



16212 NE 113TH CT. REDMOND, WA 98052
WWW.BETA.V.COM
(425) 556-9205

COURSEWARE OFFERINGS

2009

COURSEWARE OFFERINGS

2009

BETA V CORPORATION

Providing focused technical content, mentoring and training to those who design, code and deploy data access applications.

William R. (Bill) Vaughn, Beta V Corporation's founder and President, has worked in the computer industry since 1972. He spent over 14 years at Microsoft where he held positions at Microsoft University, the Visual Basic documentation team, the Visual Studio Marketing team, and at Microsoft Technical Education. Over the years, he has written seven editions of his popular Hitchhiker's Guides, two books on ADO and ADO.NET, a new book on SQL Server 2000 Reporting Services as well as dozens of published papers focusing on data access technology. His latest is *Hitchhiker's Guide to Visual Studio and SQL Server (7th Edition)* upon which many of these courses are based. His other works include:

- *Hitchhiker's Guide to SQL Server 2005 Compact Edition* (EBook) available [online](#).
- *Hitchhiker's Guide to SQL Server 2000 Reporting Services* (Addison Wesley, ISBN 0-321268-28-8).
- *ADO.NET and ADO Examples and Best Practices for VB Programmers* (Apress, ISBN 1-893115-68-2)
- *ADO.NET Examples and Best Practices for C# Programmers* (Apress, ISBN 1-590590-12-0)
- *Hitchhiker's Guide to Visual Basic and SQL Server—6th Edition* (Microsoft Press, ISBN: 1-57231-848-1)
- *ADO Examples and Best Practices* (Apress, ISBN 1-893115-16-X)

Mr. Vaughn is a regular contributor to MSDN online, SQL Server Magazine, Developer.com and has written from time-to-time for several other publications. For more information on Beta V Corporation, visit www.betav.com.

Over the last thirty-seven years, Mr. Vaughn has been a featured speaker at developer conferences all over the world. He was the top-rated speaker at TechEd in 1998 and continues to be ranked among the top presenters at TechEd as well as VSLive, Developer Connections, PASS, VBUG and other developer conferences. His light, though highly technical and forthright style wins him accolades from both peers and the attendees. He is regularly sought out by companies like Microsoft, Boeing and WatchIT as well as a number of other companies to provide insightful commentary and technical briefings for developers, architects and their managers. Mr. Vaughn has delivered several of his sessions via NetMeeting or other online delivery means which dramatically reduces the cost incurred by the students without significant loss of content and interaction.

Mr. Vaughn's focus is on data access technology and Microsoft SQL Server. He helped design, refine and document many of the data access technologies while at Microsoft, and he wrote developer documentation for Visual Basic versions 2.0 through 6.0. Bill has taught courses on ADO programming and performance tuning, SQL Server, Reporting Services, CLR executables development, as well as several focused ADO.NET topics and day-long workshops.

Since he retired from Microsoft, Mr. Vaughn has responded to thousands of questions posted on public newsgroups and has become a respected source of insight on ADO and ADO.NET, Visual Studio .NET, Reporting Services, the reporting controls in Visual Studio and their integration with all versions of SQL Server. He has currently expanded his repertoire to include courseware on SQL Server Everywhere Edition.

Mr. Vaughn was appointed a Microsoft MVP in 2002 and has been renewed every year since. In 2005 he was added to the prestigious list of INETA speakers. He has also been conferred with the honorary degree of Doctor of Computer Technology by the University of Advancing Computer Technology in Tempe, Arizona. Mr. Vaughn has also volunteered to lead the local .NET user group and act as the data access SIG leader and provide pro-bono help to the user communities by answering questions and providing advice on several public newsgroups and web forums.

COURSEWARE OVERVIEW

Beta V Corporation's courseware is designed to provide fast-paced, highly technical training or mentoring to bring your staff or conference attendees up to speed on the latest technology quickly and at the lowest possible cost. Courses do not simply teach data access foundations and fundamentals, they also teach "best practice" concepts to ensure that the designs and code created by the students meet the highest industry standards.

New for 2009 are several conference-length sessions focusing on SQL Server 2008 and Visual Studio 2008 technologies. We've also updated all of our day-long workshops to reflect the latest Visual Studio 2008 and .NET Framework 3.5 changes including enhancements and new issues raised by ADO.NET 3.5.

All live courses are accompanied by live instructor-led demonstrations that illustrate key points while at the same time giving the students an opportunity to ask questions and gain a better understanding of the technology and the tools. Depending on the venue, sessions run at the pace dictated by the audience and are typically 70-90 minutes in length.

Most courses can be delivered over the Internet to sites anywhere in the world using real-time Webcast delivery technology. Generally, this is the most cost-effective way to have a course presented. The only requirement is a system tied to the Web and a projector and a speaker-phone tied to the venue's public address system.

