in an MFC application.
- Create an image list.
- Identify the steps to create a list view control.
- Identify the steps to create a tree view control.

Unit 12: Online User Assistance

- Match the different types of help that you can provide in an MFC application with their purposes.
- Match the different help files in an MFC application with their purposes.
- Add ToolTips to an interface control by using the Resource Editor.
- Create an HTML help file for an application by using HTML Help Workshop.
- Add HTML help support files in an MFC application.
- Add context-sensitive help to an MFC application.
- Add a link to a Web-based help file from an HTML help file.

Unit 13: Exception Handling

- Identify the need to use exception handling in an MFC application.
- Identify the features of the different exception handling mechanisms that are supported by Visual C++.
- Match the statements in the C++ exception handling syntax with their purposes.
- Identify the steps in the exception handling process.
- Identify the advantages of using exception handling over using return codes.
- Match the functions of the CException class with their purposes.
- Identify the derived classes to be used for handling the errors generated in specific situations.
- Complete the code to implement C++ exception handling in an MFC application.

Unit 14: Implementing Asynchronous Processing

- Identify the differences between threads and processes.
- Identify the features of threads implemented by MFC.
- Identify the classes and functions used by MFC to implement threading in an application.
- Create a worker thread by using AppWizard.
- Identify the rules to suspend and resume threads.
- Identify the uses of the Sleep() API function.
- Identify the functions used in the different thread termination situations.
- Identify the steps to implement the critical sections thread synchronization object.
- Identify the steps to implement the mutexes thread synchronization object.
- Identify the steps to implement the semaphore thread synchronization object.

Confidential

Unit 15: Creating Dynamic Content

- Identify the functions that are used in creating dynamic user interfaces by performing read/write to the Registry.
- Modify an existing MFC application to store and retrieve personalized user settings from the registry.
- Identify the features of ISAPI DLLs.
- Create an ISAPI server extension DLL.

Unit 16: Incorporating COM and ActiveX

- Identify the features of COM components.
- Identify the benefits of using COM components.
- Identify the features of ActiveX controls.
- Identify the benefits of using ActiveX controls.
- Insert a ActiveX control into a project.
- Handle an event generated by an ActiveX control.
- Dynamically create an ActiveX control.
- Download an ActiveX control from the Internet.
- Implement a COM component.

Unit 17: Accessing System Data

- Match the MFC classes involved in Serialization with their features.
- Display data by using the CArchive class.
- Open an existing file by using the Open member function of the CFile class.
- Retrieve data from a file by using the Read member function of the CFile class.
- Store data in a file by using the Write member function of the CFile class.
- Identify the Seek function that will be used in a given situation.
- Match the registry key with the information that they contain.
- Set a registry key by using the SetRegistryKey member function.
- Retrieve a value from a registry key by using CWinApp member functions.
- Store information in a registry key by using CWinApp functions.

Unit 18: Remote Communication

- Match the member functions of the CSocket class with their uses.
- Create a socket for a server and a client.
- Set up a socket to listen for a connection.
- Establish a connection to a socket.
- Read incoming data from a given connected socket.
- Write outgoing data to the connected socket.
- Write the statement to open a COM port for overlapped I/O.
- Write the statement to close a COM port.
- Write the statement to write data to a COM port for overlapped I/O.

Miracle Corporate Solutions Pvt. Ltd., C-41,GF, Behind Nirula's Hotel, Sec-2, Noida
Call : 0120-3058446/7
Mobile : 9311305846/47

Confidential

- Write the statement to read data from a COM port for overlapped I/O.

Unit 19: Database Access: ODBC and DAO

- Configure an Open Database Connectivity (ODBC) database to a system.
- Identify the sequence in which MFC ODBC functions are called to complete a given task.
- Create an ODBC database application.
- Create a display for the database application.
- Modify an ODBC database application to manipulate the database.
- Modify an ODBC database application to filter the records that satisfy the criteria entered by the user.
- Identify the advantages of using DAO to connect to a database.
- Match the Data Access Objects (DAO) database classes supported by MFC with their functions.
- Create a DAO database application.
- Modify a DAO database application to manipulate the database.
- Modify the DAO application to filter the records that satisfy the criteria entered by the user.
- Match the data members of database exception classes with their functions.
- Sequence the steps that occur in the database exception-handling process.

