DOI : https://dx.doi.org/10.26808/rs.st.i7v6.17
International Journal of Advanced Scientific and Technical Research Issue 7 volume 6 Nov.- Dec. 2017
Available online on http://www.rspublication.com/ijst/index.html ISSN 2249-9954

A COMPARATIVE STUDY BETWEEN ADO.NET AND ENTITY FRAMEWORK

Manisha Wadhwa
India

ABSTRACT

This paper describes about data access technology from the Microsoft .NET Framework named ADO.NET and an open source object-relational mapping (ORM) framework for ADO.NET named Entity Framework and compares them in all aspects.

Key words:ADO.NET, EF, Data Provider, Data-Set, Data Reader, Data Store, Data Adapter

INTRODUCTION

ADO.NET is a data access technology from the Microsoft .NET Framework that provides communication between relational and non-relational systems through a common set of components.ADO.NET is a set of computer software components that programmers can use to access data and data services from a database. It is a part of the base class library that is included with the Microsoft .NET Framework. It is commonly used by programmers to access and modify data stored in relational database systems, though it can also access data in non-relational data sources. ADO.NET is sometimes considered an evolution of ActiveX Data Objects (ADO) technology, but was changed so extensively that it can be considered an entirely new product.ADO.NET provides a bridge between the front end controls and the back end database. The ADO.NET objects encapsulate all the data access operations and the controls interact with these objects to display data, thus hiding the details of movement of data.ADO stands for ActiveX Data Objects.ADO.NET is a database technology of .NET Framework used to connect application system and database server.ADO.NET is a part of the .NET Framework.ADO.NET consists of a set of classes used to handle data access.ADO.NET uses XML to store and transfer data among applications, which is not only an industry standard but also provide fast access of data for desktop and distributed applications.ADO.NET is scalable and interoperable.ADO.NET provides consistent access to data sources such as SQL Server and XML, and to data sources exposed through OLE DB and ODBC. Data-sharing consumer applications can use ADO.NET to connect to these data sources and retrieve, handle, and update the data that they contain.ADO.NET separates data access from data manipulation into discrete components that can be used separately or in tandem. ADO.NET includes .NET Framework data providers for connecting to a database, executing commands, and retrieving results. Those results are either processed directly, placed in an ADO.NET DataSet object in order to be exposed to the user in an ad hoc manner, combined with data from multiple sources, or passed between tiers. The DataSet object can also be used independently of a .NET Framework data provider to manage data local to the application or sourced from XML.The ADO.NET classes are found in System.Data.dll, and are integrated with the XML classes found in System.Xml.dll. ADO.NET provides functionality to developers who write managed code similar to the functionality provided to native component object model (COM)

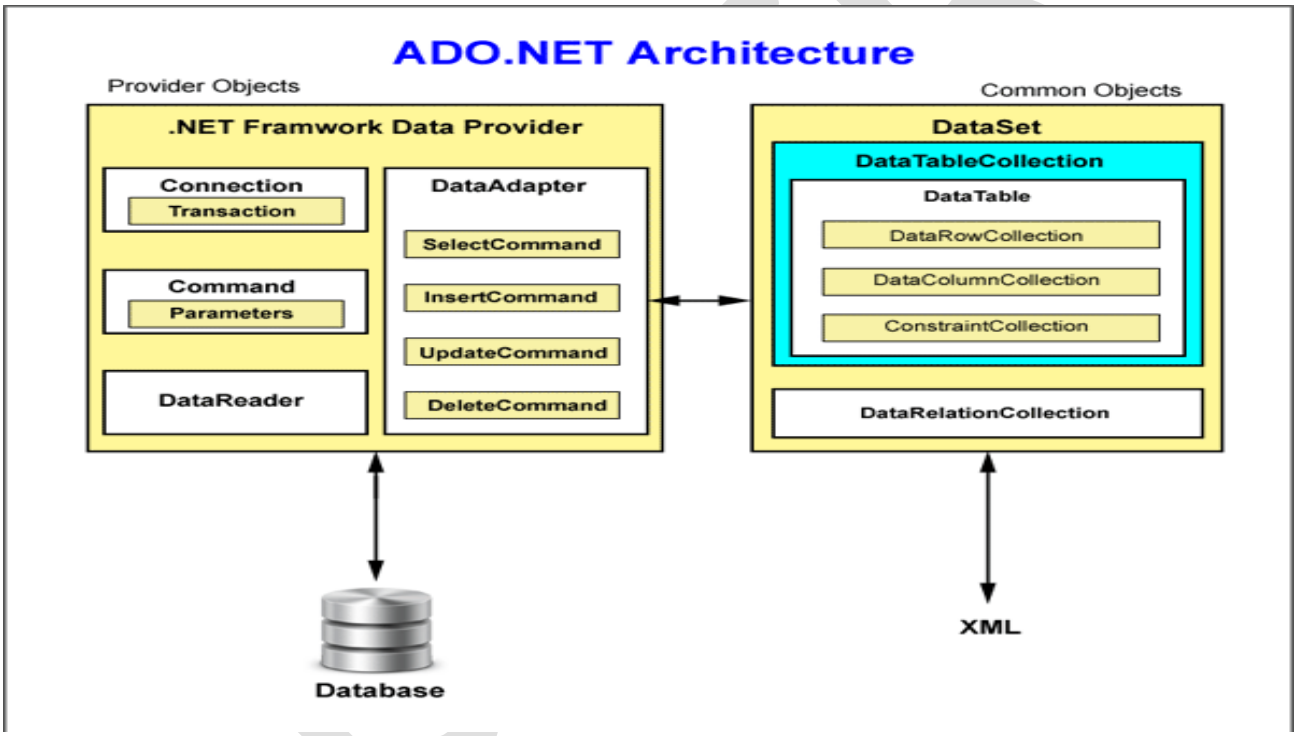
DOI : https://dx.doi.org/10.26808/rs.st.i7v6.17
International Journal of Advanced Scientific and Technical Research Issue 7 volume 6 Nov.- Dec. 2017
Available online on http://www.rspublication.com/ijst/index.html ISSN 2249-9954

