Introduction

The ability to run .NET code in the database is arguably the most exciting development in SQL Server for years. Traditionally, if T-SQL did not meet your needs, then you could write the required logic in either an external business object or an extended stored procedure. With the former, you could often generate unnecessary network traffic, and you also would lose the advantages associated with encapsulating data-centric logic in a single location (i.e., the database). The latter were complex to write and notorious for decreasing the stability of SQL Server.

In SQL Server 2005, which hosts the common language runtime (CLR), all this has changed. You can now write modules such as stored procedures, triggers, functions, and custom types in a .NET language instead of T-SQL. These modules have access to powerful .NET Framework classes, so they vastly extend the processing and formatting capabilities available through T-SQL. They also allow access to custom data sources for which there may not be an ODBC driver or OLE DB provider.

What This Book Covers

The coding requirements for a SQL Server assembly differ somewhat depending on the type of SQL Server module the assembly implements—for example, writing a user-defined type requires a bit more work than writing a CLR stored procedure.

The bulk of this book consists of chapters that walk you through how to build each type of SQL Server assembly. In each case, we provide carefully chosen examples that demonstrate business problems where assemblies could be of true benefit. For example, we show how .NET greatly simplifies working with images and XML data, and accessing external data sources and web services. We also devote chapters to debugging and error handling strategies in SQL Server assemblies and to their security implications.

The following sections provide chapter-by-chapter overviews.

Chapter 1: Introducing Assemblies

We start by putting SQL Server assemblies in context and looking at what they are, why they're useful, and what we can do with them.

Chapter 2: Writing a Simple SQL Assembly

Now that we've covered the theory, it's time to get our hands dirty with a real example. Here we show how to create and deploy a simple CLR stored procedure, covering every step of the process and providing instructions for both Visual Studio 2005 and the command-line compiler.

Chapter 3: The SQL Server .NET Programming Model

In this chapter, we introduce the new .NET classes used to build SQL Server assemblies. These include the `SqlContext` class, which provides information about the context in which the assembly is executing, and the `SqlPipe` class, which allows us to send information back to the caller. We'll also show how to access SQL Server data from within a SQL assembly.

Chapter 4: CLR Stored Procedures

Once we've covered the basics, we can start to look in detail at the individual types of SQL Server assemblies we can build. We start with the simplest type: stored procedures. We'll also show how .NET can make image manipulation and working with XML much easier, and how we can use .NET to execute external programs.

Chapter 5: User-Defined Functions

User-defined functions (UDFs) come in two flavors: scalar-valued functions, which return a single value, and table-valued functions, which return a resultset of values. We'll demonstrate how to create and use both types, and along the way we'll also cover how to work with Active Directory and how to browse the file system.

Chapter 6: User-Defined Types

User-defined types (UDTs) have been around for some time in SQL Server, but the ability to create these types in .NET greatly increases their functionality. Unlike traditional UDTs, CLR UDTs don't need to be based on the standard SQL Server types, and they can also expose properties and methods. In this chapter, we look in detail at creating CLR UDTs and illustrate their use with types representing a time duration and an e-mail address.

Chapter 7: User-Defined Aggregates

As well as the standard UDFs we met in Chapter 5, we can also use SQL Server assemblies to define custom aggregate functions that work on a set of rows. In this chapter, we look in detail at building user-defined aggregates (UDAs) and also examine how to use UDAs in combination with UDTs by building a UDA that calculates the average for a column of the duration type we defined in Chapter 6. Finally, we look at a full implementation of the population variance statistical function.

Chapter 8: CLR Triggers

As well as the ability to write triggers in a .NET language, SQL Server 2005 boasts another major development on the trigger front: the ability to write triggers to react to DDL events, such as creating or dropping stored procedures or tables. In this chapter, we look at both DDL and DML triggers in .NET and show how to use DDL triggers to create automatic backups of source code.

Chapter 9: Error Handling and Debugging Strategies

Once we've covered each type of SQL Server assembly, we pan out again to look at a couple of more general issues relating to all assembly types. In this chapter, we consider two distinct but related topics: error handling and debugging. We start by looking at debugging SQL Server assemblies, both in the full version of Visual Studio 2005 and in the pared-down version that ships with SQL Server 2005. Then we examine how to handle any errors that occur, including sending e-mails from a SQL assembly and writing entries to an event log.

Chapter 10: Security

.NET provides two main security mechanisms: role-based security, which allows us to impersonate Windows users and restrict execution of code to particular users or groups of users, and code access security (CAS), which lets the developer or administrator restrict what actions a section of code can perform. In this chapter, we look at both these mechanisms and explore how to use them in SQL assemblies.

Chapter 11: Integrating Assemblies with Other Technologies

In this final chapter, we examine a couple of different applications of SQL assemblies. First, we look at integrating SQL assemblies and XML web services, and show how to access a web service from within a SQL assembly. Then we cover how to use assemblies with a completely new SQL Server technology: the Service Broker message-queuing system.

Who This Book Is For

This book is intended for SQL Server developers and DBAs who want to learn what SQL Server assemblies can do for them and want to know what the advantages and possible pitfalls are of allowing .NET code to run in the database.

You should have a working knowledge of the .NET Framework, and specifically of the C# language, as all the examples in this book are in C#. However, the code is explained in detail, so you don't need to have an in-depth knowledge of every class in the .NET Framework Class Library.

Note All of the C# code examples for this book, along with their Visual Basic equivalents, can be downloaded from the Apress web site, `http://www.apress.com`.
