IBM Systems Group

P15 - Microsoft .NET Integration with DB2 UDB for iSeries

Jarek Miszczyk
PartnerWorld for Developers, Rochester
Special thanks extended to Brent Nelson, Rochester Development

eServer iSeries ASEAN Tech Summit
November 2003

.NET Architecture and Data Access

- .NET is the latest Microsoft environment for running code
  - code runs inside .NET execution runtime
  - .NET runtime provides memory management, garbage collection, versioning, etc.

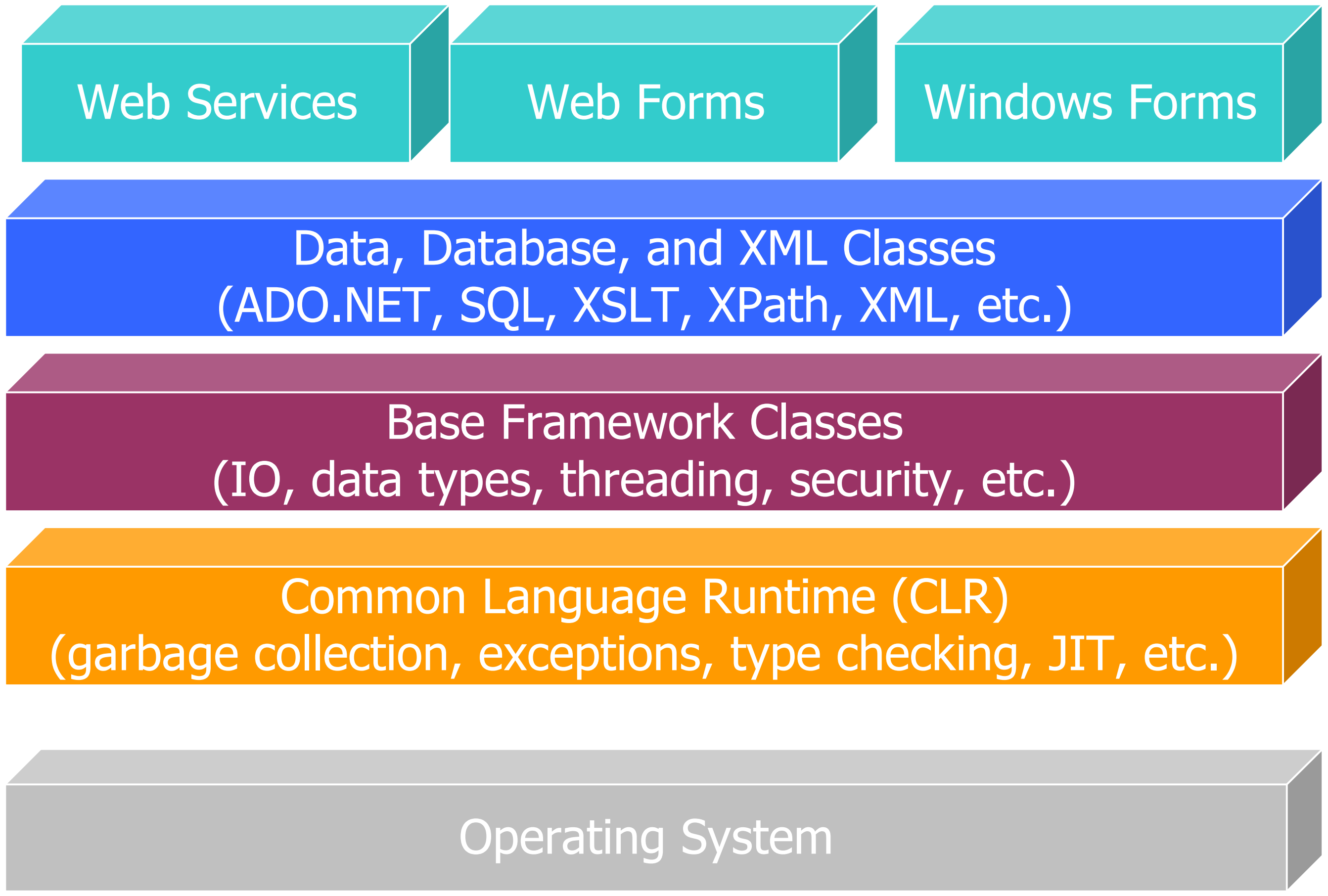
.NET Glossary

- NET framework
  - Underlying plumbing for .NET applications
  - Common Language Runtime (CLR)
  - Unified set of class libraries
- CLR
  - Language integration, security handling, memory/ thread/process management, exception handling,
- Managed code
  - code that is compiled into a .NET assembly that can be executed in the context of .NET's CLR.
- ADO.NET
  - .NET classes enabling access to databases
- ASP.NET
  - .NET classes to support development of Web-based applications and Web services

Common Language Runtime (CLR)

- Similar to Java Virtual Machine (JVM)
- Object activation, security, code execution, and garbage collection
- Contains JIT for just-in-time compilation
- Available for Windows 98 and newer
- Shipped with Windows 2003 Server
- 64-bit version not yet available
- CLI – Subset of CLR submitted to ECMA

.NET Framework Components


Web Services
Web Forms
Windows Forms
Data, Database, and XML Classes
(ADO.NET, SQL, XSLT, XPath, XML, etc.)
Base Framework Classes
(IO, data types, threading, security, etc.)
Common Language Runtime (CLR)
(garbage collection, exceptions, type checking, JIT, etc.)
Operating System

.NET versus J2EE

- Multiple languages (>25?)
- Easily call existing native code/components
- 1 IL
- 1 platform today (Windows)
- Dynamic web: ASP.NET
- Database access: ADO.NET