All courses can include appropriate code demonstrations and walk-throughs and student handouts that include the presentation slides and hands-on lab instructions where applicable. These materials can be printed and bound at your site or printed and shipped to your site in time for the course.

Courseware can be delivered "as-is" from our stock courses or customized to meet specific customer requirements. While customization of a course incurs additional fees, it is often better suited to companies with a limited budget as courses are more specifically focused on specific requirements. This means a single custom course might provide the needed training more efficiently than several, more general courses.

COURSES

As you know, Microsoft constantly reinvents data access and application architecture technology. This poses a perennial question to development managers: Should they retool for the new technology or simply maintain the status quo and try to make what's currently working work more efficiently? Beta V Corporation has courses that can meet both requirements. We can help bring developers, architects, and managers up to speed on the newest .NET technology or simply help create more efficient current-technology solutions. Much can be done to make existing code more competitive—and easier to migrate—once that decision has been made.

TABLE OF CONTENTS

Leveraging SQL Server 2008 Functionality via ADO.NET 3.5	6
Managing Query and Report Parameters with the MicrosoftReportViewer	7
Managing Server-Side Executables in SQL Server Reporting Services	8
SQL Server Compact Edition	9
Hitchhiker’s Guide to Visual Studio and SQL Server—Connecting.....	10
Hitchhiker’s Guide to Visual Studio and SQL Server—Reporting.....	11
Hitchhiker's Guide to Visual Studio and SQL Server—CLR Executables.....	12
ADO.NET Intermediate Development Topics.....	13
Managing and Writing High-Performance SQL Server Stored Procedures	15
Microsoft SQL Server Reporting Services— for Architects, DBAs and Developers.....	16
Staying Afloat in the Connection Pool	17
Hithchhiker’s Guide to Data Access Architecutures Full-Day Workshop.....	18
ADO.NET Workshop (Half-day)	20
Visual Studio .NET for ADO Classic Developers	21
ADO.NET for ADO Classic Developers.....	22
Managing Server-Side Curors and Locks with ADO.NET.....	23

LEVERAGING SQL SERVER 2008 FUNCTIONALITY VIA ADO.NET 3.5

70-90 Minutes

This session brings Visual Studio developers up to speed on the latest ADO.NET 3.5 technology as exposed by Visual Studio SP1—especially in regard to new SQL Server 2008 features. We'll focus on best practice architectures and implementation of the new Synchronization Services technology as well as the new Filestream and other useful TSQL operators that can make applications faster, more secure and easier to design and code. This session also illustrates how to leverage the new Table-valued Parameter features supported by ADO.NET 3.5 and SQL Server 2008 and how to implement the Local Data Cache functionality first exposed in Visual Studio 2008.

- SQL Server 2008 and Visual Studio 2008 SP1
 - What's been enabled and what's still missing?
 - New SQL Server 2008 Data Types
 - Time, Date, TVPs, Geospatial
- Table-Value Parameters
 - Setup, best-practice architecture, implementation
- FILESTREAM vs. BLOB
 - Setup, measuring performance
- Synchronization Architectures
 - Synchronization Services
 - SQL Server Compact Edition
 - Visual Studio 2008 Local Data Cache

**MANAGING QUERY AND REPORT PARAMETERS WITH THE
MICROSOFTREPORTVIEWER**

70-90 Minutes

- This session focuses on the techniques used to manage parameters and execute client-side rendered reports using the MicrosoftReportViewer (MRV) control. The MRV control, adapted from the Visual Studio 2005 ReportViewer control, permits developers to leverage RDL based reports executed on the server using Reporting Services or locally on a Windows Forms or ASP.NET client. Because Reporting Services is not required to generate client-side reports, developers must take on the tasks performed automatically by the server-side report processor. We demonstrate techniques to build multi-select pick-lists and incorporate these values into the report through code-managed manipulation of the Local Parameters collection. We also illustrate code used to pass parameters to sub-reports via secondary forms or via click-through.

This course discusses and demonstrates the following topics:

- Building a sample application leveraging the *MicrosoftReportViewer* control.
- Creating a Report Definition Language (Client) (RDLC) Report using the Visual Studio 2008 Report Wizard.
- Customize the report to accept one or more report parameters and feed report expressions from these parameters.
- Build a suitable data source for the report query.
- Generate and test a suitable parameter-driven query.
- Build user-interface (UI) control elements to capture parameter values.
- Construct multi-select dialog to capture parameters.
- Apply captured parameters to the MicrosoftReportViewer control's Parameter collection.
- Demonstrate use of Parameter value capture from Report cells. Pass these values to a sub-report.