Unit 20: Database Access: ADO and RDO

- Identify the advantages of using ADO to access a database.
- Create an ADO database application by using data-bound dialog wizard.
- Modify a ADO database application to manipulate the database.
- Modify the ADO application to filter the records that satisfy the criteria entered by the user.
- Match the ADO error codes with the situations in which errors occur.
- Identify the benefits of using Remote Data Objects (RDO) to access a database.
- Create an RDO database application using Wizards.
- Modify the RDO application to filter the records that satisfy the criteria entered by the user.

Unit 21: ActiveX Controls

- Identify the features of an ActiveX control.
- Identify the advantages of creating ActiveX controls.
- Sequence the steps for creating an ActiveX control.
- Create ActiveX controls by using Microsoft Foundation Classes (MFC).
- Create an ActiveX control by using the ActiveX Template Library (ATL).
- Add a property and a method to an ActiveX control.
- Add a stock event to an ActiveX control.

Confidential

Unit 22: Component Object Model

- Identify the benefits of COM.
- Identify the features of a COM interface.
- Identify the IUnknown function that is called in a given situation.
- Match the IDispatch functions names with the functions they perform.
- Identify the contents of a dual interface.
- Create a COM component by using Software Development Kit (SDK).
- Create a COM component by using MFC.
- Create a COM component by using ATL.
- Select the type of a COM server for a given set of requirements.
- Create a COM server by using ATL.
- Create a COM client to access an ATL COM server.

Unit 23: Advanced Component Implementation

- Sequence the steps performed by COM to implement aggregation.
- Sequence the steps performed by COM to implement containment.
- Enable the library debug support to debug a COM component.
- Display specific information about an application by using the Spy++ utility.
- Sequence the steps to implement error-handling in a COM component.
- Sign a COM component by using the SIGNCODE utility.
- Create an Active Document by using the MFC AppWizard.
- Modify an active document application to support active document containment.
- Configure the default security settings for a component by using the Distributed COM Configuration utility.
- Configure component security by using the Distributed COM Configuration utility.
- Identify the methods used for registering a DCOM DL.

Unit 24: Microsoft Transaction Server

- Identify the guidelines for implementing a single-threaded apartment model.
- Identify the features of a multi-threaded apartment model.
- Identify the features of Microsoft Transaction Server (MTS).
- Create a package by using the Action menu in the MTS Explorer.
- Create an MTS component by using ATL.
- Add a component to a package by using the Action menu in the MTS Explorer.
- Set the properties of a component by using the pop-up menu in the MTS Explorer.
- Create a role for a specified package by using the MTS Explorer.
- Set a user identity for an MTS package.

Unit 25: Testing and Debugging Applications

- Identify the elements of a test plan.

Miracle Corporate Solutions Pvt. Ltd., C-41,GF, Behind Nirula's Hotel, Sec-2, Noida

Call : 0120-3058446/7

Mobile : 9311305846/47

Confidential

- Match the types of software testing with their uses.
- Identify the benefits of beta testing.
- Identify the uses of stress testing an application.
- Enable the library debug support for an application.
- Match the MFC debugging macros with their uses.
- Debug an application by using the integrated debugger in the Integrated Development Environment (IDE).
- Debug an application by using the Depends utility.
- Display specific information about an application by using the Spy++ utility.
- Identify the debugging tool that is to be used to resolve the programming errors in a given scenario.

Unit 26: Deploying a Visual C++ Application

- Match the deployment method with the factors to be considered when planning a deployment.
- Select the appropriate deployment strategy for a given scenario.
- Identify the reasons to select Microsoft Systems Management Server (SMS) as an aid in deploying a solution.
- Identify the reasons to select Zero Administration for Windows (ZAW) as an aid in deploying a solution.
- Package an application.
- Create a setup project that installs and uninstalls an application by using InstallShield.
- Assign files to file groups and file groups to components by using InstallShield.
- Place a shortcut in the Programs cascading menu by using InstallShield.
- Change the default background color of the setup by using InstallShield.
- Build the medium to run a setup by using InstallShield.

Unit 27: Maintaining and Supporting Applications

- Identify the different types of maintenance for an application.
- Identify the types of load balancing.
- Implement static load balancing by using the Dcomcnfg.exe utility.
- Identify the measures to be taken to prevent errors in an application.
- Select an appropriate deployment method for an application upgrade.

Unit 28: Creating and Accessing Databases

- Identify the features of SQL Server 7.0.
- Identify the features of physical database architecture components.
- Modify the properties of a database by using Enterprise Manager.
- Create a table by using Enterprise Manager.