developers by ActiveX Data Objects (ADO). We recommend that you use ADO.NET, not ADO, for accessing data in your .NET applications.ADO.NET provides the most direct method of data access within the .NET Framework.

ARCHITECTURE

ADO.NET is conceptually divided into consumers and data providers. The consumers are the applications that need access to the data, and the providers are the software components that implement the interface and thereby provide the data to the consumer. Functionality exists in Visual Studio IDE to create specialized subclasses of the Dataset classes for a particular database schema, allowing convenient access to each field in the schema through strongly typed properties. This helps catch more programming errors at compile-time and enhances the IDE's Intelligence feature.

The following figure shows the ADO.NET objects at a glance:



.

DATA PROVIDERS

We know that ADO.NET allows us to interact with different types of data sources and different types of databases. However, there isn't a single set of classes that allow you to accomplish this universally. Since different data sources expose different protocols, we need a way to communicate with the right data source using the right protocol Some older data sources use the ODBC protocol, many newer data sources use the OleDb protocol, and there are more data sources every day that allow you to communicate with them directly through .NET ADO.NET class libraries.ADO.NET provides a relatively common way to interact with data sources, but comes in different sets of libraries for each way you can talk to a data source. These libraries are called Data Providers and are usually named for the protocol or data source type they allow you to interact with. ADO.NET Objects

DOI : https://dx.doi.org/10.26808/rs.st.i7v6.17
International Journal of Advanced Scientific and Technical Research Issue 7 volume 6 Nov.- Dec. 2017
Available online on http://www.rspublication.com/ijst/index.html ISSN 2249-9954

ADO.NET includes many objects you can use to work with data. This section introduces some of the primary objects you will use. Over the course of this tutorial, you'll be exposed to many more ADO.NET objects from the perspective of how they are used in a particular lesson. The objects below are the ones you must know. Learning about them will give you an idea of the types of things you can do with data when using ADO.NET.

THE SQLCONNECTION OBJECT

To interact with a database, you must have a connection to it. The connection helps identify the database server, the database name, user name, password, and other parameters that are required for connecting to the data base. A connection object is used by command objects so they will know which database to execute the command on.

THE SQLCOMMAND OBJECT

The process of interacting with a database means that you must specify the actions you want to occur. This is done with a command object. You use a command object to send SQL statements to the database. A command object uses a connection object to figure out which database to communicate with. You can use a command object alone, to execute a command directly, or assign a reference to a command object to an SqlDataAdapter, which holds a set of commands that work on a group of data as described below.

THESQLDATAREADER OBJECT

Many data operations require that you only get a stream of data for reading. The data reader object allows you to obtain the results of a SELECT statement from a command object. For performance reasons, the data returned from a data reader is a fast forward-only stream of data. This means that you can only pull the data from the stream in a sequential manner This is good for speed, but if you need to manipulate data, then a Dataset is a better object to work with.

THE DATASET OBJECT

DataSet objects are in-memory representations of data. They contain multiple Datatable objects, which contain columns and rows, just like normal database tables. You can even define relations between tables to create parent-child relationships. The DataSet is specifically designed to help manage data in memory and to support disconnected operations on data, when such a scenario make sense. The DataSet is an object that is used by all of the Data Providers, which is why it does not have a Data Provider specific prefix.