MANAGING SERVER-SIDE EXECUTABLES IN SQL SERVER REPORTING SERVICES

70-90 Minutes

This session is designed to help developers working with SQL Server Reporting Services (2005-2008) manage both query and report parameters, incorporate custom code into expressions and build CLR DLLs to be called by Reporting Services-rendered reports. Building efficient and meaningful reports is a function of focusing the query on a specific subset of the database, and in some cases this can only be achieved by executing complex logic on the server. While it's possible to write logic that's interpreted by the Report Processor, it's sometimes necessary to build your own Visual Basic.NET or C# DLLs to perform the needed filtering or value generation tasks. This session shows how all of this is done using a series of demonstrations.

This course discusses and demonstrates the following topics:

- Build an RDL-based report using the Visual Studio BI toolset.
- Customize the report to accept one or more report and query parameters and feed report expressions from these parameters.
- Incorporate a code-based expression into the RDL definition.
- Build a custom .NET CLR DLL to execute logic to filter report rows and affect RDL expressions.
- Create an external CLR executable function to alter data in a report cell.
- Illustrate debug and test scenarios for the server-side CLR executable.
- Illustrate deployment of the server-side DLL.

SQL SERVER COMPACT EDITION

70-90 Minutes

Based on Bill's EBook *Hitchhiker's Guide to SQL Server 2005 Compact Edition* and updated to reflect changes made in Visual Studio 2008 and the 3.5 version of the database engine, this session digs into the realities of Microsoft's newest SQL Mobile reincarnation—SQL Server Compact Edition (what Bill calls "SQL Compact"). We'll show every aspect of this SQL engine including where it fits in a comprehensive data access solution whether it's on a Windows desktop or a mobile phone. You'll learn how SQL Compact can replace other, less formal data access solutions like JET, MySQL and other free alternatives that are less secure. We'll show how to get this latest version, how to install it and most importantly, how to best leverage its strengths. We'll contrast the differences between this SQL Mobile-based engine and other "real" SQL Server editions. We'll show how to design applications that include SQL Compact, build SQL Compact databases, populate tables as well as write queries for it to process. We'll also show how to bulk-load data and synchronize a SQL Compact database from SQL Server—a critical feature for many designs. We'll introduce the SQLCE namespace that not only serves as a query interface but is used to perform database maintenance as well. When we're done you'll understand the capacities and limitations as well as how to avoid the issues raised by SQL Compact.

This course discusses and demonstrates the following topics:

- How to know where SQL Compact fits, its limitations and strengths and how it differs from other SQL Server editions.
- How to download, install and configure the SQL Compact SDK and its DLLs.
- How to create, populate and replicate SQL Compact databases—and how to encrypt them.
- How to design applications to leverage SQL Compact's strengths.
- How to write queries and use SQL Compact's unique scrollable, bindable, updatable cursors.
- How to deploy SQL Compact applications without having to run in admin mode.
- How to use Visual Studio to perform most of these operations.
- How to use the SQLCE namespace to manage, compact, backup and compress SQL Compact databases.

HITCHHIKER'S GUIDE TO VISUAL STUDIO AND SQL SERVER—CONNECTING

70-90 Minutes

Based on Bill's latest book, this session walks you through a series of live code examples that illustrate how to connect to SQL Server and other backend servers—including the new 2008 Express and other versions. The session includes examples of the new *SqlConnectionStringBuilder* (and why it makes sense), the new *User Instance* feature of SQL Express, the new Session variables and other connection options that can help your application connect quickly and stay connected as well as encrypt your connection string. We'll illustrate how and when it makes sense to use the new Visual Basic .NET *Using* statement to manage *SqlConnection* instances and how the connection pooling mechanism has been reengineered to make your exception handlers easier to write. We'll show how to setup, monitor (and for the first time) flush the connection pool so your applications won't leak connections. We'll show how to manage the *SqlConnection* instances to prevent connection pool leakages by using new ADO.NET and language features. We'll also demonstrate the new factory classes in the .NET Framework used to discover and manage the services used to support SQL Server and its adjunct functionality like FullText Search, Reporting Services and Analysis Services.

This course discusses and demonstrates the following topics:

- How to build a *ConnectionString* using traditional methods and the new *SqlConnectionStringBuilder* class.
- How to persist a *ConnectionString* in a strongly typed *Settings* variable.
- How to leverage the new Visual Basic .NET *Using* statement to manage the scope and lifetime of object instances.
- How to monitor the connection pool using component counters. How to know which counters really work.
- How to manage SQL Server Express User Instance sessions.
- How to flush the pool and know when to flush it.
- How to know when Multiple Active Resultsets (MARS) makes sense.
- How to open connections asynchronously—without ADO.NET 2.0.
- How to find visible SQL Server service instances on the network and how do start, stop, pause or simply determine their status.

