Course 80312A:

Development III in Microsoft Dynamics AX 2012

About this Course

This three-day instructor-led course puts the techniques learnt in Development I in Microsoft Dynamics AX 2012 and Development II in Microsoft Dynamics AX 2012 courses into practice directly in the application. It also introduces more advanced features of X++ and MorphX, and encourages the use of the Testing Framework to build for more reliable coding.

Audience Profile

The intended audience is experienced systems consultants typically working for a Microsoft Dynamics partner that is selling, consulting, implementing, and supporting Microsoft Dynamics AX 2012.

At Course Completion

After completing this course, students will be able to:

- Create a test case.
- Add methods to a test case.
- Run a test case.
- Build a test project and suite.
- Isolate test cases appropriately.
- Explain the MorphX development environment and the Application Object Tree
- Program optimal database access using a "while select" statement.
- Program optimal database access using queries.
- Describe the caching mechanisms in Microsoft Dynamics AX.
- Prevent and resolve database locking.
- Use temporary tables in classes, forms, and reports.
- List the reasons for using InitFrom methods.
- Use ParmId and ParmTables.
- Discuss date effectiveness and describe how to build date effective forms.
- Add a computed column to a view.
- Employ the various techniques available for integrating external data with Microsoft Dynamics AX.
- Use collection classes to store data in X++.
- List which application objects control different Graphical User Interface (GUI) components.
- Modify and use the Application Substituted Kernel Classes.
- Extend the RunBase framework to create new batch processes.
- Transfer information using the Args object.
- Identify the main sections that make up a form.
- Add data sources to a form to define what data is displayed by the form.
- Add controls to a form to display data.
- Modify form methods to the control how the form behaves when it opens and closes.
- Make decisions about where to place the code.
- Make runtime modification of the fetch of data.
- Explore the Application Object Tree (AOT) from Visual Studio.
- Create a project in Visual Studio.
- Write .NET managed code that uses X++ objects.
- Deploy managed code.
- Debug code using Visual Studio.
- Configure how the workflow engine is executed on a server.
- Specify which application module a workflow is applicable to using a workflow category.
- Link tables to workflows using a query.
- Create a new workflow type.

- Apply a workflow to a form.
- Define what happens when the workflow is approved or denied.
- Create Event Handlers and apply them to a workflow.
- Configure a workflow.

Course OutlineModule 1: X++ Unit Test Framework

This module describes how the X++ Unit Test framework allows for unit tests to be created along with the code they are designed to test.

Lessons

- Introduction
- Creating Test Cases
- Adding Methods to Test Cases
- Running Test Cases
- Build Test Projects and Suites

Lab : Create a Test Case

After completing this module, students will be able to:

- Create a test case.
- Add methods to a test case.
- Run a test case.
- Build a test project and suite.
- Isolate test cases appropriately.

Module 2: Working with Data

This chapter explains the correct approach to database functions when processing large amount of data in Microsoft Dynamics AX.

Lessons

- Introduction
- While Select
- Query
- Caching
- Locking
- Temporary Tables
- InitFrom
- ParmTables
- Date Effectiveness
- Computed Columns in Views
- Data Integration

Lab : Fetching DataLab : Converting QueriesLab : Reducing LockingLab : Temporary TablesLab : Integrating External Data

After completing this module, students will be able to:

- Explain the MorphX development environment and the Application Object Tree
- Program optimal database access using a "while select" statement.

- Program optimal database access using queries.
- Describe the caching mechanisms in Microsoft Dynamics AX.
- Prevent and resolve database locking.
- Use temporary tables in classes, forms, and reports.
- List the reasons for using InitFrom methods.
- Use ParmId and ParmTables.
- Discuss date effectiveness and describe how to build date effective forms.
- Add a computed column to a view.
- Employ the various techniques available for integrating external data with Microsoft Dynamics AX.

Module 3: Classes

This lesson introduces some of the most commonly used system classes, and demonstrates ways they can be used to support modifications.

Lessons

- Introduction
- Collection Classes
- Application Object Classes
- Application Substituted Kernel Classes
- RunBase Framework
- Args Object

Lab : Create a MapLab : Create a Query from CodeLab : Create a Global MethodLab : Make a RunBase ClassLab : Using Args

After completing this module, students will be able to:

- Use collection classes to store data in X++.
- List which application objects control different Graphical User Interface (GUI) components.
- Modify and use the Application Substituted Kernel Classes.
- Extend the RunBase framework to create new batch processes.
- Transfer information using the Args object.

Module 4: Forms

This module provides a comprehensive foundation for using forms in Microsoft Dynamics AX 2012 to interact with the end-user.

Lessons

- Introduction
- Architecture
- Data Sources
- Form Controls
- Form Methods
- Placement of Code
- Additional Controls

Lab : Create a FormLab : Use Unbound ControlsLab : Initialize a FormLab : Add a Window Control

After completing this module, students will be able to:

- Identify the main sections that make up a form.

- Add data sources to a form to define what data is displayed by the form.
- Add controls to a form to display data.
- Modify form methods to the control how the form behaves when it opens and closes.
- Make decisions about where to place the code.
- Make runtime modification of the fetch of data.

Module 5: Visual Studio Integration

This module explains the Visual Studio tools built specifically for Microsoft Dynamics AX development.

Lessons

- Introduction
- Application Explorer
- Visual Studio Projects
- Managed Code Projects
- Deploying Managed Code
- Visual Studio Debugging Experience for X++

Lab : Create a Managed Code ProjectLab : Create an Event Handler in Managed Code

After completing this module, students will be able to:

- Explore the Application Object Tree (AOT) from Visual Studio.
- Create a project in Visual Studio.
- Write .NET managed code that uses X++ objects.
- Deploy managed code.
- Debug code using Visual Studio.

Module 6: Workflow

This module introduces the development side of creating a workflow. Workflow is a system in Microsoft Dynamics AX 2012 that allows business processes to be automated.

Lessons

- Workflow Configuration
- Create a Workflow Category
- Create a Query
- Create a Workflow Type
- Enable Workflow on a Form
- Create a Workflow Approval
- Create Event Handlers
- Author a Workflow

Lab : Add another Condition to the Submit ActionLab : Enable Resubmit

After completing this module, students will be able to:

- Configure how the workflow engine is executed on a server.
- Specify which application module a workflow is applicable to using a workflow category.
- Link tables to workflows using a query.
- Create a new workflow type.
- Apply a workflow to a form.

- Define what happens when the workflow is approved or denied.
- Create Event Handlers and apply them to a workflow.
- Configure a workflow.

Before attending this course, students must have:

- working experience with Microsoft Dynamics AX and knowledge of Microsoft Dynamics AX 2012 development environment.
- completed Course 80303A, Development I in Microsoft Dynamics AX 2012.
- completed Course 80304A, Development II in Microsoft Dynamics AX 2012.