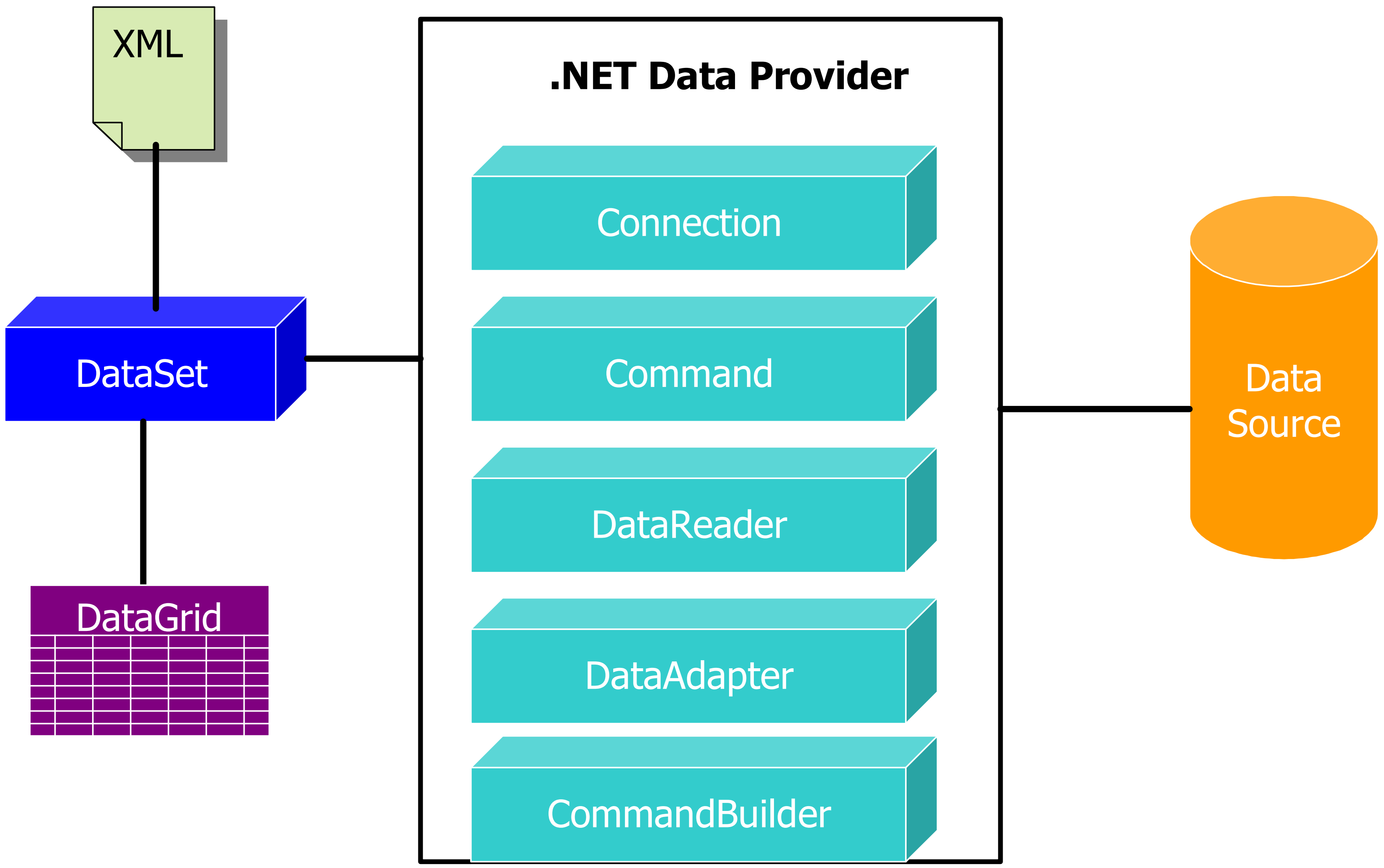
- 1 language
- JNI/System calls for native components
- 1 IL
- Multiple platforms (JVM)
- Dynamic web: JSPs
- Database access: JDBC SQL/J

What is ADO.NET

- An evolutionary improvement to Microsoft ActiveX® Data Objects (ADO) programming interface
- Set of classes that .NET applications can use to access data on a database
- Built for “disconnected” record sets
- Data provider is used for connecting to a database, executing commands, and retrieving results
  - Database provider specific (DB2, SQL Server, Oracle, etc.)
  - Providers implement vague interfaces in System.Data (IDbConnection, etc.)
- MS Ships SQL Server, Oracle (with 1.1 Framework), and OleDb and ODBC bridge providers

IBM

ADO.NET Object Model



The .NET Provider Classes

- Connection - for example, OleDbConnection
  - Transaction - for example, OleDbTransaction
  - Exception - for example, OleDbException
  - Error - for example, OleDbError
- Command - for example, OleDbCommand
  - Parameter - for example, OleDbParameter
- DataReader - for example, OleDbDataReader
- DataAdapter - for example, OleDbDataAdapter