The SqlDataAdapter Object

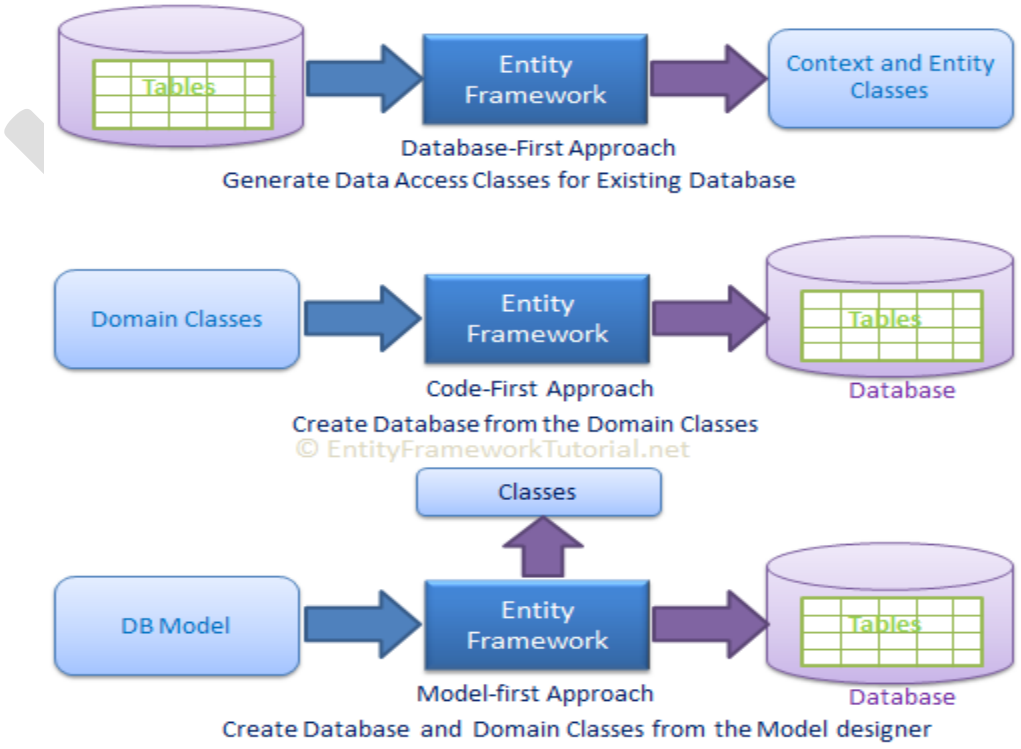
Sometimes the data you work with is primarily read-only and you rarely need to make changes to the underlying data source .Some situations also call for caching data in memory to minimize the number of database calls for data that does not change. The data adapter makes it easy for you to accomplish these things by helping to manage data in a disconnected mode. The data adapter fills a DataSet object when reading the data and writes in a single batch when persisting changes back to the database. A data adapter contains a reference to the connection object and opens and closes the connection automatically when reading from or writing to the database. Additionally, the data adapter contains command object references for SELECT, INSERT, UPDATE, and DELETE operations on the data. You will have a data adapter defined for each table in a DataSet and it will take care of all communication with the database for you. All you need to do is tell the data adapter when to load from or write to the database.

DOI : https://dx.doi.org/10.26808/rs.st.i7v6.17
International Journal of Advanced Scientific and Technical Research Issue 7 volume 6 Nov.- Dec. 2017
Available online on http://www.rspublication.com/ijst/index.html ISSN 2249-9954

ENTITY FRAMEWORK

Entity Framework (EF) is an open source object-relational mapping (ORM) framework for ADO.NET, part of .NET Framework. It is a set of technologies in ADO.NET that supports the development of data-oriented software applications. Architects and developers of data-oriented applications have typically struggled with the need to achieve two very different objectives. The Entity Framework enables developers to work with data in the form of domain-specific objects and properties, such as customers and customer addresses, without having to concern themselves with the underlying database tables and columns where this data is stored. With the Entity Framework, developers can work at a higher level of abstraction when they deal with data, and can create and maintain data-oriented applications with less code than in traditional applications.Writing and managing ADO.Net code for data access is a tedious and monotonous job. Microsoft has provided an O/RM framework called "Entity Framework" to automate database related activities for your application. Microsoft has given the following definition of Entity Framework:"The Microsoft ADO.NET Entity Framework is an Object/Relational Mapping (ORM) framework that enables developers to work with relational data as domain-specific objects, eliminating the need for most of the data access plumbing code that developers usually need to write. Using the Entity Framework, developers issue queries using LINQ, then retrieve and manipulate data as strongly typed objects. The Entity Framework's ORM implementation provides services like change tracking, identity resolution, lazy loading, and query translation so that developers can focus on their application-specific business logic rather than the data access fundamentals."Entity framework is an Object/Relational Mapping (O/RM) framework. It is an enhancement to ADO.NET that gives developers an automated mechanism for accessing & storing the data in the database. Entity framework is useful in three scenarios. First, if you already have existing database or you want to design your database ahead of other parts of theapplication. Second, you want to focus on your domain classes and then create the database from your domain classes. Third, you want to design your database schema on the visual designer and then create the database and classes.