HITCHHIKER'S GUIDE TO VISUAL STUDIO AND SQL SERVER—REPORTING

70-90 Minutes

Based on Bill's latest book, this session walks you through a series of live code examples that illustrate how to leverage the Visual Studio 2008 *MicrosoftReportViewer* control and SQL Server 2008 second-generation Report Definition Language (RDL) (and RDLC)-based reporting features. We'll see how to create a new report from scratch and find out why we can no longer import a report created for SQL Server Reporting Services into Visual Studio's new *MicrosoftReportViewer* control. We'll setup a report-specific data source, capture user parameters to focus the report data, capture multiple-select parameters, pass parameters to the queries and execute stored procedures to return data for the report. We'll see how to create custom data sources as well as build a report DLL to permit report updates without having to redeploy the application. We'll show how to deploy the reports and implement (fake) some of the features you'll find only on Reporting Services. We'll also look at the SQL Server Reporting Services features that can leverage this same technology as implemented in SQL Server Express and other editions. When you're done you'll have a better understanding of the differences between the Visual Studio and SQL Server Reporting Services implementations.

This course discusses and demonstrates the following topics:

- How to setup a report project.
- How to build a suitable report-specific Data Source.
- How to use the ReportViewer control to address local as well as server-based reports.
- How to layout a report using the Table, Matrix, Graph and Image controls.
- How to address the data source columns and map them to report elements.
- How to capture and insert parameters into the data source queries.
- How to deploy the report or report application.
- Learn the difference between the Visual Studio and Reporting Services implementations provided by SQL Server.

HITCHHIKER'S GUIDE TO VISUAL STUDIO AND SQL SERVER—CLR EXECUTABLES

60-90 minutes

One of the most talked-about new features in SQL Server is its ability to execute CLR-based assemblies on the server. This topic is especially important for DBAs concerned with developers' use of CLR code. While SQL Server has been able to execute non-TSQL code for some time, this session discusses how developers and DBAs can use Visual Studio to create, test, deploy and debug CLR-based stored procedures, User-defined Types (UDTs) and User-Defined Functions (UDFs). We'll discuss where this technology makes sense (and where it doesn't) and how it's implemented. The course walks through three examples: one that does not make sense for conversion but illustrates the development techniques, and two others that provide a rich example of the issues and techniques associated with writing CLR Assemblies for SQL Server.

This course discusses and demonstrates the following topics:

- What is CLR-based code? How is it written? What languages can be used? What happens to TSQL?
- When does CLR-based code make sense? What are good examples of its use?
- How does SQL Server support managed assemblies? Where are they stored, and how can they be managed?
- How are CLR Stored Procedures or Triggers created in Visual Studio? How is the code deployed and tested?
- How are CLR User Defined Types (UDTs) created, deployed and tested? How can they be optimized to reduce storage and data transport requirements?
- How are User-defined aggregates created, deployed and tested?
- How are CLR assemblies secured? How do they protect the server resources?
- How can CLR assemblies be debugged? Can execution be paused when the assembly is called by SQL Server?

ADO.NET INTERMEDIATE DEVELOPMENT TOPICS

60-90 minutes

This session includes several shorter topics that discuss solutions to problems frequently encountered by ADO.NET developers. These include handling identity issues, managing the connection pool, and efficiently executing stored procedures.

This session includes a discussion on how to manage Identity columns in an ADO.NET architecture that uses disconnected DataSets. The session starts with a short discussion of Identity and uniqueidentifier columns and when each makes sense. Next, we'll demonstrate how to write code to get applications to create Identity values and use these values to manage client-side parent-child relationships. Once the data has been updated on the server, we'll show how to retrieve the new server-generated Identity values using techniques that won't fall apart when your DBA adds a trigger or your update stored procedure executes a function.

This session explains the inner workings of the connection pool, how to configure it, which applications can best utilize it, when to disable it and how to recover from exceptions that can damage it. We also discuss which application coding techniques are likely to cause an overflow or damage. We'll also discuss how the connection pool mechanism can be monitored and how it's going to change when the 2.0 version of the .NET Framework arrives.

SQL Server stored procedures are used by companies large and small to fetch intelligent rowsets, perform updates, manage data and referential integrity, enforce business rules and protect the underlying data. These server-side blocks of TSQL code can improve performance if they're written and executed correctly or just make a developers (or DBA's) life miserable. The problem is, most developers do not fully understand how SQL Server compiles, caches and reuses the query plan. This session focuses on a number of "best-practices" used to optimize developer and code performance when stored procedures are part of your application's design.