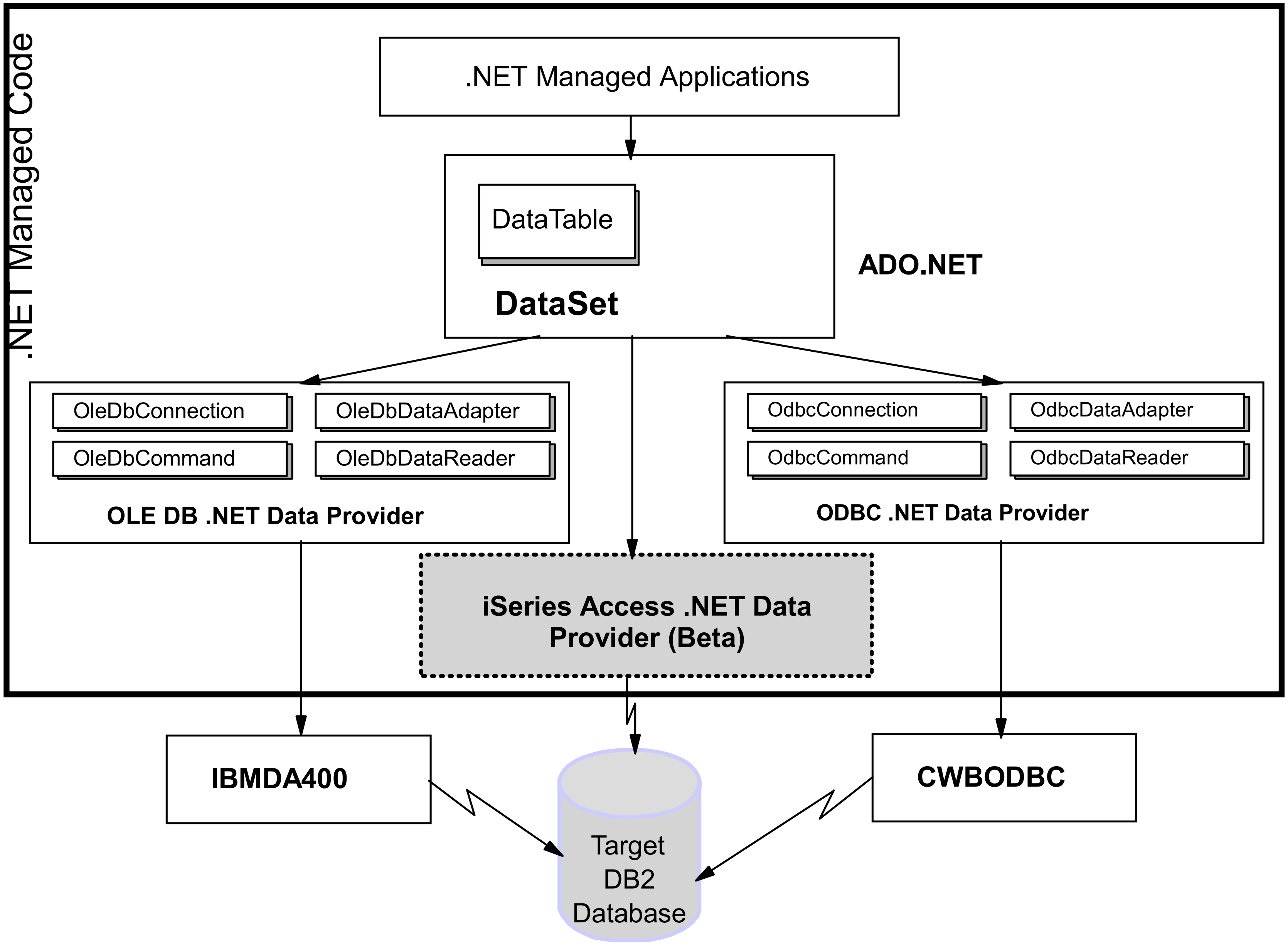
Providers contained in .NET Framework v1.1

- SQL Server
  - System.Data.SqlClient
- Oracle
  - System.Data.OracleClient
- Bridge to ODBC drivers
  - System.Data.Odbc
- Bridge to OLE DB providers
  - System.Data.OleDb

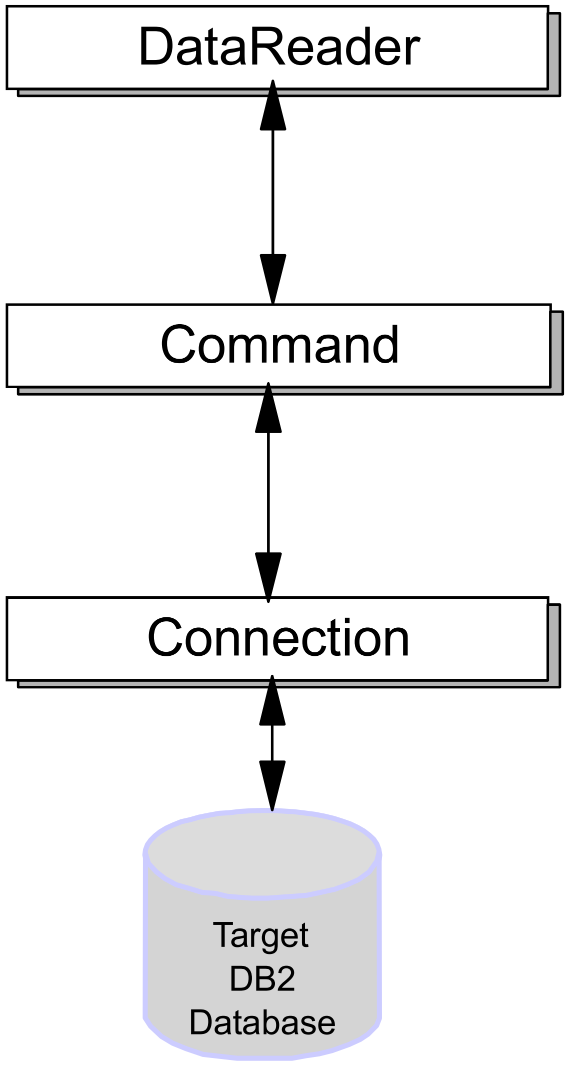
Providers to access DB2 UDB for iSeries

- iSeries Access .NET provider (Beta)
  - **Available as of October 31, 2003!**
- DB2 for .NET provider (Technology Preview)
  - managed .NET provider for DB2 LUW
  - IBM.Data.DB2
  - iSeries officially supported with the latest Technology Preview
- Microsoft.Data.Odbc bridge to iSeries Access
- System.Data.OleDb bridge to iSeries Access

iSeries Access from .NET



ADO.NET in a Connected Scenario



- Resources are held on the server as long as the connection is open
- The DataReader class provides forward-only, read-only access to data in a data source

