Entity Framework

Entity Framework Basics

Lesson Objectives

- Overview Entity Framework
- EF Architecture
- EF Features

Lesson Objectives

- Workflow in Entity Framework
- Entity State
- Development approaches
- Discussion and Feedback

Section 1

OVERVIEW ENTITY FRAMEWORK

Did you work with ADO.NET?

- Save or retrieve application data from the underlying database
- Boring with repeating code:
  - ✓ open a connection to the database
  - ✓ create a DataSet to fetch or submit the data to the database
  - ✓ convert data from the DataSet to .NET objects or vice-versa to apply business rules

Did you work with ADO.NET?

- Stressful when:
  - ✓ Cannot remember all SQL schema and data
  - ✓ Forget to close connection
  - ✓ Exception when database changed without any notification
  - ✓ SQL Injection
  - ✓ …..

What is Entity Framework?

- Automate all database related activities
- Open-source ORM framework for .NET applications
- Supported by Microsoft

What is Entity Framework?

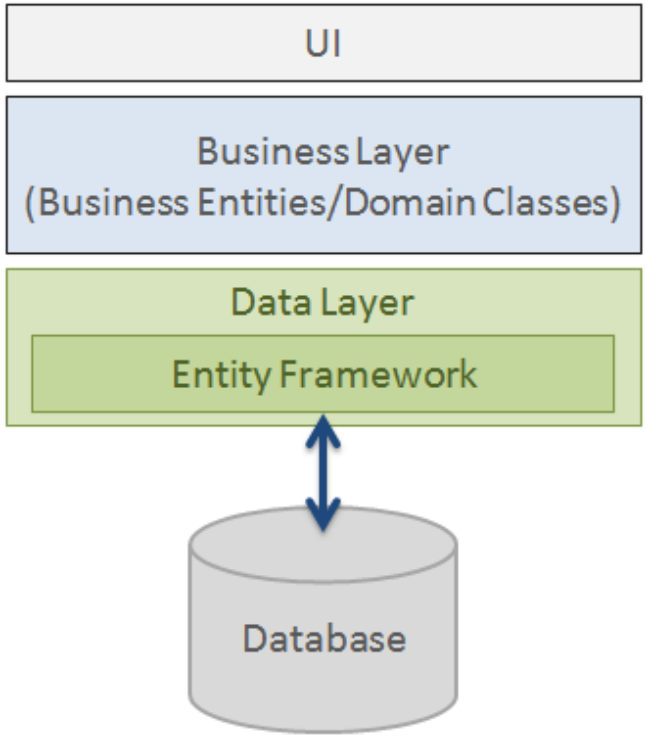
- Enables developers to work with data using objects of domain specific classes without focusing on the underlying database tables and columns where this data is stored

What is Entity Framework?

- Enables developers to work with data using objects of domain specific classes without focusing on the underlying database tables and columns where this data is stored
- 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 compared with traditional applications

What is Entity Framework?

*"Entity Framework is an **object-relational mapper** (O/RM) that enables .NET developers to **work with a database using .NET objects**. It **eliminates** the need for most of the data-access code that developers usually need to write."*

