ADO.NET Programming (code ADO.Net)

Part I Fundamentals and Theory

Overview
- What is a database?
- Evolution of databases
- Data and file structure
- Data architecture and data independence
- Relational Data Structure
- Relations
- Domains and attributes
- Keys
- Integrity rules
- Extensions and intensions

Structured Query Language (SQL) Overview
- Data Manipulation Language (DML)
- Data Definition Language (DDL)
- Tuples: free and bound variables
- Normalisation (third normal form)

SQL Statements and Clauses
- SELECT statement
- Aggregate functions
- WHERE, ORDER BY and LIKE clauses
- INSERT, UPDATE and DELETE statements
- Joining tables

Data Definition Language (DDL)
- Creating and dropping databases
- Creating and altering tables
- Creating views
- Stored procedures and parameterised queries

Specific Data Sources
- SQL Server
- Microsoft Access and Microsoft Excel
- MySQL
- DB2 and Oracle CSV and text files

Data Providers
- SQL Server
- OLE DB
- ODBC
- Other providers

Part II Core Functionality
Overview of ADO.NET
- Goals of ADO.NET
- Connected and disconnected classes
- Security
- Queries

Connecting to Data
- Connection strings
- Building a connection string
- Connecting to various data providers Advanced

Connection Issues
- Changing the database for an open connection
- Connection pooling Transactions and connection pooling
- Monitoring connections

Other Connected Classes
- Command
- DataReader
- DataAdapter
- Examples and applications

Disconnected Data Objects
- Dataset
- DataTable and DataColumn
- DataRow
- DataView
- DataRelation
- Constraint

Using Disconnected Data Objects: Some Examples
- Mapping between data source and data set
- Working with keys
- Building a data set programmatically
- Converting between a data table and a data row array
- Accessing data values in a data row array
- Persisting disconnected data

Using ADO.NET Features
- CommandBuilder
- Querying and Retrieving Data
- Searching and Analysing Data
- Adding and Modifying Data
- Copying and Transferring Data

Part III Advanced Methods
Concurrency
- Pessimistic concurrency
- Optimistic concurrency

- Update strategies: Last in wins, all columns or timestamps
- Detecting concurrency errors
- Resolving update conflicts

Validation and Constraints
- Column level constraints: Read only, allow null, max length and unique constraints
- Table level constraints: Foreign keys and unique values
- Expression based columns
- Auto increment columns and retrieving generated number
- Submitting hierarchical changes

Transactions
- Connections and transactions
- Beginning transactions
- Commit transactions
- Rollback transactions
- Transactions with data adapters and command builders

Strongly Typed DataSets
- Advantages of strongly typed data sets
- Creating strongly typed data sets
- Creating XML schemas oUsing strongly typed data sets
- Adding rows
- Working with null data
- Hierarchical data
- Strongly typed data sets in Visual Studio

Part IV Optimisation, Design and Applications
ADO.NET and Multithreading
- Using multiple threads
- Asynchronous processing
- Multiple active result sets

Optimisation Techniques
- Moving large amounts of data
- Paging
- Minimising roundtrips and conversions
- Debugging and optimising
- Database independence

Some Scenarios
- Executing a SQL statement asynchronously
- Executing multiple commands on a single connection
- Executing simultaneously SQL statements asynchronously C
- reating a DataReader asynchronously
- Filling a DataSet asynchronously
- Caching data
- Improving paging performance

Languages
- C# versus Managed C++
- Excel integration
- Native C++ interoperability

ADO.NET and XML
- Overview and objectives
- Loading and saving DataSets and data tables using an XML file
- Load and save a DataSet structure using XSD schema file
- XML file with changes in a DataSet
- Synchronising a DataSet and an XML Document

Language Integrated Query (LINQ)
- Overview and rationale
- LINQ queries and operators
- Query syntax versus SQL syntax
- ADO.NET and LINQ

ADO.NET Entity Framework
- Architecture
- Entity framework stack
- Entity Data Model (EDM)
- Schema definition language
- Querying data
- ADO.NET providers

Your Trainer

Daniel J. Duffy started the company Datasim in 1987 to promote C++ as a new object-oriented language for developing applications in the roles of developer, architect and requirements analyst to help clients design and analyse software systems for Computer Aided Design (CAD), process control and hardware-software systems, logistics, holography (optical technology) and computational finance. He used a combination of top-down functional decomposition and bottom-up object-oriented programming techniques to create stable and extendible applications (for a discussion, see Duffy 2004 where we have grouped applications into domain categories). Previous to Datasim he worked on engineering applications in oil and gas and semiconductor industries using a range of numerical methods (for example, the finite element method (FEM)) on mainframe and mini-computers. Daniel Duffy has BA (Mod), MSc and PhD degrees in pure and applied mathematics and has been active in promoting partial differential equation (PDE) and finite difference methods (FDM) for applications in computational finance. He was responsible for the introduction of the Fractional Step (Soviet Splitting) method and the Alternating Direction Explicit (ADE) method in computational finance. He is also the originator of the exponential fitting method for time-dependent partial differential equations.

He is also the originator of two very popular C++ online courses (both C++98 and C++11/14) on www.quantnet.com in cooperation with Quantnet LLC and Baruch College (CUNY), NYC. He also trains developers and designers around the world. He can be contacted dduffy@datasim.nl for queries, information and course venues, in-company course and course dates