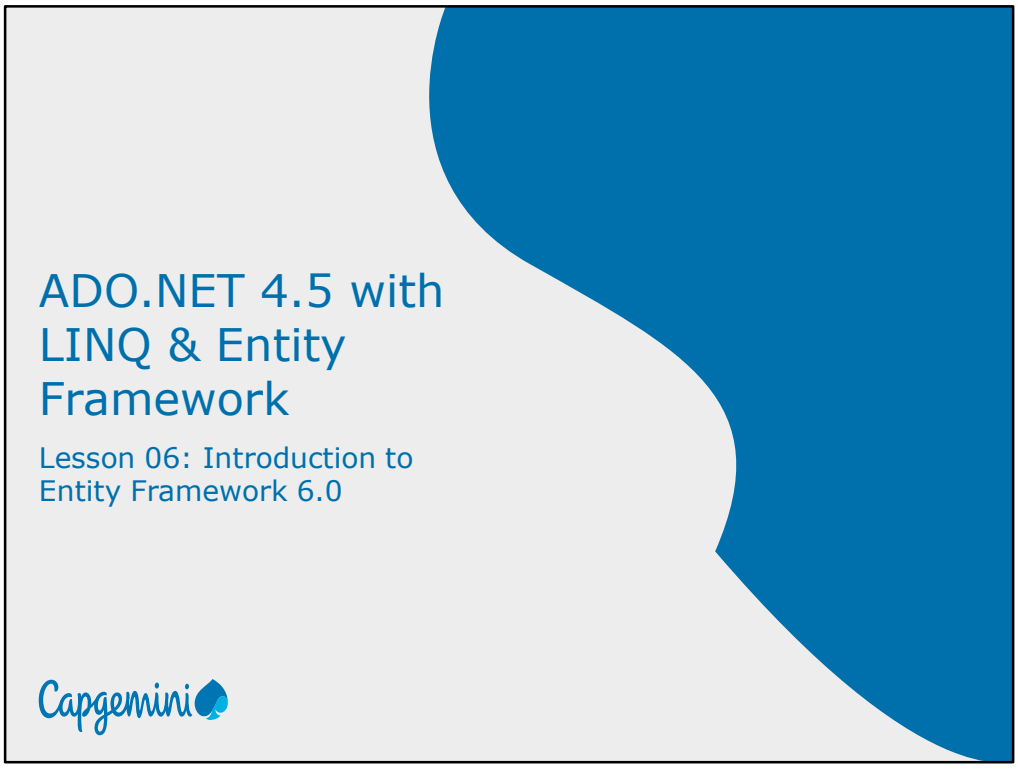

ADO.NET 4.5 with LINQ & Entity Framework
Lesson 06: Introduction to Entity Framework 6.0
Capgemini

Lesson Objectives

- In this lesson we will cover the following
  - Need of Entity Framework
  - Entity Data Model
  - Working with Entities
  - EF Features
  - Back End Support

Need of Entity Framework

- Accessing data is one of the main activity almost every application must do
- The data for these application comes from different source i.e in memory data, XML Files, database ,text files etc.
- Developer need to write the code for accessing these data which is a very cumbersome task
- Developer need to write down .NET Classes which will be used access data from the database
- Writing these classes and mapping them to database makes the application complex and it's timing consuming as well
- To overcome these problem Entity Framework was introduced as part of .NET Framework 3.5 sp1

Accessing is one of the main activity almost every application must do .

In .NET a developer use ADO. Net as a data access technology for accomplishing this task , for which they write down data access code with in the application .
Writing down these code and managing them is a lot difficult task. Developers uses DataReader and DataSet for accessing data usinf ADO.NET.
Yet, with all the improvements being made to ADO.NET, there was still a disconnect between the application and the back-end database.

Developer has to spend a lot time trying to keep up change being made to database including change in schema or adding a new stored procedure which could potentially break the application flow. Due which developer has to just focus on the data access code instead of the application logic .