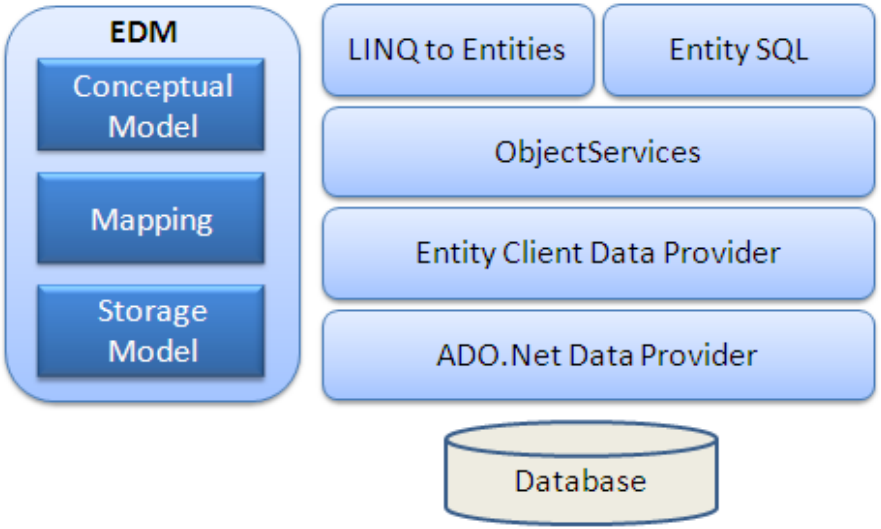


Section 2

EF ARCHITECTURE

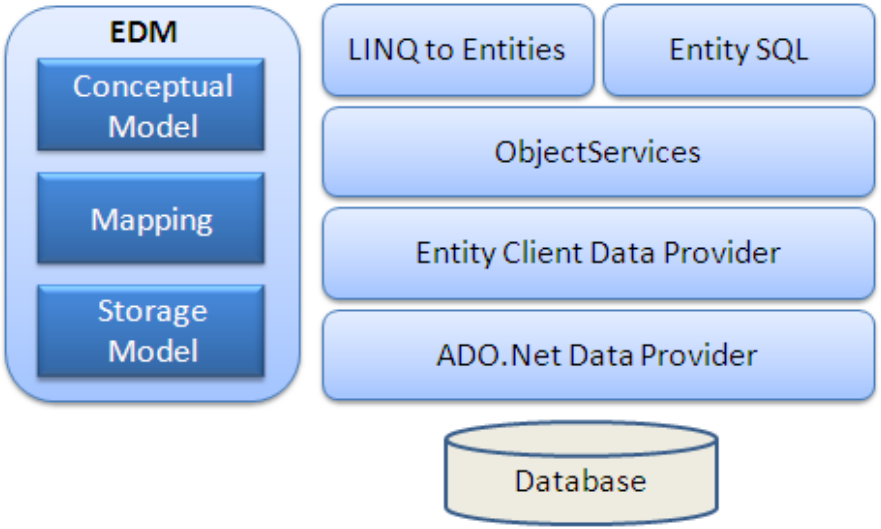
Entity Framework Architecture

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



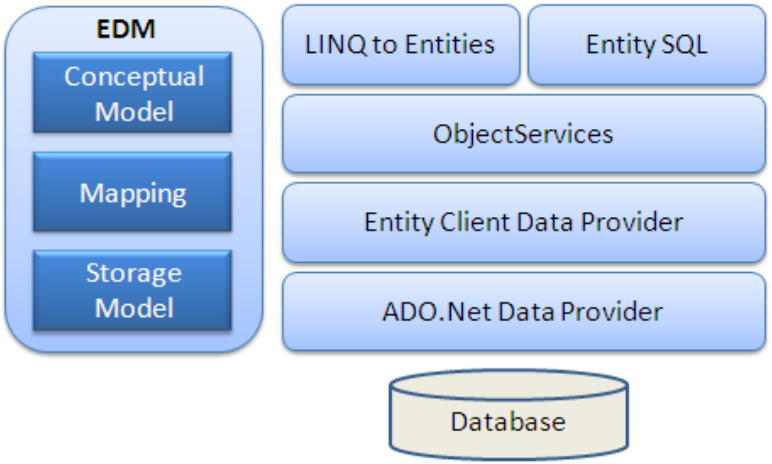
Entity Framework Architecture

- **Mapping:** Mapping consists of information about how the conceptual model is mapped to the storage model.
- **Storage Model:** The storage model is the database design model which includes tables, views, stored procedures, and their relationships and keys.



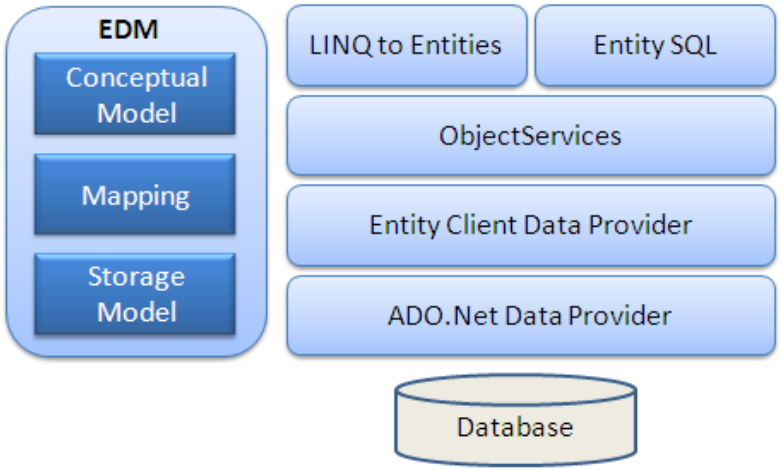
Entity Framework Architecture

- **LINQ to Entities:** LINQ-to-Entities (L2E) is a query language used to write queries against the object model. It returns entities, which are defined in the conceptual model. You can use your LINQ skills here.