The following figure illustrates the above scenarios.

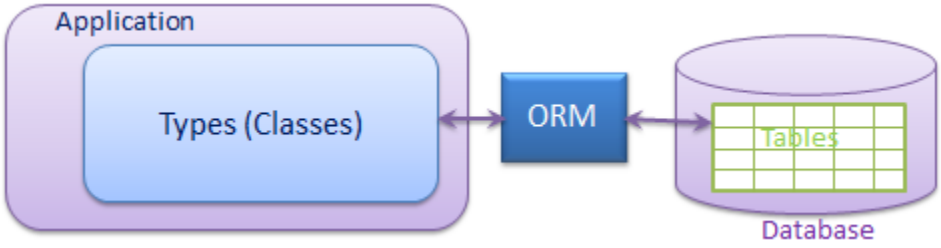


DOI : https://dx.doi.org/10.26808/rs.st.i7v6.17
International Journal of Advanced Scientific and Technical Research Issue 7 volume 6 Nov.- Dec. 2017
Available online on http://www.rspublication.com/ijst/index.html ISSN 2249-9954

As per the above figure, EF creates data access classes for your existing database, so that you can use these classes to interact with the database instead of ADO.Net directly.EF can also create the database from your domain classes, thus you can focus on your domain-driven design.EF provides you a model designer where you can design your DB model and then EF creates database and classes based on your DB model.

O/RM

ORM is a tool for storing data from domain objects to relational database like MS SQL Server, in an automated way, without much programming. O/RM includes three main parts: Domain class objects, Relational database objects and Mapping information on how domain objects map to relational database objects (tables, views & storedprocedures). ORM allows us to keep our database design separate from our domain class design. This makes the application maintainable and extendable. It also automates standard CRUD operation (Create, Read, Update & Delete) so that the developer doesn't need to write it manually.A typical ORM tool generates classes for the database interaction for your application as shown below.



There are many ORM frameworks for .net in the market such as DataObjects.Net, NHibernate, OpenAccess, SubSonic etc. Entity Framework is an open source ORM framework from Microsoft.

ADO.NET V/S EF

All of the standard ORM arguments apply here. The highlights are that you can write code against the Entity Framework and the system will automatically produce objects for you as well as track changes on those objects and simplify the process of updating the database. The EF can therefore replace a large chunk of code you would otherwise have to write and maintain yourself. Further, because the mapping between your objects and your database is specified declaratively instead of in code, if you need to change your database schema, you can minimize the impact on the code you have to modify in your applications--so the system provides a level of abstraction which helps isolate the app from the database. Finally, the queries and other operations you write into your code are specified in a syntax that is not specific to any particular database vendor--in ado.net prior to the EF, ado.net provided a common syntax for creating connections, executing queries and processing results, but there was no common language for the queries themselves; ado.net just passed a string from your program down to the provider without manipulating that string at all, and if you wanted to move an app from Oracle to SQL Server, you would have to change a number of the queries. With the EF, the queries are written in LINQ or Entity SQL and then translated at runtime by the providers to the particular back-end query syntax for that database.

CONCLUSION

ADO.NET is the .NET technology for interacting with data sources. You have several Data Providers, which allow communication with different data sources, depending on the protocols they use or what the database is. Regardless, of which Data Provider used, you'll
use a similar set of objects to interact with a data source. The SqlConnection object lets you manage a connection to a data source. SqlCommand objects allow you to talk to a data source and send commands

to it. To have fast forward-only read access to data, use the SqlDataReader. If you want to work with disconnected data, use a DataSet and implement reading and writing to/from the data source with a SqlDataAdapter.

REFERENCES

1. http://csharp-station.com/Tutorial/AdoDotNet/Lesson01
2. https://blogs.msdn.microsoft.com/dsimmons/2008/05/17/why-use-the-entity-framework/