Connection Object

- Connection string provides parameters required to establish a connection
  - Provider (OLE DB only)
  - Data Source
  - Initial Catalog
  - User ID/Password
- Opening and closing connection explicitly
  - Open and Close methods on the Connection object
- Opening and closing connection implicitly
  - Data Adapters open and close connections automatically

Example:

```
Dim dbConn As OleDbConnection = New _
            OleDbConnection("Provider=IBMDA400.1;Data Source=FORUM01;" & _
            "User ID=db2user;Password=db2user;Default Collection=SAMPLEDB")
```

Command Object

- A reference to a SQL statement or stored procedure
- Properties
  - Connection, CommandType, CommandText,Parameters
- Methods
  - ExecuteScalar, ExecuteReader, ExecuteNonQuery

Example:

```
Dim catCMD As OleDbCommand = dbConn.CreateCommand()
catCMD.CommandText = "SELECT Name FROM staff"
dbConn.Open()
Dim myReader As OleDbDataReader = catCMD.ExecuteReader()
```

Cammand Parameters

- Parametrized SQL statements and stored procedures may require parameters
  - Command object has a collection of Parameters
- Command parameter objects allow you to set and retrieve parameters associated with a command object
- Properties
  - ParameterName, DbType, Size, Direction

Example:

```
Dim cmStaff As New System.Data.OleDb.OleDbCommand
Dim cniSeries As New System.Data.OleDb.OleDbConnection
Dim prmJob As New System.Data.OleDb.OleDbParameter("JOB", _
System.Data.OleDb.OleDbType.VarChar, 5, "JOB")

cmStaff.CommandText = "SELECT NAME FROM STAFF WHERE (JOB = ?)"
cmStaff.Connection = cniSeries

prmJob.Direction = System.Data.ParameterDirection.Input
cmStaff.Parameters.Add(prmJob)
```

DataReader Object

- Read-only, forward-only, stream of rows
- Properties
  - Item
    - gets the value of a column with a specific name or ordinal position
- Methods
  - GetXxx methods
    - return Common Language Specification (CLS) data types such as String, Int32, Double
  - GetValues
    - returns an array of objects representing all column values for the current row
  - IsDbNull, Close
  - GetName, GetOrdinal, GetSchemaTable
    - return metadata for the result set

Example:

```
Dim rdrStaff As OleDb.OleDbDataReader
rdrStaff = cmStaff.ExecuteReader()
Do While rdrStaff.Read()
   ListBox1.Items.Add(rdrStaff.GetString(0))
Loop
rdrStaff.Close()
```

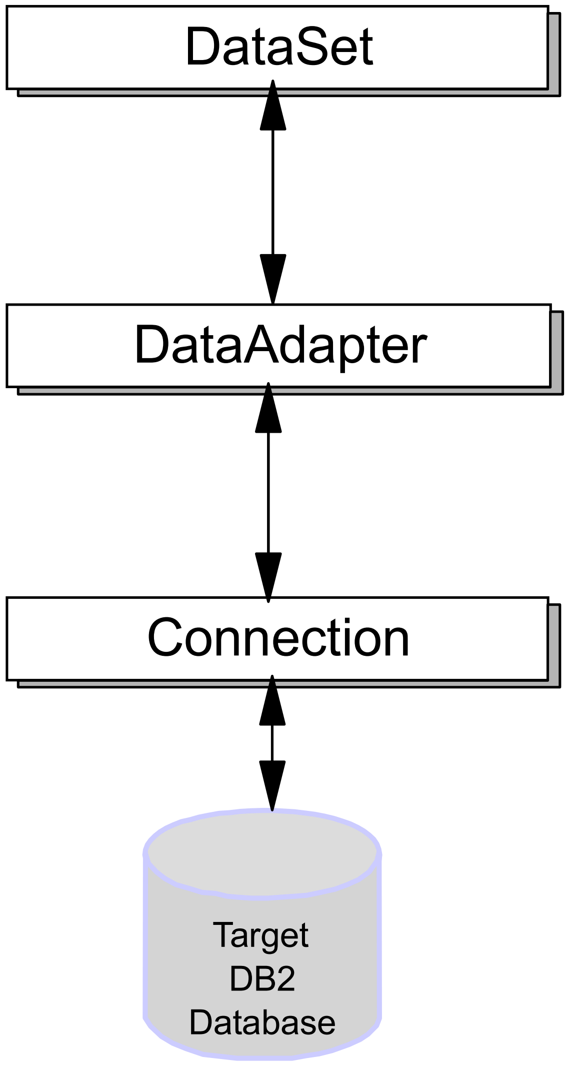
OLE DB .NET Provider example

```
Imports System
Imports System.Data
Imports System.Data.OleDb
Imports Microsoft.VisualBasic

Public Class Sample
  Public Shared Sub Main()
 [1]Dim dbConn As OleDbConnection = New _
            OleDbConnection("Provider=IBMDA400.1;Data Source=FORUM01;" & _
            "User ID=db2user;Password=db2user;Default Collection=SAMPLEDB")
 [2]Dim catCMD As OleDbCommand = dbConn.CreateCommand()
    catCMD.CommandText = "SELECT Name FROM staff"
    dbConn.Open()
 [3]Dim myReader As OleDbDataReader = catCMD.ExecuteReader()
   Do While myReader.Read()
      Console.WriteLine(vbTab & "{0}", myReader.GetString(0))

   Loop
    myReader.Close()
    dbConn.Close()
  End Sub
End Class
```