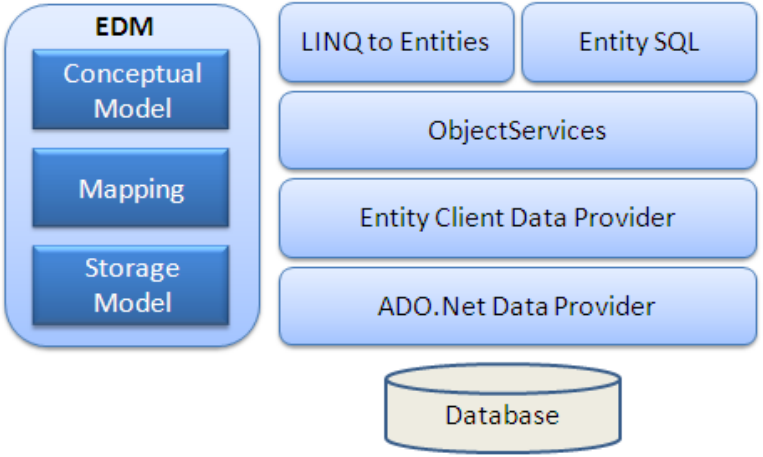
Entity Framework Architecture

- **Entity SQL:** Entity SQL is another query language (For EF 6 only) just like LINQ to Entities. However, it is a little more difficult than L2E and the developer will have to learn it separately.



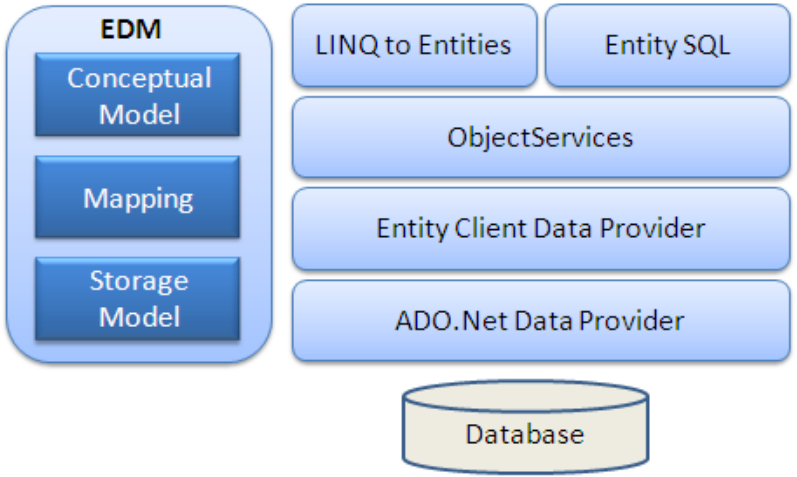
Entity Framework Architecture

- **Object Service:** Object service is a main entry point for accessing data from the database and returning it back. Object service is responsible for materialization, which is the process of converting data returned from an entity client data provider (next layer) to an entity object structure.