Miracle Corporate Solutions Pvt. Ltd., C-41,GF, Behind Nirula's Hotel, Sec-2, Noida
Call : 0120-3058446/7
Mobile : 9311305846/47

Confidential

- Create an index by using Enterprise Manager.
- Create a relationship between two tables to enforce referential integrity.
- Import data from an external source by using Enterprise Manager.
- Access data from a single table by using the SELECT statement.
- Access data from multiple tables by using joins with the SELECT statement.
- Create summary queries to display data in customized sorted format.
- Modify the data contained in a table by using SQL statements.
- Create a view for restricting access to the specific columns to tables by using Enterprise Manager.
- Identify the features of the ODBC data access technology.
- Identify the features of the RDO data access technology.
- Identify the features of the OLE DB data access technology.
- Identify the features of the ADO data access technology.

Unit 29: Implementing Data Access in Applications

- Set up data source for SQL Server.
- Match the ODBC handles with their features.
- Sequence the steps associated with connecting to a data source.
- Sequence the steps to disconnect from a data source.
- Identify the advantages of stored procedure access model.
- Access data by using the stored procedure access model.
- Create a trigger to ensure referential integrity.
- Create a stored procedure that returns information.
- Identify the features of cursors.
- Identify the appropriate type of cursor that can be used in a given situation.
- Create a cursor to access data from a table.
- Match the fetch options with their functions.
- Update data by using a cursor.
- Close a cursor by using the SQL statement.
- Identify the features of a transaction.
- Create a transaction for the specified set of tasks.
- Match the types of locks with their uses.
- Identify the situations in which the locking strategies are implemented.

Exam 70-015: Designing and Implementing Distributed Applications with Microsoft Visual C++ 6.0

1. Deriving the Physical Design

- a) Explain the elements of an application that is based on the MFC framework.
 - i. Identify differences between developing an MFC application for Microsoft Windows NT®, Microsoft Windows® 95, and Windows 98.
 - ii. Explain when to use the Platform SDK (SDK) for an MFC application and when to use the functionality provided by the MFC framework.
 - iii. Choose whether to use an MFC regular DLL or an MFC extension DLL.
 - iv. Explain how command messages are routed between a user interface object and a handler function.
 - v. Describe the Document/View architecture
 - vi. Explain the MFC drawing, printing, and print preview architecture
 - vii. Explain how the MFC architecture supports multithreading

2. Evaluate whether access to a database should be encapsulated in an object.
 - i. Evaluate whether a database should be incorporated in the application.
 - ii. Identify which type of library to use. Valid libraries include MFC, ATL, and the SDK.

- iii. Identify which type of object to use. Valid objects include ADO, ODBC, and RDO.
 3. Design the properties, methods, and events of components.
 4. Establishing the Development Environment
 - i. Establish the environment for source-code control by using Microsoft Visual SourceSafe™. Issues include multiple user/multiple location development and versioning of the source code.
 5. Install the Visual C++ development tools that are necessary for developing a distributed application on various platforms. Platforms include Windows NT Workstation, Windows NT Server, Windows 95, and Windows 98.
 6. Install and configure server services. Services include Microsoft Transaction Server (MTS), SQL, and MSMQ.
 7. Configure a client computer to use an MTS component.
 8. Creating the User Interface
 - a) Implement the navigation for the user interface.
 - i. Create and integrate toolbars in an MFC application.
 - ii. Implement tool tips for toolbar buttons.
 - iii. Implement and write to the status bar in an MFC application.
 - iv. Given a scenario, select the appropriate options to create a new application by using the MFC AppWizard.
 - v. Create and edit user interface objects by using the resource editors.
 - vi. Create a new class by using ClassWizard.
 - vii. Add member variables by using ClassWizard.
 - viii. Add a message handler for an event by using ClassWizard.
 9. Create data input forms and dialog boxes.
 - i. Create a static menu by using the Menu editor.
 - ii. Create a dialog box by using the Dialog editor.
 - iii. Create property sheets by using ClassWizard.
 - iv. Create dialog box classes and members by using ClassWizard.
 - v. Use the **CFormView** class to create a view that contains controls.
 10. Validate user input.
 - i. Validate user input by using DDV.
 - ii. Validate user input by using ClassWizard.
 11. Process user input from a form or a dialog box by using DDX.
 12. Use an ActiveX® user interface control.
 - i. Insert a control into a project by using the Component Gallery.
 - ii. Handle an event from an ActiveX user interface control.
 - iii. Dynamically create an ActiveX user interface control.
 13. Use the MFC AppWizard to create an ISAPI DLL that can dynamically change Web content.
 14. Incorporate existing code into an application by using wizards and scriptlets.