To overcome all these problem Microsoft Introduced ADO.NET Entity Framework as a part of .Net Framework 3.5 SP1

X.1:Introduction to Entity Framework

Overview

- Entity Framework or EF in an ORM framework for ADO.NET
  - ORM "Object Relation Mapping" is a Programming technique for converting data between incompatible type system.
  - It Creates a virtiual object database that can be used with in the programming language
- EF enables developer to work with relational data as a domain specific object .
- It Eliminates the need of most of the data access code usually which developer need to write.
- It works on top ADO.NET
- Latest Version of EF is Entity Framework 6

Entity Framework Overview

Microsoft Introduced Entity Framework in .NET Framework 3.5 SP1 and continued it in the next releases of .NET.

The Latest version is Entity Framework 6.0 which has got a lot changes as compared to the previous version of Entity Framework

Entity Framework is an Object Relation Mapping Framework which allows a developer to map .NET classes to Database Tables in a Real time Application. It reduces the developer efforts for writing down the data access code.
It work on top of ADO.NET for doing all the database related operations. So the developer doesn't need to write ADO.NET Access code in the application they have to create Entity Data Model which add all the database functionality to the application.

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.

X.1:Introduction to Entity Framework

Overview (Contd..)

- Hidden slide with notes

Entity Framework Overview (Contd..)

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

X.2: Breadcrumb

Overview

- History of Entity Framework

EF Version	Features
EF 3.5	Basic O/RM support with Database First approach
EF 4.0	POCO Support, Lazy loading, testability improvements, customizable code generation and the Model First approach.
EF 4.1	First to available in the NuGet, Simplified DBContext API over ObjectContext, Code First approach. EF 4.1.1 patch released with bug fixing of 4.1
EF 4.3	Code First Migrations feature that allows a database created by Code First to be incrementally changed as your Code First model evolves. EF 4.3.1 patch released with bug fixing of EF 4.3.
EF 5.0	Announced EF as Open Source. Introduced Enum support, table-valued functions, spatial data types, multiple-diagrams per model, coloring of shapes on the design surface and batch import of stored procedures, EF Power Tools and various performance improvements.
EF 6.0 Current Release	EF 6.0/6.1 is the latest release of Entity Framework. It includes many new features related to Code First & EF designer like asynchronous query & save, connection Resiliency, dependency resolution etc.

The table describes the history of Entity Framework

The above table list all the version of Entity Framework with all the features provided by them with their release

• The first version of Entity Framework was limited, featuring basic ORM support and the ability to implement a single approach known as *Database First.*

• Version 4 brought us another approach to using Entity Framework: Model First, along with full Plain Old CLR Object (POCO) support and default lazy loading behavior. Soon after, the Entity Framework team released three smaller, or point releases

• Version 4.1 through 4.3, which represented yet another approach to using Entity Framework: Code First.

• Version 5 of Entity Framework coordinated with the release of the .NET 4.5 framework and Visual Studio 2012,delivering significant performance improvements along with support for enums, table value functions, spatial types,the batch import of stored procedures, and deep support with the ASP.NET MVC framework.

• Version 6 of the Entity Framework. Version 6 delivers asynchronous support for querying and updates, stored procedure support for updates in Code First, improved performance, and a long list of new features,

X.2: Breadcrumb

Overview

Entity Framework 6

- Entity Framework 6 is the latest release of Entity Framework
- Entity Framework 6.0 has introduced many new exciting features for Database-First (designer) and Code-First approaches.
- It can download from Nuget Packager Manager in visual studio.
- It can be used in Visual Studio 2012 and higher version of Visual Studio .

Entity Framework 6 is the latest release of Entity Framework providing a lot of new features a compared to the previous version. It is available as part of Visual Studio 2012 and higher version of Visual Studio .

Entity Framework can be added to project using Nuget Package Manager Interface or Nuget Package Manager Console .