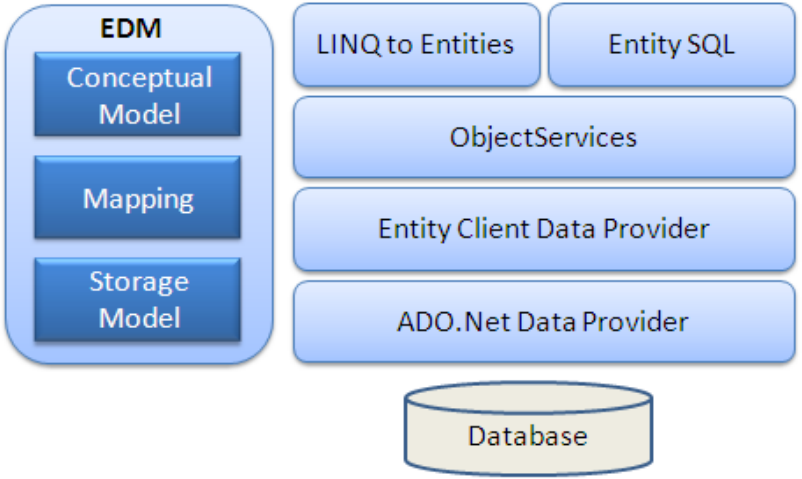
Entity Framework Architecture

- **Entity Client Data Provider:** The main responsibility of this layer is to convert LINQ-to-Entities or Entity SQL queries into a SQL query which is understood by the underlying database. It communicates with the ADO.Net data provider which in turn sends or retrieves data from the database.



Entity Framework Architecture

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



Section 3

EF FEATURES

Entity Framework Features

- **Cross-platform**

  EF Core is a cross-platform framework which can run on Windows, Linux and Mac.

- **Modelling**

  EF (Entity Framework) creates an EDM (Entity Data Model) based on POCO (Plain Old CLR Object) entities with get/set properties of different data types. It uses this model when querying or saving entity data to the underlying database.

- **…**

Entity Framework Features

- **Querying**

  EF allows us to use LINQ queries (C#/VB.NET) to retrieve data from the underlying database. The database provider will translate this LINQ queries to the database-specific query language (e.g. SQL for a relational database). EF also allows us to execute raw SQL queries directly to the database.

- **Change Tracking**

  EF keeps track of changes occurred to instances of your entities (Property values) which need to be submitted to the database.

- **…**

Entity Framework Features

- **Saving**

  EF executes INSERT, UPDATE, and DELETE commands to the database based on the changes occurred to your entities when you call the SaveChanges() method. EF also provides the asynchronous SaveChangesAsync() method.

- **Concurrency**

  EF uses Optimistic Concurrency by default to protect overwriting changes made by ano ther user since data was fetched from the database.

- **…**

Entity Framework Features

- **Transactions**

  EF performs automatic transaction management while querying or saving data.

  It also provides options to customize transaction management.

- **Caching**

  EF includes first level of caching out of the box. So, repeated querying will return data from the cache instead of hitting the database.

- **…**

Entity Framework Features

- **Built-in Conventions**

  EF follows conventions over the configuration programming pattern, and includes a set of default rules which automatically configure the EF model.

- **Configurations**

  EF allows us to configure the EF model by using data annotation attributes or Fluent API to override default conventions.