Confidential

15. Create or modify an MFC application to store and retrieve personalized user settings from the registry.
16. Display data from a data source.
 - i. Implement remote data sources by using **CSocket**.
 - ii. Implement standard serialization by using **Serialize**.
 - iii. Implement persistence by using **CFile**.
 - iv. Display data by using **CArchive**.
 - v. Connect a recordset to dialog box controls.
17. Instantiate and invoke a COM component.
18. Add asynchronous processing.
 - i. Download ActiveX user interface controls.
 - ii. Create secondary threads.
19. Implement online user assistance in an application.
 - i. Implement status bars.
 - ii. Implement tool tips.
 - iii. Implement context-sensitive Help.
 - iv. Create Help for an application that provides links to a Web page that contains Help files.
20. Implement error handling.
 - i. Implement exception handling.
 - ii. Given an error, determine how to handle the error.
21. Use an active document.
22. Creating and Managing COM Components
 - a) Create a COM component that implements business rules or logic.
 - i. Create a COM component by using ATL.
 - ii. Create a COM component by using the SDK.
 - iii. Create a COM component by using MFC.
 - iv. Create an ATL COM in-process COM component and an ATL COM client to access it.
 - v. Create an ATL COM out-of-process COM component and an ATL COM client to access it.
 - vi. access it.
23. Create ActiveX user interface controls.
 - i. Create an ActiveX user interface control by using MFC.
 - ii. Create an ActiveX user interface control by using the SDK.
 - iii. Create an ActiveX user interface control by using ATL.

Confidential

24. Create a COM component that reuses existing components.
 - i. Explain the difference between aggregation and containment.
25. Add error handling to a COM component.
26. Log errors into an error log.
27. Create and use an active document.
28. Create a COM component that can participate in a transaction.
29. Sign a COM component.
30. Debug a COM component.
31. Create a COM component that supports apartment-model threading. Models include single-threaded apartment, multithreaded apartment, or both.
32. Create a package by using the MTS Explorer.
33. Add components to the MTS package.
34. Use role-based security to limit use of an MTS package to specific users.
35. Creating Data Services
36. Access and manipulate data by using ad hoc queries. Methods include ODBC, ADO, DAO, RDO, and data source control.
37. Access data by using the Stored Procedure model.
38. Manipulate data by using different cursor locations.
39. Manipulate data by using different cursor types.
40. Handle database errors.
41. Manage transactions to ensure data consistency and recoverability.
42. Write SQL statements that retrieve and modify data.
43. Write SQL statements that use joins to combine data from multiple tables.
44. Write SQL statements that create views or queries.
45. Use appropriate locking strategies.
 - i. Implement pessimistic locking.
 - ii. Implement optimistic locking.
46. Create a stored procedure that returns information.
47. Create triggers that implement rules.
48. Create reports that use summary data.
 - i. Display data in a customized stored format.
49. Create stored procedures to enforce business rules.
50. Create user-defined procedures and system-stored procedures.
51. Creating a Physical Database
52. Implement a data storage architecture by creating and managing files, file groups, and transaction logs.
53. Create databases, and create database tables that enforce data integrity and referential integrity.

Confidential

54. Create and maintain indexes.
55. Populate the database with data from an external data source.
56. Testing and Debugging the Solution
 - a. Determine appropriate debugging techniques.
 - i. Use library debug support.
 - ii. Use the IDE.
 - iii. Use Depends.
 - iv. Use Spy++.
 - v. Given a scenario, describe the type of debugging support Visual C++ provides for resolving programming errors.
 - vi. Step through code by using the integrated debugger.
 - vii. List and describe the MFC macros used to debug applications.
57. Identify and describe the elements of a test plan. Elements include beta testing, regression testing, unit testing, integration testing, and stress testing.
 - i. Evaluating the need for beta testing
58. Deploying an Application
59. Create a Setup program that installs an application and registers the COM components.
60. Register a component that implements DCOM.
61. Configure DCOM on the client computer and on the server computer.
62. Use .cab files to package and distribute an application.
63. Plan floppy disk-based deployment or compact disc-based deployment for an application.
64. Plan Web-based deployment for an application.
65. Plan network-based deployment for an application.
66. Given a scenario, evaluate the use of Microsoft Systems Management Server as an aid to deploying a solution.
67. Create a Setup program that installs an application and allows for the application to be uninstalled.
68. Evaluate Zero Administration for Windows (ZAW) as an aid to deploying a solution.
69. Maintaining and Supporting an Application
70. Implement static load balancing.
71. Fix errors, and take measures to prevent future errors.
72. Deploy application updates.