Microsoft

LESSON 6.1

10754 Microsoft .NET Fundamentals

Understand Language Interoperability

Lesson Overview

Understand language interoperability.

In this lesson, you will:

- Learn about language interoperability in the Microsoft® .NET Framework.
- Understand the benefits of language interoperability.
- Learn how to use assemblies from a different language.

Guiding Questions

1. What does “language interoperability” mean?
2. What is the Common Language Runtime (CLR), and how does it enable language interoperability?
3. What is language parity?

Anticipatory Set

1. Visit the Bing® Translator (*http://www.microsofttranslator.com/*) and try translating a few sayings into different languages.
2. Try some complex translations, such as long sentences or even slang. Then try converting the translated text back into English.
3. Why do some phrases translate better than others?

Language Interoperability

- Language interoperability is the ability of code to interact with code that is written in a different programming language.
  - For example, a developer working in Microsoft C# can reference and use code written in Microsoft Visual Basic®.
- Language interoperability is an important feature of the .NET Framework; it can help maximize code reuse and improve the efficiency of the development process.

The CLR and Language Interoperability

- .NET Framework enables cross-language interoperability through its use of the CLR.
- The CLR is a run-time environment that executes .NET code. It supports language interoperability by doing the following:
  - Providing a common system of data types that all .NET languages can use.
  - Using metadata to store data about custom types; the CLR can read this metadata and use the defined classes, regardless of their original language.

Interoperability Features Provided by the CLR

.NET Framework code benefits from the CLR's support for language interoperability in the following ways:

- Types can inherit implementation from other types, pass objects to another type's methods, and call methods defined on other types, regardless of the language the types are implemented in.
- Debuggers, profilers, or other tools are required to understand only one environment—the Microsoft intermediate language (MSIL) and metadata for the CLR—and they can support any programming language that targets the .NET CLR.
- Exception handling is consistent across languages. Your code can throw an exception in one language, and that exception can be caught and understood by an object written in another language.

Common Language Specification

- There is no guarantee that the functionality of the types that you create can be fully used by the programming languages that other developers use.
  - For example, the code may use features that are not available in the other language.
- To ensure that your managed code is accessible to developers who are using other programming languages, the .NET Framework provides the Common Language Specification (CLS).
  - The CLS describes a fundamental set of language features and defines rules for how those features are used.
  - Code that is CLS-compliant can be used effectively by any .NET language, since it only uses features that are common to all .NET languages.

CLS-Compliant Code

- If you want your code to be CLS-compliant, you must expose functionality in a way that is CLS-compliant in the following places:
  - Definitions of your public classes
  - Definitions of the public members of public classes, and of members accessible to derived classes
  - Parameters and return types of public methods of public classes, and of methods accessible to derived classes
- The features that you use in the definitions of your private classes, in the definitions of private methods on public classes, and in local variables do not have to follow CLS rules.
  - You also can use any language features you want in the code that implements your class and still have a CLS-compliant component.

Using Assemblies from Other Languages

- Developers can use CLS-compliant assemblies from a different language in the same way they use assemblies from the same language—by adding a reference.
  - From the Project menu, select Add Reference.
  - To add a DLL, do the following: in the Add Reference dialog box, choose the Browse tab and locate the desired dynamic-link library (DLL) file.
  - To add a project, do the following: Select the Project tab and choose the desired project.
- Once the reference is added, classes from the assembly can be referenced with their fully qualified name.

.NET Language Parity

- Beginning with .NET version 4.0, Microsoft has focused on establishing **language parity**.
  - That is, the teams that develop .NET are striving to make sure that language features are consistent (or “equal”) across all the .NET languages.
- This has the effect of expanding the CLS so that more features are CLS-compliant, and more code that developers write is interoperable across languages.

Ticket Out the Door

Answer the following questions:

1. What is language interoperability?
2. What is the CLS?
3. What is language parity?