- **…**

Entity Framework Features

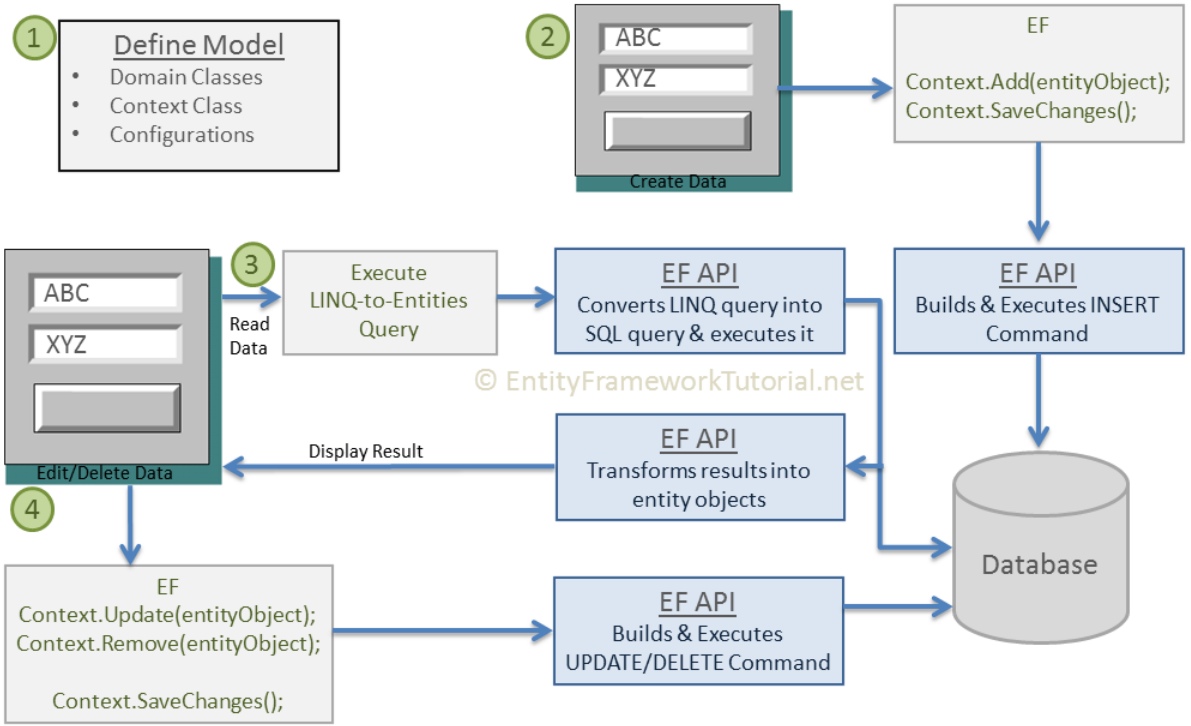
- **Migrations**

  EF provides a set of migration commands that can be executed on the NuGet Package Manager Console or the Command Line Interface to create or manage underlying database Schema.

Section 4

WORKFLOW IN ENTITY FRAMEWORK

Basic Workflow in EF



Define model

- Defining the model includes defining
  - ✓ domain classes,
  - ✓ context class derived from DbContext,
  - ✓ configurations (if any).

Define model

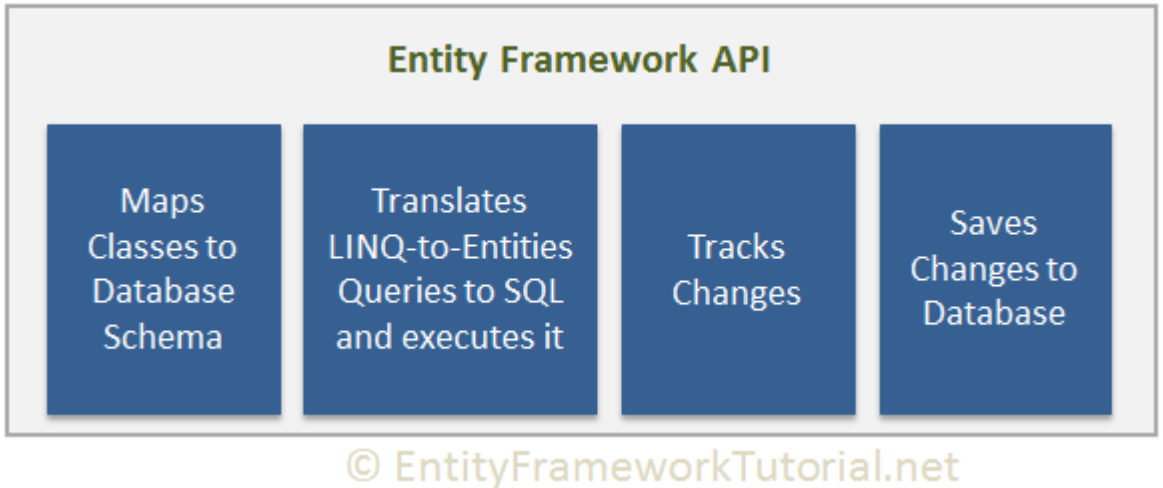
- EF will perform **CRUD** operations based on your model.
- CRUD:
  - ✓ **C**reate
  - ✓ **R**ead
  - ✓ **U**pdate
  - ✓ **D**elete

Define model

Entity Framework API (EF6 & EF Core) includes the ability to map domain (entity) classes to the database schema, translate & execute LINQ queries to SQL, track changes occurred on entities during their lifetime, and save changes to the database.

Define model

Entity Framework API (EF6 & EF Core) includes the ability to map domain (entity) classes to the database schema, translate & execute LINQ queries to SQL, track changes occurred on entities during their lifetime, and save changes to the database.



Insert data

- Add a domain object to a context
- Call the SaveChanges() method.
- EF API will build an appropriate INSERT command and execute it to the database.

