# ТЕХНОЛОГИЯ ENTITY FRAMEWORK CORE

**Алексей КОЗЬМА**

*Департамент Программной Инженерии и Автоматики, группа TI-195, Факультет Вычислительной Техники, Информатики и Микроэлектроники, Технический Университет Молдовы, Кишинев, Республика Молдова*

Автор корреспондент: Алексей КОЗЬМА, e-mail: cozma.alexei@isa.utm.md

**Научный руководитель: Дориан САРАНЧУК,** DISA, FCIM, UTM

***Аннотация:*** *Данная статья посвящена технологии Entity Framework Core. Из нее можно будет узнать, для чего была придумана данная технология, и какие проблемы в сфере Баз Данных она решает. В статье будут рассмотрены отличия EF Core от его предшественника, а также преимущества использования именно этой технологии, и в каких сферах и почему она наиболее широко применяется.*

***Ключевые слова:*** *ORM-инструмент (Object-Relational Mapping), Code-First, Database-First, LINQ(Language-Integrated Query), .NET Core.*

## Введение

Entity Framework Core – это технология компании Microsoft, пришедшая на смену технологии Entity Framework, представленная миру 27 июня 2016 года вместе с такими технологиями как NET Core и ASP.NET Core. Ключевыми свойствами новой серии Core стали открытый исходный код, уменьшенный размер, расширяемость и кросс-платформенность [1].

## ORM-технология

Entity Framework Core – это ORM-инструмент, то есть он позволяет работать с базами данных на более высоком уровне абстракции, взаимодействуя с таблицами и находящимися в них данными, как с объектами.

ORM – это механизм, который позволяет создать связь между объектами, написанными при помощи объектно-ориентированных языков, таких как, например, Java или C#, и реляционной базой данных. Благодаря этому механизму у разработчиков имеется возможность взаимодействовать с объектами, не задумываясь, где и каким образом хранятся данные. Такой уровень абстракции позволяет не изменять приложение каждый раз, когда меняется способ хранения данных и существенно сократить время при разработке программного обеспечения [2, 3].

## С какими СУБД можно использовать EF Core

Entity Framework Core поддерживает любую СУБД(систему управления базами данных), для которой существует нужный провайдер. По умолчанию Microsoft предоставляет встроенные провайдеры для работы с MS SQL Server, PostgreSQL, SQLite, однако можно найти провайдеры также для MySQL, OracleDB и т. д..

## Какие методы разработки поддерживает EF Core

Entity Framework Core поддерживает 2 метода разработки:

1. Code-First(основной способ)
2. Database-First

В первом случае EF Core создает базу данных с отношениями, используя миграции, основывающиеся на конфигурации предоставленной в классах.

Во втором случае EF Core, исходя из базы данных, создает необходимые классы, однако данный метод не является предпочтительным, так как EF Core не предоставляет удобного дизайнера для создания базы данных.

**Различия между EF Core и EF 6.x**

В первую очередь стоит отметить, что новый EF Core стал более компактным и расширяемым по сравнению со своим предшественником и получил возможность применяться на любых системах (т.н. кросс-платформенность). Однако это не единственные их различия. Кроме этого можно выделить:

- поддержку баз данных, хранящихся в оперативной памяти (In-memory databases), что значительно облегчает тестирование приложений, за счет того, что обращения к ней происходят намного быстрее и вероятность нечаянно допустить потерю каких-нибудь данных из рабочей БД сводится к нулю;
- удаление EDM (Entity Data Model) дизайнера, вследствие чего EF Core не поддерживает подход Model-First в отличие от EF;
- поддержка IoC (Inversion-of-Control);
- поддержка unique-ограничения;
- поддержка альтернативных ключей [4].

**Преимущества работы с EF Core:**

Entity Framework Core обладает рядом преимуществ для разработки программного обеспечения. Среди них:

- применение подхода Code-First;
- использование одного языка программирования для разработки приложения и для взаимодействия с БД;
- возможность получения информации из БД с помощью технологии LINQ (Language-Integrated Query);
- кросс-платформенность;
- возможность переключаться между различными базами данных, внося минимальные изменения в само приложение;
- удобство в тестировании взаимодействия приложения с БД [5].

**Области применения EF Core**

Благодаря своим преимуществам Entity Framework Core получил широкое распространение в различных типах приложений (Рис. 1).