This course discusses and demonstrates the following topics:

- Coding and testing stored procedures in Visual Studio .NET and Query Analyzer. Understanding and examining the generated query plan and watching how changes in the parameters affect the plan.
- Optimizing stored procedures: Understanding how SQL Server constructs and caches the query plan. How do changes in the parameter set affect performance and cached plans? What other factors influence how the query plan is built?
- When should you use recompilation to be sure you're executing the right cached query plan? What statements or operations can be used to flush the cache, force recompile and ensure that you're testing with the right plan?
- Understanding when CLR-based procedures or Extended Stored Procedures can help performance or solve "unsolvable" problems.
- How should stored procedures be coded to optimize performance? How are stored procedures executed with ADO.NET and what are the performance trade-offs?
- Building the stored procedure parameters collection.

- Managing OUTPUT parameters. Why are OUTPUT parameters more efficient than rowsets? How are OUTPUT Parameter objects coded? What debug implications will you encounter?
- How to manage the pool—how to monitor its state and know when it's going to overflow. How to design applications and queries to best utilize the pool. Which coding techniques tend to lead to pool overflows?
- How to gain the most scalability from your applications so as to minimize connect time.
- How to know beforehand that your pool is going to overflow. How does one code Performance counters and other techniques to monitor the pool?
- What are Identity columns? How many identity values can the server manage? What happens to deleted identity values?
- How are INSERT statements coded when Identity columns are being used? How can you override this behavior and force your own value to be used instead of the system-generated Identity value?
- How can you create Identity values in a disconnected DataTable so they don't collide with server-created values? How can parent-child relationships be created and managed with client-side identity values?
- How can server-generated values be retrieved post Update? What's the difference between @@IDENTITY, IDENT_CURRENT and SCOPE_IDENTITY?
- How are GUID UniqueIdentifiers created? How are these passed to the server at Update time?

**MANAGING AND WRITING HIGH-PERFORMANCE
SQL SERVER STORED PROCEDURES**

60-90 minutes

SQL Server stored procedures are used by companies large and small to fetch intelligent rowsets, perform updates, manage data and referential integrity, enforce business rules and protect the underlying data. These server-side blocks of TSQL code can improve performance if they're written and executed correctly or just make a developers (or DBA's) life miserable. The problem is, most developers do not fully understand how SQL Server compiles, caches and reuses the query plan. This session focuses on a number of "best-practices" used to optimize developer and code performance when stored procedures are part of your application's design.

This course discusses and demonstrates the following topics:

- Coding and testing stored procedures in Visual Studio .NET and Query Analyzer. Understanding and examining the generated query plan and watching how changes in the parameters affect the plan.
- Optimizing stored procedures: Understanding how SQL Server constructs and caches the query plan. How do changes in the parameter set affect performance and cached plans? What other factors influence how the query plan is built?
- How can developers or DBAs view the query plan and understand the GUI representation of the plan?
- When should you use recompilation to be sure you're executing the right cached query plan? What statements or operations can be used to flush the cache, force recompile and ensure that you're testing with the right plan?
- Understanding when CLR-based procedures or Extended Stored Procedures can help performance or solve "unsolvable" problems.
- How should stored procedures be coded to optimize performance? How are stored procedures executed with ADO.NET and what are the performance trade-offs? How can developers improve stored procedure performance by choosing how they're executed?
- Building the stored procedure parameters collection. How to use drag-and-drop or the DataAdapter Configuration Wizard to build the Parameters collection. Using (or avoiding) the CommandBuilder. How to use the Add method overloads to simplify your code.
- Managing OUTPUT parameters. Why are OUTPUT parameters more efficient than rowsets? How are OUTPUT Parameter objects coded? What debug implications will you encounter?

**MICROSOFT SQL SERVER REPORTING SERVICES—
FOR ARCHITECTS, DBAS AND DEVELOPERS**

60-90 minutes

At long last, developers have a new Microsoft server-based reporting solution built from the ground up, almost completely but not entirely in .NET managed code. The session drills into the inner workings of Reporting Services so that architects, database administrators and developers can know how best to understand and leverage its features. This talk is extracted from the highly acclaimed book “*Hitchhiker’s Guide to Microsoft SQL Server 2000 Reporting Services*” (Addison Wesley). The session includes extended demonstrations of the Visual Studio .NET Report Designer addin used to create dedicated and shared data sources, parameterized DataSets, layout and format report data and deploy the finished report to the catalog database. It also demonstrates how to manage reports and users with the Report Manager.