Insert data

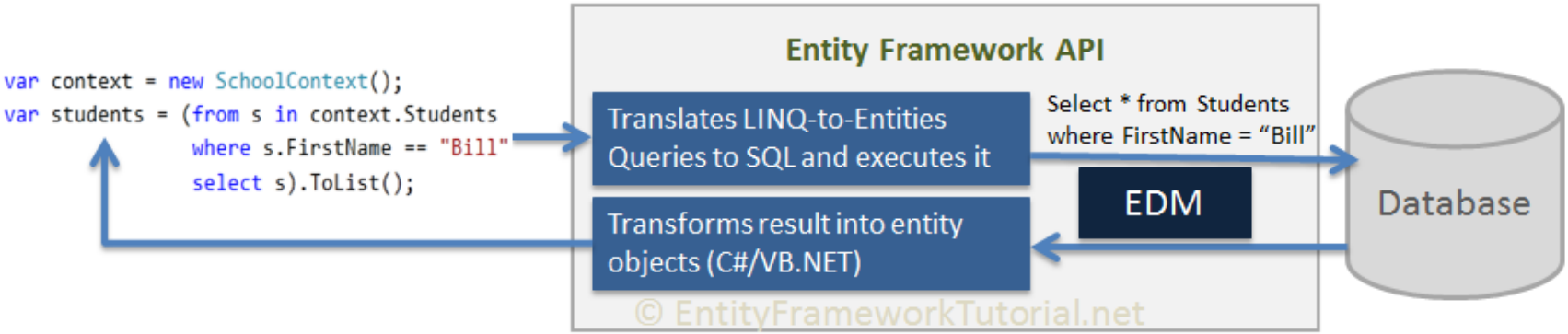
- Add a domain object to a context
- Call the SaveChanges() method.
- EF API will build an appropriate INSERT command and execute it to the database.

Question: Can we add more than **one** object at the time?

Read data

- Execute the LINQ-to-Entities query in your preferred language (C#/VB.NET).
- EF API will convert this query into SQL query for the underlying relational database and execute it.
- The result will be transformed into domain (entity) objects and ready for display

Read data



EF API translates LINQ-to-Entities queries to SQL queries for relational databases using EDM and also converts results back to entity objects.

Edit or Delete data

- Update or Remove entity objects from a context
- Call the SaveChanges() method.
- EF API will build the appropriate UPDATE or DELETE command and execute it to the database.

Edit or Delete data

- Update or Remove entity objects from a context
- Call the SaveChanges() method.
- EF API will build the appropriate UPDATE or DELETE command and execute it to the database.

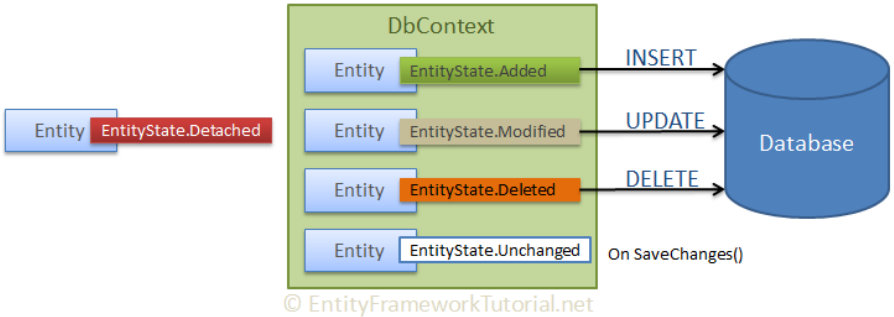
Question: What happed if we forget calling SaveChanges()?

Section 5

ENTITY STATE

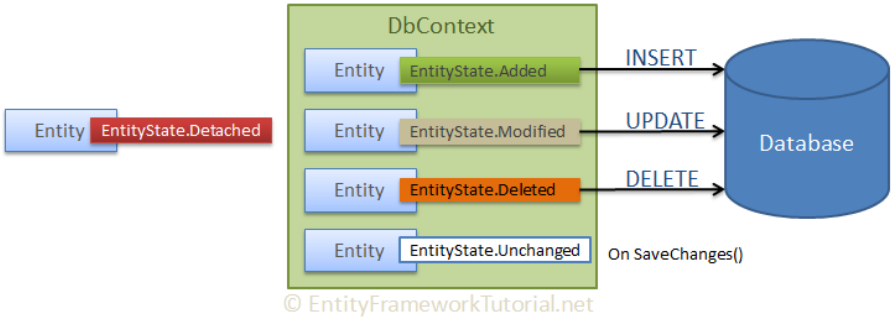
Entity State

- EF API maintains the state of each entity during an its lifetime.
- Each entity has a state based on the operation performed on it via the context class.