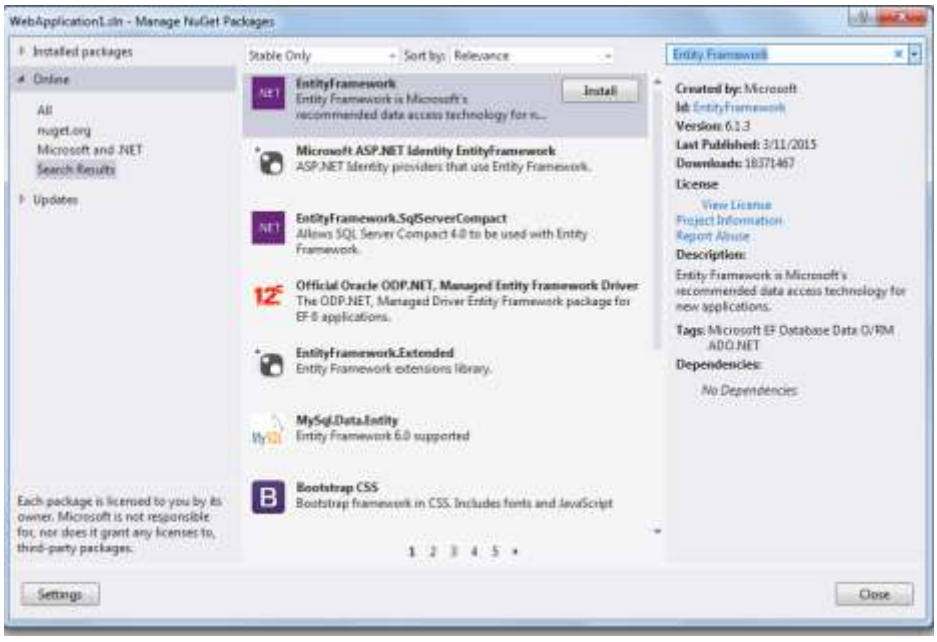
What is Nuget ?

NuGet is the package manager for the Microsoft development platform including .NET. The NuGet client tools provide the ability to produce and consume packages. The NuGet Gallery is the central package repository used by all package authors and consumers.

Nuget Package Manager :- Nuget Package Manager provides a dialog using which we search for a specific package and perform installation or uninstallation of packager

Nuget Package Manager Console :- It is command line tool to add and remove package to a project . It uses power shell command for accomplishing this task.

Adding EF 6.0 using Nuget Package Manager

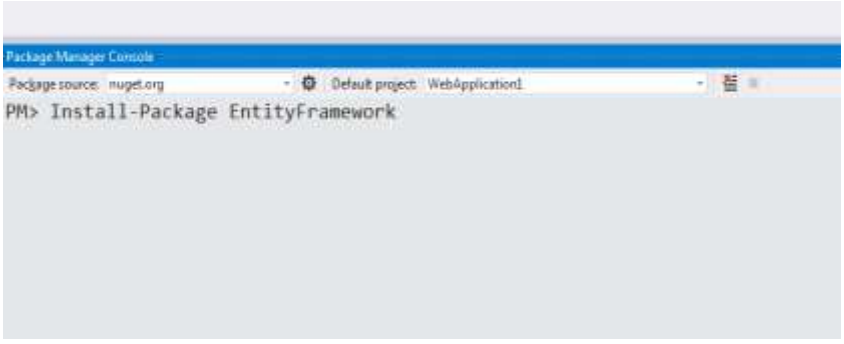


The above image show the dialog for adding the Entity Framework .