The course discusses and demonstrates the following topics:

- What are Microsoft SQL Server Reporting Services? What are the fundamental components? How do they interact?
- What needs to be in place for Reporting Services—what OS and application platforms are required to run it? How much does it cost?
- How does Reporting Services protect company data? Is confidential or personal data or credentials encrypted as it passes over the Internet? How are the reports themselves protected? How can DBAs use the Report Manager to grant rights to users or domain roles?
- How are reports authored? What Microsoft and third party tools are available to create the reports? How can the Visual Studio .NET addin be used to author and deploy reports?
- How are dedicated and shared data sources created? How can data be extracted from different (non-SQL Server) data sources.
- How are DataSets created? How are DataSets parameterized and the parameter values captured when the report is rendered? What are linked reports?
- How are Report and Query parameters created? How are parameters used in expressions? How can data be filtered, sorted, grouped and organized using and after the query?
- How are reports compiled, deployed and rendered? How can DBAs deploy and create report subscriptions? How are reports cached?
- How are reports rendered? How can users render reports in other formats such as Excel, delimited, PDF, XML or custom formats?

STAYING AFLOAT IN THE CONNECTION POOL

60 minutes

Virtually all applications use the .NET or ADO classic connection pool whether they know it or not. Unfortunately, some developers discover this only after the pool has overflowed and the application drowns. Connection pools are used differently by ASP.NET and client/server or Web Service applications—we'll discuss all three architectures. This session explains the inner workings of the connection pool, how to configure it, which applications can best utilize it, when to disable it and how to recover from exceptions that can damage it. We also discuss which application coding techniques are likely to cause an overflow or damage. We'll also discuss how the connection pool mechanism can be monitored and how to manage it using ADO.NET enhanced functionality.

This course discusses and demonstrates the following topics:

- How to build a connection object in ADO.NET.
- How to build a connection string that best leverages the pool and still provides good performance and adequate security.
- How to manage the pool—how to monitor its state and know when it's going to overflow.
- How to design applications and queries to best utilize the pool. Which coding techniques tend to lead to pool overflows?
- How to gain the most scalability from your applications so as to minimize connect time.
- How to know beforehand that your pool is going to overflow.
- How does one code Performance counters and other techniques to monitor the pool?
- What control do you have over the pool mechanism?
- How can the pool be cleared when it's full of corrupted connections? How can you prevent a connection from getting corrupted?
- How does Visual Studio .NET and SQL Server 2005 impact programming the pool?

**HITHCHHIKER'S GUIDE TO DATA ACCESS ARCHITECTURES
FULL-DAY WORKSHOP**

8 Hours

COM-based ADO classic (ADOC) and ASP developers world-wide are ramping up to learn ADO.NET 2.0, as well as how to best port existing code to run in the .NET Framework. This day-long session is designed with these developers in mind. It can help ADO.NET developers building Smart Client, ASP or XML Web Services get a more in-depth understanding of how ADO.NET 2.0 can be used to address simple to complex data access problems. This course ensures that developers get a running start at ADO.NET 2.0 and become productive far faster than if they simply try to learn how to code within the new .NET technology on their own. By taking an entire day to discuss ADO.NET 2.0, we can afford to spend more time discussing specific details of the technology. We cover all of the new ADO.NET 2.0 features including the new connection-management tools, asynchronous operations, schema management tools and much more.

This course discusses and demonstrates the following topics:

- What is ADO.NET? How is it different from COM-based ADOC? Is it faster, smaller, and more efficient or not? How can existing ADOC code be morphed to run in the .NET Framework? Should it be? Does the upsizing wizard convert ADOC code? If so, how? What code does not convert? What's new with ADO.NET 2.0?
- How do .NET data access strategies differ between the various ADO.NET architectures such as Windows Forms, Web Services, and Web Applications? How are pessimistic locks or server-side cursors created and managed in ADO.NET? Should developers re-embrace data binding?
- What's the difference between the various .NET Data Providers? How are Oracle, DB2, Sybase, and other non-Microsoft database platforms supported? What about OLE DB and ODBC data sources?
- How does your code connect to a .NET Data Provider? How does this approach differ in the various .NET architectures? Do existing ADOC or ODBC connection strings still work in ADO.NET? How can you test for network and server visibility using simple Framework calls? How can you start, pause or stop a service?
- How do applications manage connection security? How can connection strings be created and persisted for Windows, Web Service and ASP applications. How do you connect to a database using integrated security or hard-coded Login IDs? What do you have to do to support automated security?
- What role does the connection pool play when connecting? How can you monitor the pool and determine how many connections or pools are created or hung? How can you tell when the pool is full, and how can you ensure that it does not overflow? How is a damaged pool flushed? How can you prevent connections from being orphaned? How does ADO.NET improve connection pool management?
- How do the ADO.NET DataReader, DataSet, and DataTable compare with the ADOC Recordset? Which is faster? Which uses fewer resources? Which scales better? What role do strongly typed *DataSet* objects play? What about the new ADO.NET 2.0 *TableAdapter*?