Entity State

- The entity state are:
  1. **Added**
  2. **Modified**
  3. **Deleted**
  4. **Unchanged**
  5. **Detached**



Section 6

DEVELOPMENT APPROACHES

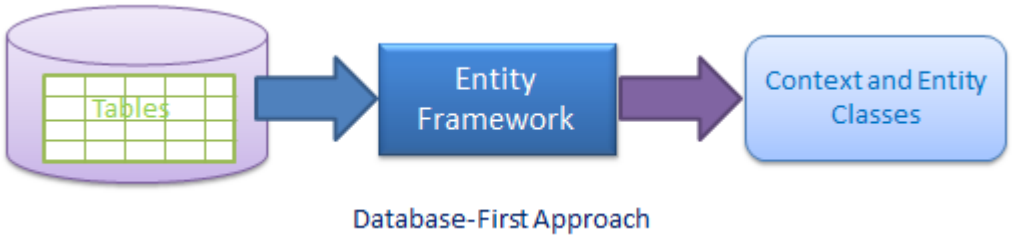
Development Approaches

- There are three different approaches you can use while developing your application using Entity Framework:
  - ✓ Database-First
  - ✓ Code-First
  - ✓ Model-First

Database-First Approach

In the database-first development approach, you generate the context and entities for the existing database using EDM wizard integrated in Visual Studio or executing EF commands.

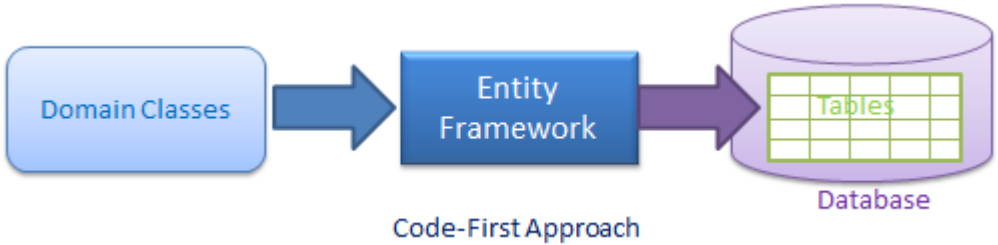
- EF 6 supports the database-first approach extensively.
- EF Core includes limited support for this approach.



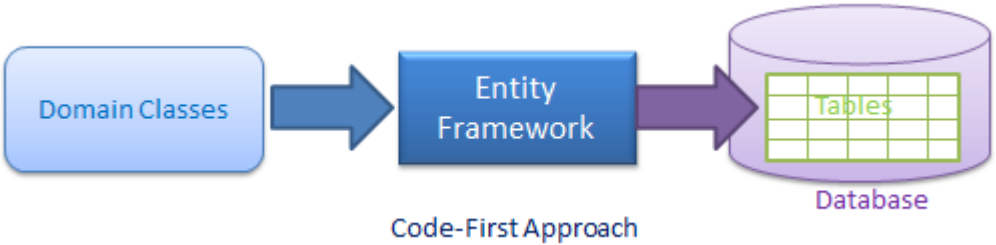
Database-First Approach

Code-First Approach

- Use this approach when you do not have an existing database for your application.
  - ✓ start writing your entities (domain classes) and context class first
  - ✓ create the database from these classes using migration commands.
- Why do developers prefer this approach?



