Namespace

Software projects consist of several pieces of code such as classes, declarations, procedures and functions etc., known as the component or identifiers of the software project. In large projects the number of these components can be very large. These components can be grouped into smaller subcategories. This logical grouping construct is known as a "Namespace" or we can say that the group of code having a specific name is a "Namespace". In a Namespace the groups of components are somehow related to each other. Namespaces are similar in concept to a folder in a computer file system. Like folders, namespaces enable classes to have a unique name or we can say that it is a logical naming scheme for grouping related types. A Namespace is sometimes also called a name scope. Every project in VB.NET starts with a Namespace, by default the same name as the name of the project.

Need of Namespace

We must add a reference of the Namespace object before using that object in a project. Several references are automatically added in the project by default. The VB.Net "Imports" keyword is used to add a reference of a namespace manually.

Example

Imports System

Note: Imports allow access to classes in the referenced Namespace only not in its internal or child Namespaces. If we want to access internal Namespace we might need to write:

Imports System.Collections

Namespaces are basically used to avoid naming collisions, when we have multiple classes with the same name, and it is also helpful for organizing classes libraries in a hierarchal structure. Namespaces allow us to organize Classes so that they can be easily accessed in other applications. Namespaces also enable reusability.

A class in .Net Framework cannot belong to multiple Namespaces. One class should belong to only one Namespace. VB.NET does not allow two classes with the same name to be used in a program.

We can define a Namespace using the "Namespace" keyword. The syntax for declaring a Namespace is:

Namespace <Namespace\_name>

// Classes and/or structs and/or enums etc.

End Namespace

Example

All the classes in the .Net Framework belongs to the System Namespace. The "system" Namespace has built-in VB functionality and all other Namespaces are based on this "system" Namespace.

Accessing Members of a Namespace

We can access a member of a Namespace by using a dot(.) operator, also known as the period operator. The members of a Namespace are the variables, procedures and classes that are defined within a Namespace. To access the member of a namespace in a desired location type the name of the namespace followed by the dot or period operator followed by the desired member of the namespace.

Example

MyNamespace.Class1.disp() 'Accessing elements of the MyNamspace

we can access a member of a namespace in various ways. The following program shows accessing the element of a namespace in various ways.

Imports System

Namespace Birds 'user defined namespace Bird

Class Parrot ‘Parrot is a class in the namespace Animals

Public Shared Function fly () ‘Fly is a function in this Class

Console.WriteLine("Parrot can fly")

End Function

Public Shared Function color() ' color is another function in parrot class

Console.WriteLine("normally Parrots are green")

End Function

Public Shared Function type()

Console.WriteLine("Different type of parrot are found around the world")

End Function

End Class

End Namespace

Module Module1

Public Function myfunction()

Dim P As Birds.Parrot

P = New Birds.Parrot()

P.type() 'accessing member of the namespace bird

End Function

Sub main()

Console.Clear()

Birds.Parrot.fly() 'accessing member of the namespace

ConsoleApplication5.Birds.Parrot.color() 'another way to access member of the namespace

myfunction()

End Sub

End Module

Output

Namespace MyNamespace 'class with in a namespace

Public Class Class1

Public Shared Function disp() 'function declared within the class

Console.Write("hello" & vbCrLf)

End Function

End Class

End Namespace

ADO.NET Data Namespaces

System.Data and its five supporting namespaces define the ADO.NET functionality. These namespaces reside in the System.Data.dll assembly. Figure shows the contents of this assembly in the IL DASM utility. (The ILDASM stands for Intermediate Language Disassembler. This is a de-compiler which helps to get the source code from the assembly. This ILDASM converts an assembly to instructions from which source code can be obtained.)

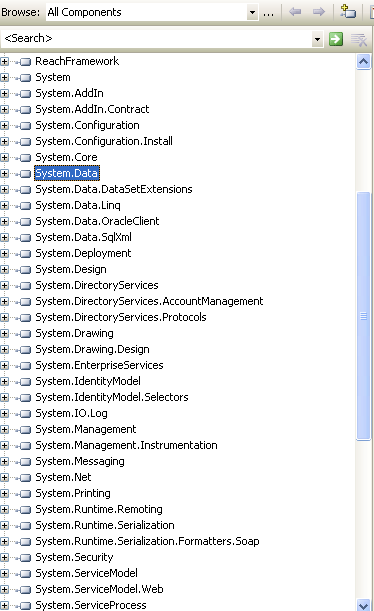


Figure: The system. Data assembly contents in the IL DASM utility

The six ADO.NET namespaces are the System.Data, System.Data.Common, System.Data.Oledb, Microsoft.Data.Odbc, System.Data.SqlClient, and System.Data.SqlTypes.

System.Data

The System.Data namespace is the core namespace of ADO .NET. It consists of the base classes for the ADO.NET architecture. All data providers use these classes. It defines classes that represent table, columns, row, and datasets some common classes from this namespace DataView, DataViewManager DataSet, DataTable, DataRow, DataColumn, and DataRelation. To use these classes in your applications, you need to add a reference to the System.Data namespace.



System.Data.Common



The System.Data.Common namespace defines common classes. These classes are base classes for concrete data provider classes. These classes are shared among all data providers. DBConnection, DataAdapter, DBDataAdapter, DataColumnMapping and DataTableMapping are some of the classed defined in this namespace. To use these classes in your application, you need to add a reference to the System.Data.Common namespace in your application.



The System.Data.OleDb



The System.Data.OleDb namespace defines classes to work with OLE-DB data sources using.NET OleDb data providers. To work with an OLE-DB data source, you must have an OLE-DB provider for that data source. Each data provider component has a class corresponding to it. These classes start with OleDb followed by the component. For example, OleDbConnection class represents a Connection object. Some of the common classes of this namespace are OleDbDataAdapter, OleDbDataReader, OleDbCommand, OleDbCommandBuilder OleDbError, OleDbParameter, OleDbPermission, and OleDbTransaction. To use these classes in your application, your need to add a reference to the System.Data.OleDb namespace in your application.



System.Data.Odbc



Similar to the System.Data.OleDb namespace, the Microsoft.Data.Odbc namespaces define ODBC.NET data provider classes to work with the ODBC data sources. To work with ODBC data source, you need to install an ODBC driver for a database. The Microsoft.Data.Odbc namespace classes start with odbc, followed by the component.



For example, the OdbcConnection class represents a connection object. Some of the common classes of this namespace are OdbcDataAdapter, OdbcDataReader, OdbcCommand, OdbcCommandBuilder, OdbcError, OdbcParameter, OdbcPermission, and OdbcTransaction. To use these classes in your application, you need to add a reference to the Microsoft.Data.Odbc namespace in your application.

System.Data.SqlClient

The System.Data.SqlClient namespaces define Sql.NET data provider classes to work with SQL server 7.0 or later databases. Similar to ODBC and OleDb classes, the classes in this namespace start with sql. For example, common classes are SqlConnection, SqlDataAdapter, SqlCommand, SqlDataReader, and SqlTransaction.



System.DataSqlTypes

The last namespace, System.DataSqlTypes, providers a group classes representing the specific types found in SQL server. Some of these classes are SqlBinary SqlMoney, SqlString, SqlDouble, SqlDateTime, and SqlNumeric.