- How can data structures be passed back and forth between ADOc and ADO.NET? For example, can ADO.NET open an existing ADOc Recordset? Can an ADOc program open an ADO.NET DataReader or DataSet? How can you create an XML data structure and morph it so it can be opened by ADOc as an updatable Recordset?
- What's the best way to move data from tier to tier or between servers? How can does the new ADO.NET BCP class work and why is it so important?
- How do exception handlers differ from existing error handlers? What about the investment you have made in ON ERROR handlers? Why do you need exception handlers around Dim constructors? Why does ADO.NET throw an exception just because one of the connection string arguments is wrong? Why doesn't ADO.NET throw an exception like ADOc when a parameter is set to an incorrect value?
- Can ADO.NET actually generate SELECT and action queries for your application? How can the Visual Studio .NET tools help develop stored procedures, views and most database objects? What's missing from the Visual Studio .NET tools that forces developers to fall back on Enterprise Manager or other tools?
- How does ADO.NET manage and execute stored procedures? How does it handle parameters? Can parameters collections be automatically generated? Should they be? Can you use named parameters?
- How do you handle hierarchical resultsets? That is, how do you construct DataSets to deal with related DataTables? How do you update these data structures?
- How can you deal with Identity issues with ADO.NET DataSets? How does ADO.NET extract newly generated Identity values?
- How do you deal with complex multiple-resultset procedures? What about OUTPUT or Return Value parameters?
- How does ADO.NET deal with data set locking? Can you create a server-side cursor in ADO.NET? How about multi-user issues? Can you create a true multi-user application without server-side locking?
- Does data binding work any better in ADO.NET? What features don't have to be programmed because they're now supported by Visual Studio .NET bound controls? How well do bound controls work with updatable DataSets? Can you bind to a DataReader?
- How do you debug an ADO.NET application? What's needed to enable TSQL debugging? How can Visual Studio .NET help you create SQL queries and debug them? How has SQL Server SP3 affected TSQL debugging?
- What factors improve performance—both code and developer performance? That is, how can you develop faster code faster?

ADO.NET WORKSHOP (HALF-DAY)

4 Hours

This workshop is designed for conference presentations where a full-day workshop slot is not available. It is a condensed version of the full-day workshop with fewer demonstrations. Many of the same topics are covered, but in less detail.

COM-based ADO classic (ADOC) developers world-wide are ramping up to learn ADO.NET, as well as how to best port existing ADOC code to run in the .NET Framework. This class is designed with the ADOC developer in mind. It ensures that developers will get a running start at ADO.NET and become productive far faster than if they simply try to learn how to code within the new .NET technology on their own. By taking an entire day to discuss ADO.NET, the instructor can afford to spend more time discussing specific details of the technology.

This course discusses and demonstrates the following topics:

- What is ADO.NET? How is it different from ADOC?
- How does the ADO.NET DataReader, DataSet, and DataTable compare with the ADOC Recordset?
- What's the difference between the various .NET Data Providers? Does Microsoft support Oracle, DB2, Sybase, and other non-Microsoft database platforms?
- How does your code connect to a .NET Data Provider? How does this approach differ in the various .NET architectures? Do existing ADOC or ODBC connection strings still work in ADO.NET? How can you get Visual Studio .NET to generate these automatically?
- How do exception handlers differ from existing error handlers? What about the investment you have made in ON ERROR handlers?
- Can ADO.NET actually generate SELECT and action queries for your application? How can the Visual Studio .NET tools help develop stored procedures, views and most database objects?
- How does ADO.NET manage and execute stored procedures? How does it handle parameters? Can parameters collections be automatically generated? Should they be? Can you use named parameters?
- How does ADO.NET deal with DataSet locking? Can you create a server-side cursor in ADO.NET? How about multi-user issues? Can you create a true multi-user application without server-side locking?

VISUAL STUDIO .NET FOR ADO CLASSIC DEVELOPERS

60-90 minutes

Microsoft started work on its new integrated language-independent development environment long before the .NET version arrived in 2002. Visual Studio .NET is the culmination of work that saw the first light of day with Visual InterDev in the Visual Studio 6.0 timeframe. Regardless of the language developers use, or the architecture they're using, they'll find Visual Studio .NET up to the task. However, without a smart overview, many of the new features might go untapped for some time. This session is designed for data access developers wishing to learn how to port their skills to Visual Studio .NET.