ADO.NET in a Disconnected Scenario

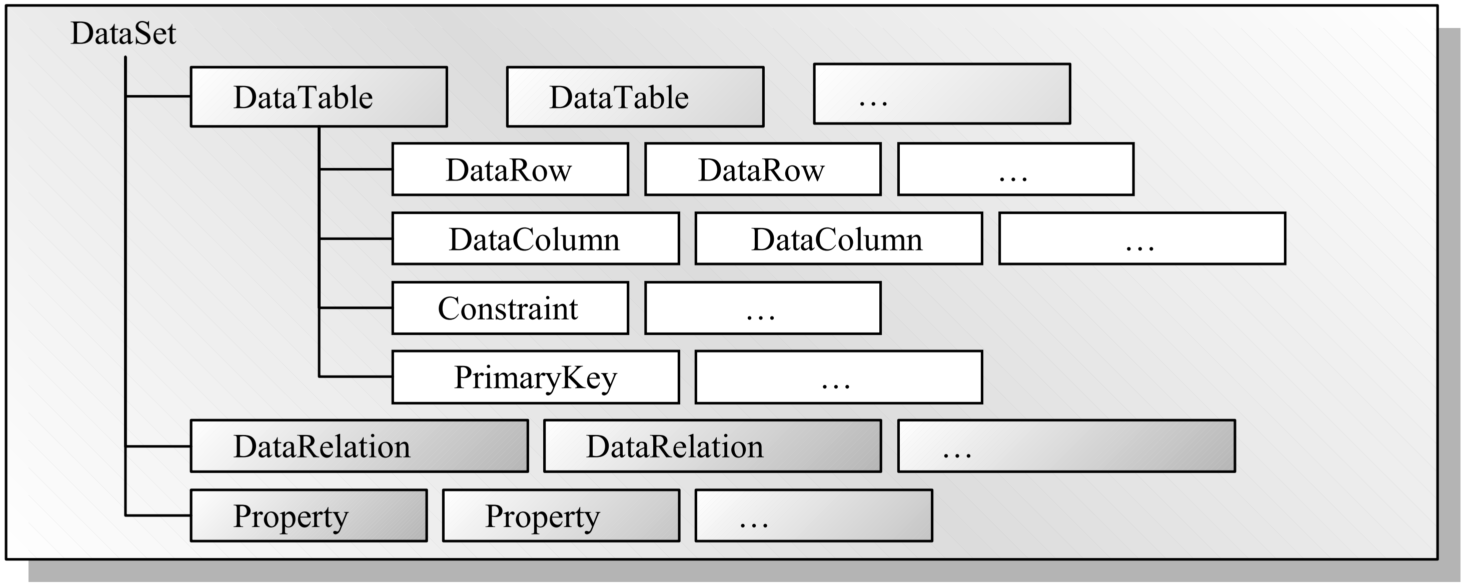


- Resources are not held on the server while the data is processed
- DataAdapter used to implicitly populate a DataSet and to update the central data store with changes made to the DataSet

DataSet Object

- corner stone of ADO.NET's disconnected architecture
  - provides a disconnected cache of data from any source
- structure follows the relational data model
  - looks like in-memory database
  - contains a collection of DataTables
    - DataTables contain DataColums and DataRows
    - DataTables can have primary key, foreign keys, relationships, and unique constraints
  - supports computed columns
- is NOT a relational database though
  - supports just a subset of SQL-like syntax
  - no transactions or locking
  - CLR data types that are not compatible with SQL-99 standard

DataSet Object Model



- DataSet - High-level container class that holds collections of other objects
- DataTable - Table with columns and rows
- DataRow - Relational tuple
- DataColumn - Ralational columns
- Constraint - Data constraints to ensure table and inter-table consistency
- DataRelation - Relationl between two tables to ensure data consistency
- Property - user-defined, extended property

How to create DataSets and DataTables?

- Programmatically
- By using the graphical tools in Visual Studio .NET
- By using DataAdapter and filling the DataSet from a relational data source
- By loading and storing DataSet contents using XML

Building DataSet pragrammatically

Add DataTable to DataSet programmatically

```
Dim dsSample As New DataSet("Sample")
Dim dtStaff As DataTable = dsSample.Tables.Add("Staff")
```

Creating DataColumns programmatically

```
Dim colID As DataColumn = dtStaff.Columns.Add("ID", GetType(System.Int16))
colID.AllowDBNull = False
```

Setting the PrimaryKey property of a DataTable

```
dtStaff.PrimaryKey = New DataColumn() {dtStaff.Columns("ID")}
```