It can be accessed by using the following step :-

Tool → Nuget Package Manager → Manage Nuget Package for Solution

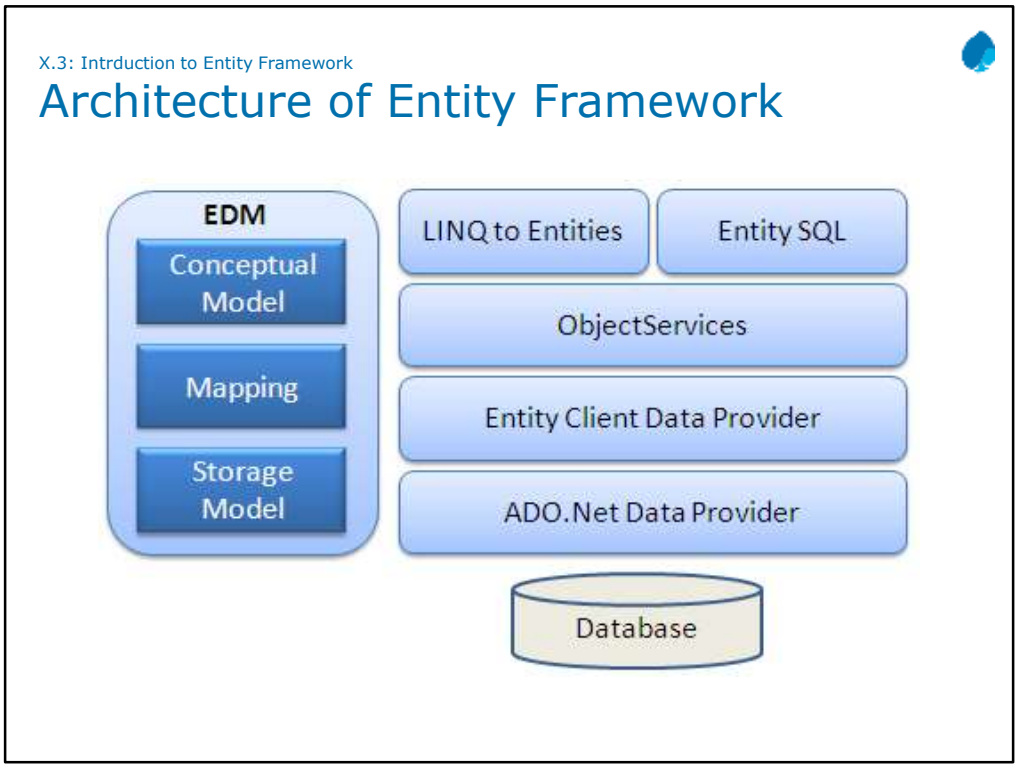
Adding EF 6.0 using Nuget Package Manager



The above image show the command line tool for adding the Entity Framework .

It can be accessed by using the following step :-

Tool → Nuget Package Manager → Package Manager Console



The above diagram show the architecture and component of Entity Framework

Conceptual Model: The model classes and their relationships. This will be independent of the database table design.

Storage Model: Storage model is the database design model which includes tables, views, stored procedures and their relationships and keys.

Mapping: Mapping contain information about how the conceptual model is mapped to the storage model.

LINQ to Entities: returns entities which are defined in the conceptual model.

Entity SQL: a query language similar to LINQ, but a little more difficult.

Object Service: this provides the entry point for accessing data from a database.

Entity Client Data Provider: The main responsibility of this layer is to convert Linq or Entity SQL queries into SQL queries understood by the underlying database

ADO.Net Data Provider: This layer communicates with the database using standard ADO.Net.