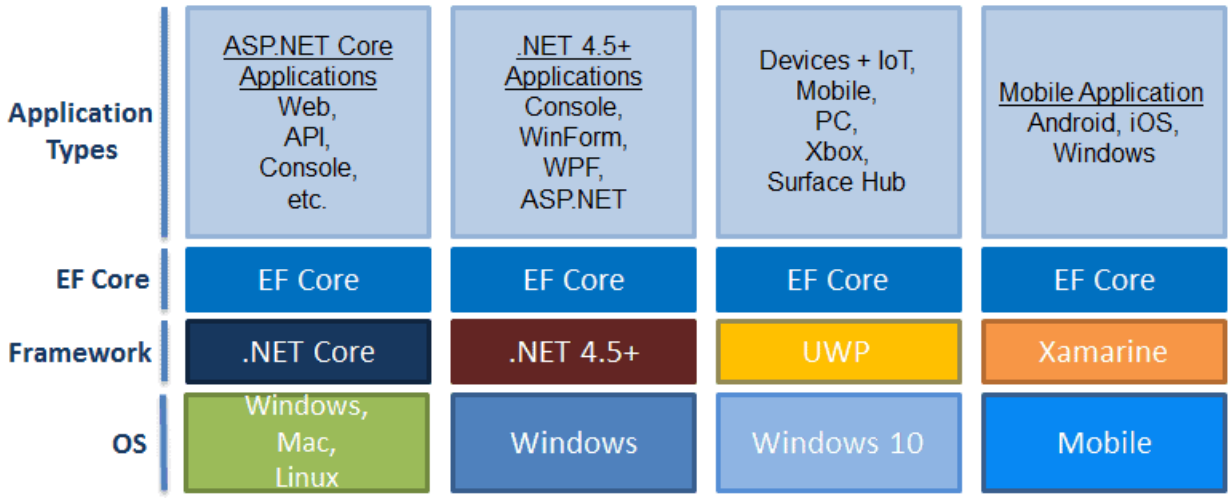


**Рисунок 1. Области применения EF Core**

**Миграции в EF Core**

Благодаря миграциям Entity Framework Core синхронизирует модели, используемые в коде, с базой данных. Таким образом при изменении класса-модели изменится и соответствующий ему объект в БД. Для этого EF Core предоставляет ряд специальных команд для работы с миграциями, исполняемых в NuGet Package Manager Console либо в CLI(Command Line Interface):

- Add-migration <migration name> — создает снэпшот миграции
- Remove-migration — удаляет последний снэпшот миграции
- Update-database — обновляет базу данных, используя последний снэпшот
- Script-migration – генерирует SQL код, используя все снэпшоты миграции

**Добавление миграции**. После создания новой миграции командой Add-migration в папке проекта появятся следующие классы(Рис. 2):

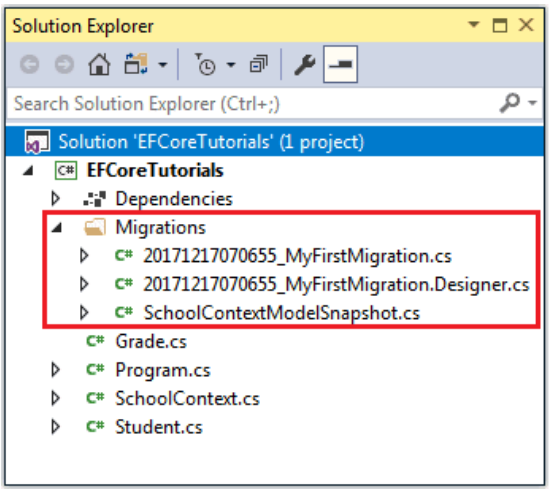


**Рисунок 2. Классы, созданные добавлением миграции**

1. **<timestamp>_<migration name>.cs — главный файл миграции, в котором содержится информация о добавляемых и удаляемых из БД объектов;**
2. **<timestamp>_<migration name>.Designer.cs — файл с метаданными для работы EF Core;**
3. **<contextclassname>ModelSnapshot.cs — снэпшот действующей БД, использующийся для определения изменений.**

**Удаление миграции**. При использовании команды Remove-migration удаляется последний сделанный снэпшот, если он не был применен к базе данных. Попытка удалить снэпшот после обновления базы данных на его основе приведет к ошибке Entity Framework Core.

**Обновление базы данных**. При использовании команды Update-database будет использован последний снэпшот для создания или обновления базы данных. Кроме этого Entity Framework Core добавит запись в таблицу _EFMigrationHistory с информацией о миграции нужной для того, чтобы прослеживать изменения БД и совершить откат к предыдущим миграциям при необходимости. Важной особенностью команды Update-database является то, что, указав после самой команды название миграции, пользователь может вернуться к необходимой миграции.

**Получения SQL-скрипта.** При помощи команды script-migration можно получить все изменения в виде SQL-кода в случае, когда необходимо выполнить миграцию вручную [6].

**Заключение**

В заключении можно сказать, что технология Entity Framework Core является прекрасным решением для C#-разработчиков, когда появляется необходимость создать взаимодействие приложения с базой данных. Принимая во внимание частоту использования баз данных в различных проектах и область применения Entity Framework Core, можно утверждать, что данная технология будет полезна к изучению любому программисту, работающему на языке C#.

**Библиография**

1. Введение в Entity Framework Core. [online]. [просмотрен 23.12.2021]. Доступно: https://metanit.com/sharp/entityframeworkcore/1.1.php
2. object-relational mapping (ORM). [online]. [просмотрен 23.12.2021]. Доступно: https://www.theserverside.com/definition/object-relational-mapping-ORM
3. Object-Relational Mapping (ORM). [online]. [просмотрен 23.12.2021]. Доступно: https://www.techopedia.com/definition/24200/object-relational-mapping--orm
4. Compare EF Core & EF6. [online]. [просмотрен 23.12.2021]. Доступно: https://docs.microsoft.com/en-us/ef/efcore-and-ef6/
5. What Is Entity Framework And How Entity Framework Core Is Different. [online]. [просмотрен 23.12.2021]. Доступно: https://www.c-sharpcorner.com/article/what-is-entity-framework-how-entity-framework-core-is-different/
6. Entity Framework Core. [online]. [просмотрен 23.12.2021]. Доступно: https://www.entityframeworktutorial.net/efcore/entity-framework-core.aspx