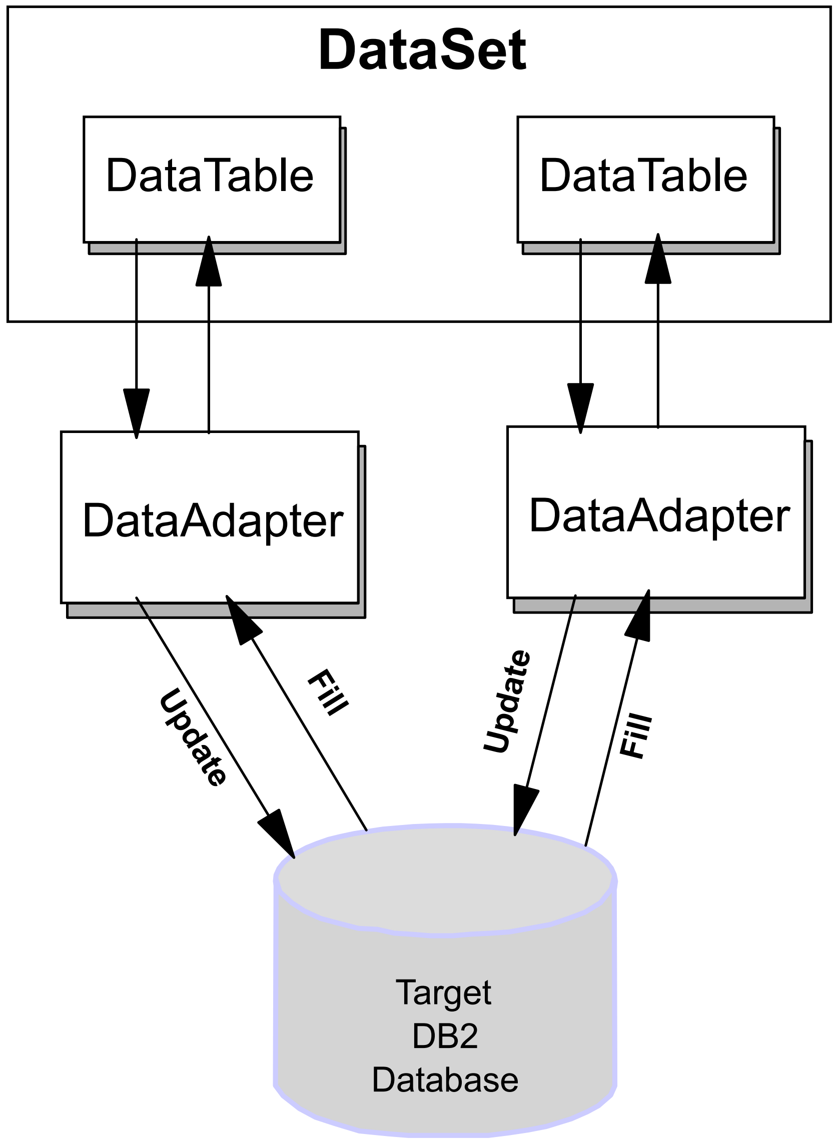
Creating a new row

```
Dim drNewStaff As DataRow = dtStaff.NewRow
drNewStaff("ID") = 99
drNewStaff("Name") = "Novak"
dtStaff.Rows.Add(drNewStaff)
```

Saving a DataSet using XML

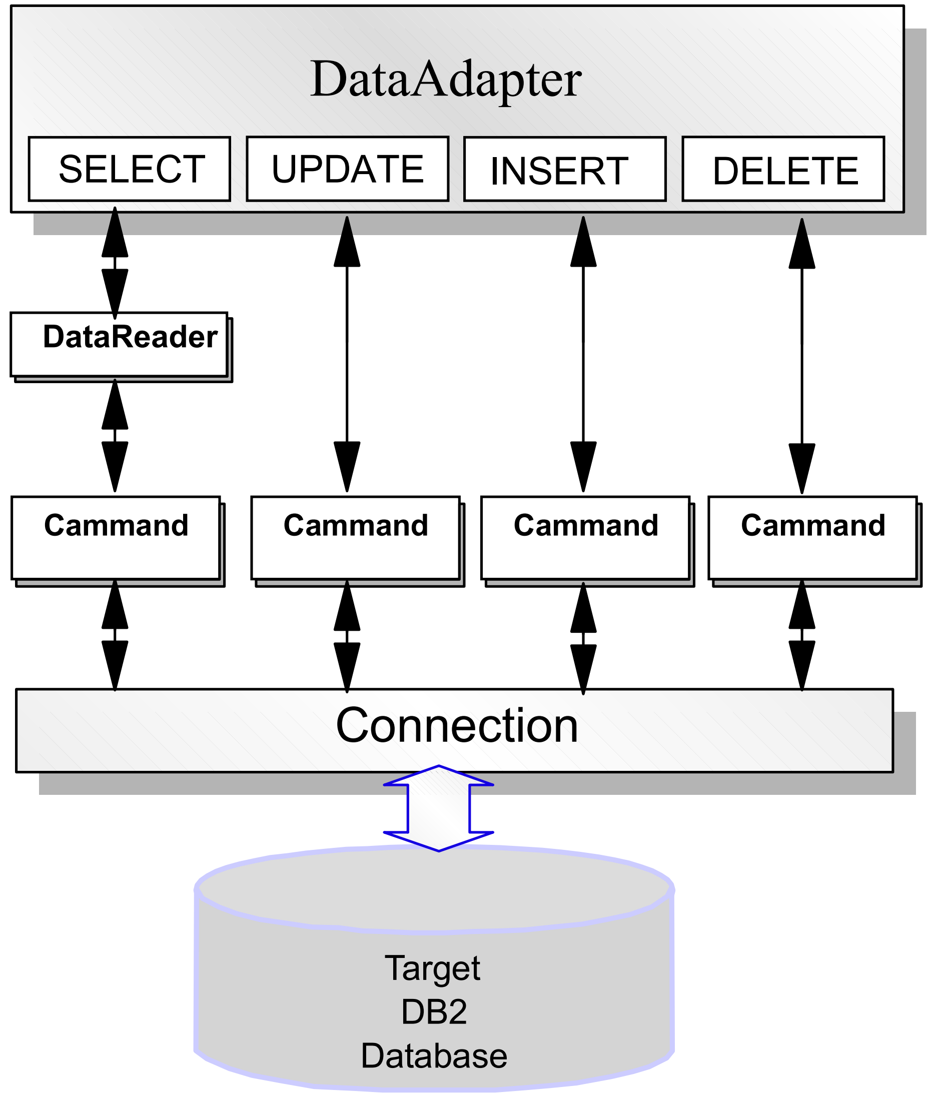
```
dsStaff.WriteXml("\My Documents\Staff.ds")
```