X.X: Entity Data Model

Entity Data Model

- Entity Framework uses the Entity Data Model (EDM) to describe the application-specific object or a conceptual model against which the developer programs
- It is an conceptual model of data as you want to represent it in your code.
- It usually consist of .NET Classes that can be manipulated as any other object in code.
- EDM Consist of three main parts
  - Conceptual Model
  - Storage Model
  - Logical Model

EDM:-

EDM is the foundation of the Entity Framework and is comprised of the three model i.e Conceptual Model, Storage Model, Logical Model. The EDM is the component that describes the overall structure of your business object

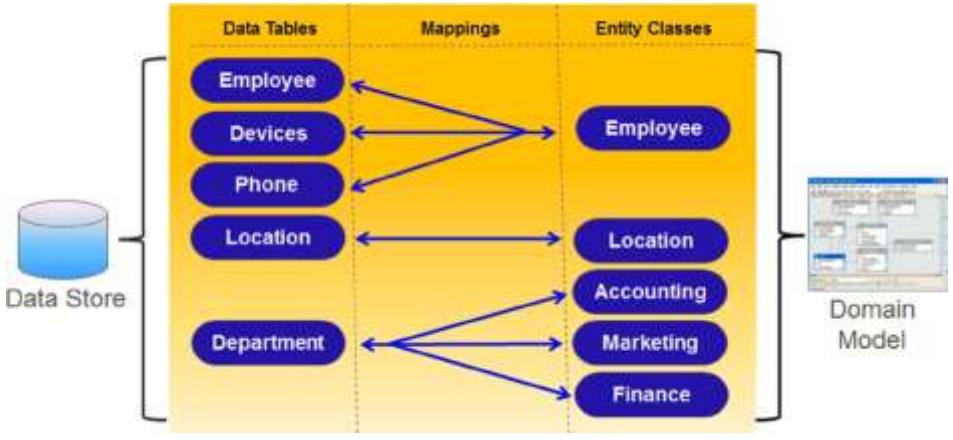
Conceptual Model :- The conceptual model contains the classes and their relationships. This will be independent from your database table design.

Storage Model :- Storage model is the database design model which includes tables, view, stored procedures and their relationship and keys.

Logical Model :- Logical Model consist of the information about how the conceptual model is mapped to storage model.

X.X: Entity Data Model

Entity Data Model



The above diagram show a Entity Data Model

In the above diagram note how the database tables (on the left) do not directly map to the entity classes, which we code against (on the right). Instead, the mapping capabilities built into the Entity Data Model enable the developer to code against a set of entity classes that more closely resemble the problem domain, as opposed to a highly normalized database, designed for performance, scalability, and maintainability.

For example, note above how the Employees, Devices, and Phone numbers) are physically stored in three different tables, which from a DBA perspective makes perfect sense. But the developer codes against a single Employee entity class that contains a collection of Devices and Phone Numbers. From a developer and project stakeholder perspective, an employee is a single object, which happens to contain phone numbers and devices.The developer is unaware, and does not care, that the DBA has normalized this employee object into three separate database tables. Once configured, the mapping between the single class and three database tables is abstracted away and handled by the Entity Framework.

Working With Entities

- A key term any Entity framework developer need to know is the term entity
- Entity are like objects eg:-
  - Entities have a know type
  - Entities have properties and these properties can hold scalar values
  - Entity properties can hold to references to other entites
  - Each entity has a distinct identity
- Entities are extremely flexible
- Entities can have relationships between them

X.X: [Topic]

Working With Entities

- Entities can be created using the following strategies
  - Code First
  - Database First
  - Model First

Code First :- In the Code First approach, you avoid working with visual model designer (EDMX) completely. You write your POCO classes first and then create database from these POCO classes. Developers who follow the path of Domain-Driven Design (DDD) principles, prefer to begin by coding their domain classes first and then generating the database required to persist their data.

Database First :- In the Database First Approach ,you can use to existing database to entity data model and work on it.

Model First :- In the Model First approach, you create Entities, relationships, and inheritance hierarchies directly on the design surface of EDMX and then generate database from your model.So, in the Model First approach, add new ADO.NET Entity Data Model and select **Empty EF Designer model** in Entity Data Model Wizard.

X.X: Entity Data Model

Demo

- How to use Code First Model and Database First Model ?

Add the notes here.
Demonstrate Code First and Database First Model

Steps for Creating EDM using Database First Model.