This course discusses and demonstrates the following topics:

- How to use the Visual Studio .NET integrated development environment (IDE) to create data access applications, components, and services.
- How to establish Data Connections using the Server Explorer. How to customize the Server Explorer to incorporate the Odbc, Oracle, or any third-party .NET Data Provider.
- How to incorporate and register “installed” .NET Data Providers. Is this still necessary?
- How to use the “drag and drop” paradigm to build Connections, DataAdapters, stored procedures prototypes and XSL strongly typed DataSets?
- How to construct queries and stored procedures with the integrated Query Builder. We'll build stored procedures from scratch without having to code a single line of SQL—and know it's right.
- How to set breakpoints in stored procedures and walk right into stored procedures code—just as any other logic. How has SQL Server SP3 changed configuration?
- How to customize the IDE to show just the windows you need to see—and no more. How to put it back when it gets to be a jumbled mess.
- How to get IntelliSense to reduce the amount of typing (and errors) when working with multi-layered namespace objects.
- How to import and convert an existing Visual Basic 6.0 ADO “classic” (ADOC) application using Visual Studio .NET. We'll see which ADOC constructs in Visual Basic 6.0 work in Visual Basic .NET and which don't. We'll also discuss the conversion report and how to tell the difference between cosmetic issues, compile issues, and hidden issues.

ADO.NET FOR ADO CLASSIC DEVELOPERS

60-90 minutes

This course provides an overview of the issues that COM-based ADO “classic” (ADOC) developers face; it summarizes the day-long ADO.NET seminar. Some developers (and their managers) actually think that Microsoft has provided a conversion wizard that makes “minor changes” to existing code and automatically morphs it to ADO.NET. While it’s true that there is a Visual Basic conversion wizard, it simply recodes the ADOC code to run almost as-is in a Visual Basic .NET application. It makes no attempt to convert ADOC to ADO.NET. That’s because ADO.NET is an entirely new data access interface. It uses new data providers in a new way. Its created object model and the data structures are only vaguely similar to those created by ADOC. The objects, methods, properties, and events are far more different from RDO when compared with DAO, or ADOC when compared with either DAO or RDO. No, ADO.NET is not necessarily more difficult to code; in fact, it’s really easier in some respects. It is, however, very different from any data access interface that developers have been faced with up to this point. More importantly, there are significant functional aspects that are entirely new—new ways to fetch data and persist it on the client. Without some insight as to these differences and innovations, ADOC developers will spend far too much time trying to get round pegs to fit in the new trapezoidal holes.

This course discusses and demonstrates the following topics:

- The ADO.NET object model—comparing and contrasting it with ADOC. What objects are most like ADO.NET and which appear to be, but aren’t.
- Problem solving with ADO.NET. How do you solve basic and complex data access problems with ADO.NET? How are these approaches different from those used in ADOC?
- How to safely and consistently open a Connection object—regardless of the .NET Data Provider. How to build a workable, scalable connection string that won’t overflow the connection pool.
- How to create ADO.NET Command objects, construct their parameters collection, and execute ad-hoc or stored procedures. How to use the Add method constructors to reduce the complexity (and bulk) of these tasks. How to get ADO.NET or Visual Studio .NET to create these for you.
- How to create ADO.NET updatable DataSets. How to use stored procedures to perform the updates and how to debug these stored procedures as if they were Visual Basic code.
- How ADO.NET exception handlers differ from those written for ADOC. Why must you ensure that Dim statements using constructors must be included within the error handling scope?
- How to import a Visual Basic 6.0 ADOC code block to Visual Basic .NET. How to decide when it’s best to import versus recode using the new architectures.
- What can be done in a Visual Basic 6.0 ADOC application to permit the code to more easily transport to Visual Basic .NET?

MANAGING SERVER-SIDE CURORS AND LOCKS WITH ADO.NET

60-90 minutes

For over a decade now SQL Server and other DBMS developers have been using server-side cursors to access their databases and scroll through updatable rowsets. In addition, some applications are better served by a “connected-pessimistic concurrency” data access paradigm. For a litany of reasons, Microsoft chose not to implement server-side cursors or “traditional” pessimistic locking cursors in their new .NET Framework data access interface ADO.NET. This session discusses how you can work around these limitations. Even though they said it couldn’t be done, we’ll show you how to create and manage your own server-side cursors and pessimistic locks. We’ll show how to open a cursor, fetch rows serially or randomly or in bulk and how to use the cursor to update the data. We’ll also demonstrate how to lock a specific row, page or table and hold that lock until you release it.

This course discusses and demonstrates the following topics:

- How to establish a dedicated connection used to manage a server-side cursor.
- Opening the cursor using a standard SQL SELECT statement.
- Fetching individual rows based on their position or ordinal in the cursor.
- Fetching all rows in the cursor in a single operation using batch queries.
- Updating rows in the cursor.
- Setting up a transaction to manage a pessimistic lock.
- Understanding the performance, scalability and concurrency issues associated with pessimistic locks.
- Accessing a pessimistically locked row, page or table.
- Understanding how other applications react to the locked row(s).
- Releasing the pessimistic lock using watchdog timers and code-driven techniques.