DataAdapter Object



- DataAdapter is a bridge between a DataSet and a data source
- It encapsulates database connection and database commands used to retrieve and save data
- It exchanges data between a single DataTable and a single result set

DataAdapter Object Model



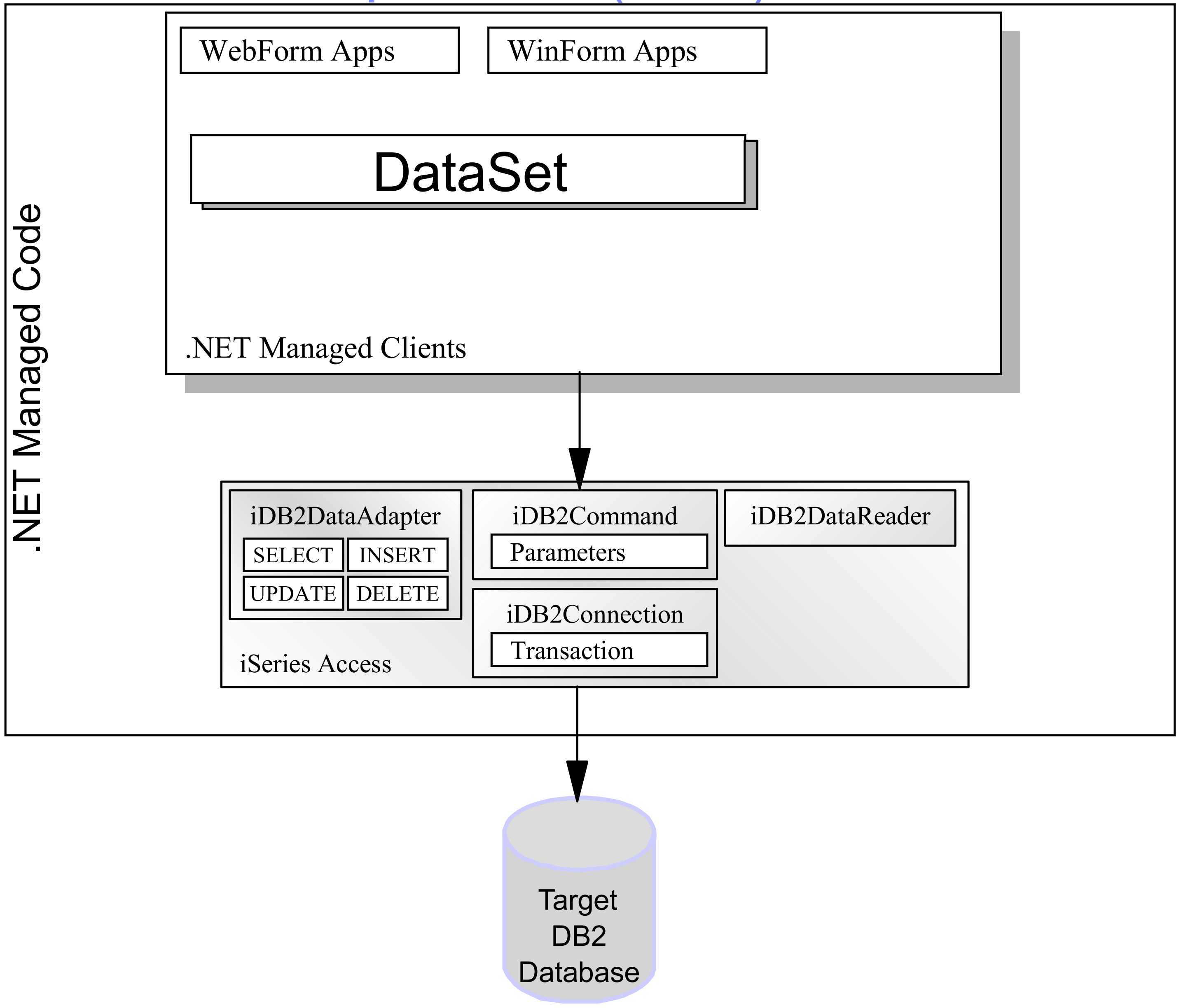
- Properities
  - SelectCommand
  - UpdateCommand
  - InsertCommand
  - DeleteCommand
- Methods
  - Fill
  - Update

ODBC .NET Provider example

```
 Private Sub Form1_Load(ByVal sender As Object, ByVal e As System.EventArgs) _
             Handles MyBase.Load
   Dim strConnectionString As String
   strConnectionString = "DSN=iSeries;UID=db2user;PWD=db2user;DBQ=SAMPLEDB"
[1]cniSeries = New OdbcConnection(strConnectionString)
   daStaff = New OdbcDataAdapter
   cmdDelete = New OdbcCommand
   cmdInsert = New OdbcCommand
   cmdSelect = New OdbcCommand
   cmdUpdate = New OdbcCommand
   dsSample = New DataSet

[2]daStaff.DeleteCommand = cmdDelete
   daStaff.InsertCommand = cmdInsert
   daStaff.SelectCommand = cmdSelect
   daStaff.UpdateCommand = cmdUpdate
   ...
   cmdSelect.CommandText = "SELECT ID, NAME, DEPT, JOB, ""YEARS"", SALARY, COMM FROM STAFF"
   cmdSelect.Connection = cniSeries
[3]cmdDelete.CommandText = "DELETE FROM STAFF WHERE (ID = ?)"
   cmdDelete.Connection = cniSeries
   cmdDelete.Parameters.Add(New OdbcParameter("Original_ID", OdbcType.SmallInt, 0, _
      System.Data.ParameterDirection.Input, False, CType(0, Byte), CType(0, Byte), _
      "ID", System.Data.DataRowVersion.Original, Nothing))
[4]dsSample.Locale = New System.Globalization.CultureInfo("en-US")
 End Sub
```