•**Project -> Add New Item…**

•Select **Data** from the left menu and then **ADO.NET Entity Data Model**

•Enter **BloggingModel** as the name and click **OK**

•This launches the **Entity Data Model Wizard**

•Select **Generate from Database** and click **Next**

•Select the connection to the database you created in the first section, enter **BloggingContext** as the name of the connection string and click **Next**

•Click the checkbox next to 'Tables' to import all tables and click 'Finish'

EF Features

- POCO Support
- Model First Support
- Deferred Loading of Related Objects
- Functions in LINQ to Entities queries
- Plurality Naming Support
- Complex Type Support
- Customized Object Layer Code Generation

EF Features

Poco Support :- One of the more powerful new features of the Entity Framework is the ability to add and use your own custom data classes in conjunction with your data model. This is accomplished by using CLR objects, commonly known as “POCO” (Plain Old CLR Objects). The added benefit comes in the form of not needing to make additional modifications to the data classes. This is also called persistence-ignorance.

Model First Support :- It Allows the developer to create a conceptual model using the create database wizard. It creates the database based on the conceptual model specified.

Deferred Loading of Related Objects :- Deferred Loading also know as Lazy loading . It helps the query result to be shaped by composing queries that explicitly navigate the relationship via the navigation properties

LINQ to Entities :- Adds the Linq query capabilities .Using this you can access data from the entity using LINQ

Plurality Naming :- It Pluralize the name of entities while creating them . This adds a proper Naming Convention which helps developer to access using code.

Complex Type :- Complex types are like entities in that they consist of a scalar property or one or more complex type properties. Thus,complex types are non-scalar properties of entity types that enable scalar properties to be organized within entities.

Object Layer Code Generation :- Object Layer Code is generated by default .It allows developer to add text templates to a project . By using custom text template , the EDM will generate the object context and entity classes.

EF 6 Features for Database First

- Connection resiliency
- Asynchronous query and save
- Code-based configuration
- Database command logging
- Database command interception
- Dependency Resolution
- DbSet.AddRange/RemoveRange
- Better Transaction Support
- Pluggable pluralisation and singularization service
- Testability improvements
- Creating context with an open connection
- Improved performance and warm-up time

EF 6 Features for Code-First

- Custom conventions
- Insert, update & delete stored procedures for entity CUD operation
- Index attribute (EF 6.1)
- Multiple context per database
- Nested entity types
- Custom migration operations
- Configurable migration history table

Back End Support

- Entity Framework is database or data source independent
- It does not really care about the about the data store from which the data is being queried
- It uses following provider to support this feature
  - EntityClient Provider for the Entity Framework
  - .NET Framework Data Provider for SQL

The great thing about the Entity Framework is that in essence it does not really care about the data store from which the data is being queried. Neither the type of database nor the schema itself is completely unknown to the Entity Framework, and they will have no impact on your model.

The Entity Framework ships with two providers:
• EntityClient Provider for the Entity Framework*:* Used by Entity Framework applications to access data described in the EDM. This provider uses the .NET Framework Data Provider for SQL Server (SqlClient) to access a SQL Serverdatabase.
• NET Framework Data Provider for SQL Server (SqlClient) for the Entity Framework: Supports the Entity Framework for use with a SQL Server database.

As Being database or data source independent we can create custom providers to access other database like Oracle, MySql, PostgreSQL, DB2 etc

Summary

- In this lesson you have learnt about:
  - Entity Framework
  - Entity Data Model
  - Architecture of Entity Framework
  - Feature of Entity Framework

Add the notes here.

Review Question

- Which of the following EDM consist of ?
  - Conceptual Model
  - Storage Model
  - Code Model
  - Logic Model

- Entity Framework works on top of ADO.NET?
  - True
  - False

- The _____ contains the classes and their relationships.
  - Logic Model
  - Storage Model
  - Conceptual Model

Add the notes here.