Code-First Approach



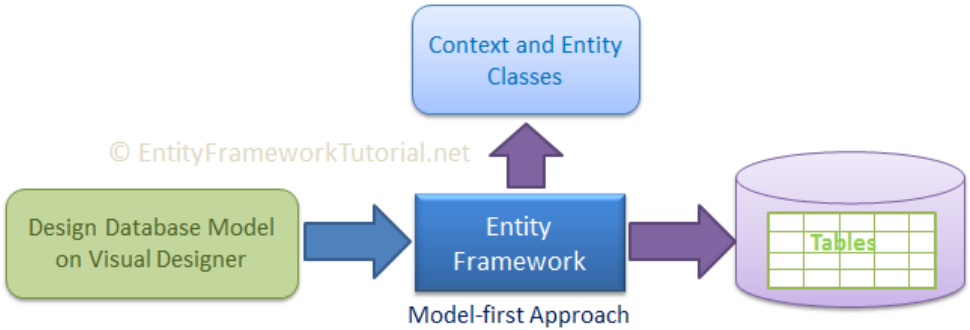
- Why do developers prefer this approach?

Model-First Approach

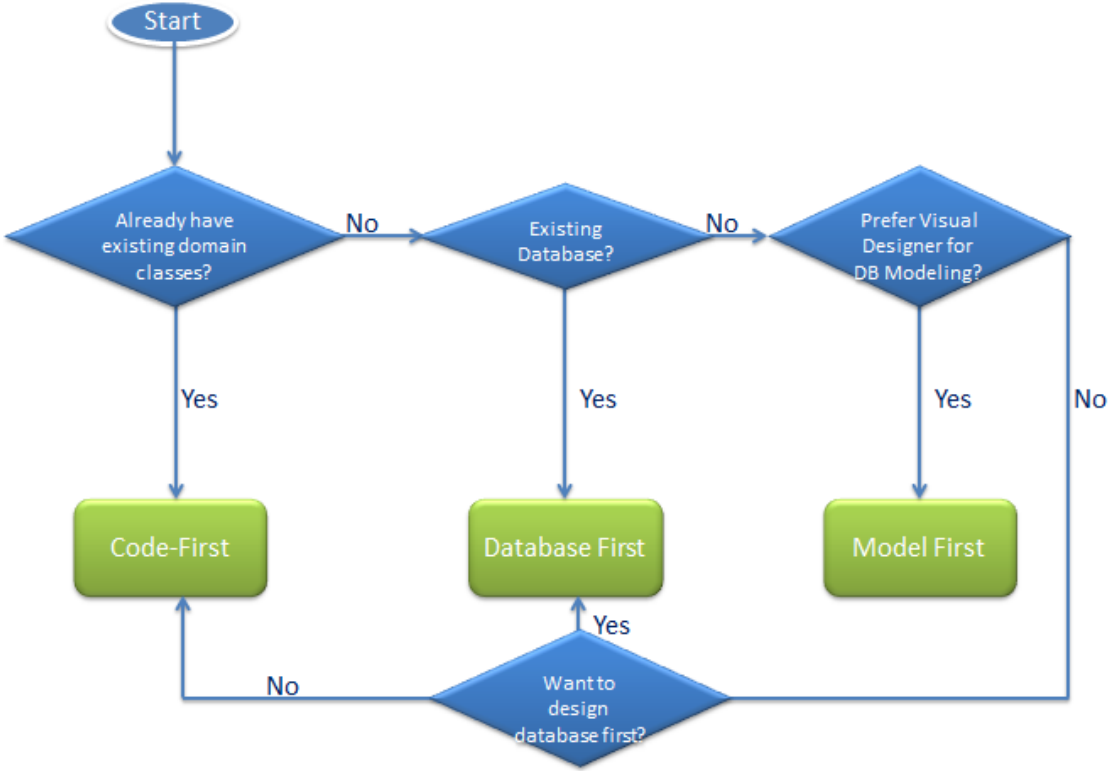
- In the model-first approach, developer
  - ✓ creates entities, relationships,
  - ✓ inheritance hierarchies directly on the visual designer integrated in Visual Studio
  - ✓ generate entities, the context class, and the database script from your visual model.

Model-First Approach

- EF 6 includes limited support for this approach.
- EF Core **does not** support this approach.



Choosing the Approach



Discussion

- With Entity Framework, developer **does NOT** need to know SQL?

Discussion

- What is/are **disadvantage(s)** of Entity Framework?
  1.
  2.
  3.

Give your feedback

- Should you use Entity Framework or not?
- Explain your decision:
  - ✓
  - ✓
  - ✓
  - ✓

Thank you