iSeries Access .NET provider (Beta)



iSeries Access .NET notes

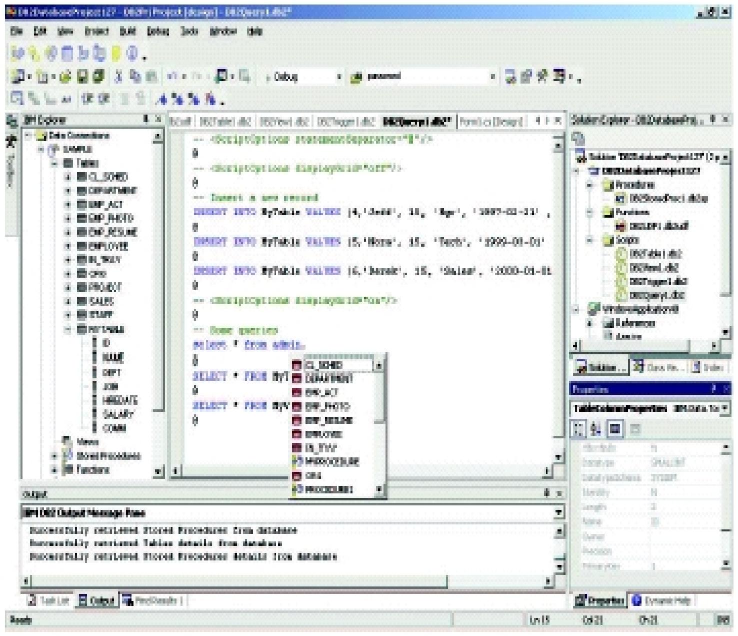
- iSeries Access .NET provider (Beta)
  - Details available at:
    http://www-1.ibm.com/servers/eserver/iseries/access/
    Beta available October 31, 2003
- Install requires the .NET framework be on PC
  - Windows Server 2003 installs .NET framework by default
- Same basic requirements as iSeries Access ODBC and OLE DB to use
  - Database host server must be up and running
- Limited support on pre-V5R2 servers

What is in the Beta?

- Supported
  - SQL (INSERT,UPDATE, DELETE)
  - Commitment Control
  - Connection Pooling
  - SQL naming
  - Unicode
  - Tracing
  - Threads
  - IASPs (multiple databases)
  - Compression
  - Stored procedure support
- Not supported
  - Large Objects (LOBs)
  - System naming (/)
  - Package support
  - Data links
  - User Defined Types
  - Record Level Access
  - CMD/PGM call
  - Data Queues

.NET Framework Integration with DB2 UDB

- DB2 Connect V8 includes ADO.NET Managed Provider
- DB2 Add-ins for Visual Studio .NET
  - Integrate DB2 into Microsoft Development Tooling
  - Ease Creation of
    - Stored Procedures
    - Triggers
    - User Defined Functions



```
-- <ScriptOptions statementSeparator="@"/>
@
-- <ScriptOptions displayXml="off"/>
@
-- Insert a new record
INSERT INTO MyTable VALUES (4,'Jedi', 10, 'Mgr', '1997-02-11' ,
@
INSERT INTO MyTable VALUES (5,'Nora', 15, 'Tech', '1999-01-01'
@
INSERT INTO MyTable VALUES (6,'Derek', 15, 'Sales', '2000-01-01
@
-- <ScriptOptions displayXml="on"/>
@
-- Some queries
select * from admin.
@
SELECT * FROM MyT
@
SELECT * FROM MyV
@
```

```
Successfully retrieved Stored Procedures from database
Successfully retrieved Tables details from database
Successfully retrieved Stored Procedures details from database
```

DB2 Application Development Technology Preview!

- Improved DB2 .NET Data Provider with better performance and support for DB2 UDB for iSeries servers
- Enhanced DB2 add-ins for Microsoft Visual Studio. NET
- Support for building and deploying stored procedures and user defined functions using Visual Basic .NET, C# and other CLR programming languages
- Support for creating and deploying SQL/PL stored procedures and functions
- To participate in this technology preview program visit: *https://www6.software.ibm.com/reg/dm/dm-adtpapp-i*

DB2 .Net Provider Example

```
   Imports IBM.Data.DB2
   ...
   Dim strConnectionString As String
   strConnectionString = "database=MYiSERIES;user Id=db2user;password=db2user;"
[1]db2cniSeries = New IBM.Data.DB2.DB2Connection(strConnectionString)
   db2daDummy = New IBM.Data.DB2.DB2DataAdapter
   db2cmdSelect = New IBM.Data.DB2.DB2Command
   db2cmdDelete = New IBM.Data.DB2.DB2Command
[2]daDummy.DeleteCommand = db2cmdDelete
   daDummy.SelectCommand = db2cmdSelect
      ...
   db2cmdSelect.CommandText = "SELECT ID, COL2 FROM DUMMY"
   db2cmdSelect.Connection = db2cniSeries
[3]db2cmdDelete.CommandText = "DELETE FROM DUMMY WHERE  (ID = ?) "
        db2cmdDelete.Connection = db2cniSeries
        db2cmdDelete.Parameters.Add(New IBM.Data.DB2.DB2Parameter("Original_ID", _
        IBM.Data.DB2.DB2Type.SmallInt, 1, System.Data.ParameterDirection.Input, _
        False, CType(0, Byte), CType(0, Byte), "ID", _
        System.Data.DataRowVersion.Original, Nothing))
[4]dsSample.Locale = New System.Globalization.CultureInfo("en-US")
 End Sub
```

Trademarks and Disclaimers