Grundlagen von Entity Framework Core

In diesem Artikel behandeln wir die folgenden Themen:

1. Was ist Entity Framework und seine Vorteile
2. Was ist ein ORM
3. Was sind Nachteile?
4. Gibt es Alternativen?

1 Was ist Entity Framework und seine Vorteile

Entity Framework ist prinzipiell eine weitere Abstraktionsschicht auf ADO.Net. Es dient dazu Daten von einem Datenbank management system zu verwalten.

Warum dann EF benutzen und nicht ADO.Net? Unter anderem aus folgenden Gründen (technische Details folgen im Kurs)

- Vereinfachte Entwicklung
    - SQL wird von EF generiert (nur selten muss SQL geschrieben werden)
    - Wartung vereinfacht
    - In den meisten Fällen auch bessere Performance der Queries
    - Typisierte Abfragen
    - Kein SQL Injektionsgefahr
    - Es kann auf die Domänenlogik konzentriert werden
    - State von Entities wird verwaltet
- Single point of truth
    - Das Datenbankschema ist unter Source Control
    - EF verwaltet die Datenbank
    - Datenbank Provider kann relativ einfach gewechselt werden
- Vereinfacht Zugriffsmuster
    - Unit of Work und Transaktion integriert
    - Repository integriert
    - LINQ macht Abfragen lesbar, erweiterbar und transportierbar
- Sonstiges
    - Logging von Queries ist eingebaut
    - Einfache Integration mit anderen .Net Technologien (DI, Asp.Net, dotnet core, Wpf usw.)
    - Standard Technologie die jeder .Net Entwickler kennen sollte

Was ist ein ORM

Als Objekt-Relational-Mapper (ORM) bezeichnet man Technologien wie EF u.a., die Datenbank Entitäten (Tabellen oder auch Relationen genannt) auf Objekte in objektorientiertem Code mappen.

Das bedeutet wir transferieren die Daten von jeder Reihe einer Datenbanktabelle in ein Objekt. Jede Spalte wird dabei zu einer Property.

Warum brauchen wir dazu nun eine eigene Technologie?

- Objekte und Tabellen verwalten ihre Abhängigkeiten unterschiedlich
- Das führt zum sogenannten Objekt relationalen impedanz Mismatch.

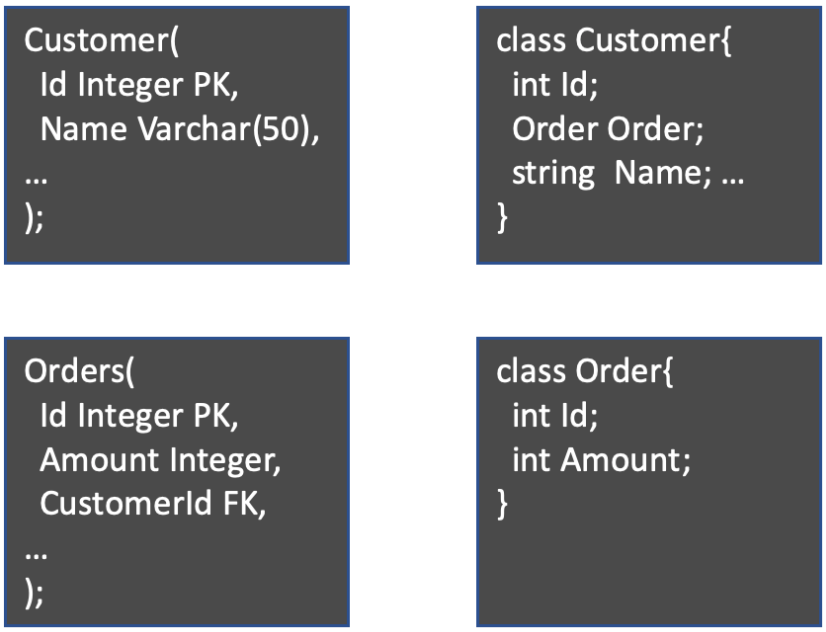

Abbildung 1 - Objekt relationale Impedanz Mismatch

Dieser ergibt sich daraus, dass das Objekt Customer eine Referenz zur Order hat, während in den dazugehörigen Tabellen (links) die **Ordertabelle** per **Fremdschlüssel (FK)** auf den **Primarschlüssel (PK)** der **Customer-Tabelle** verweist.

Um das zu überbrücken, gibt es die sogenannten ORMs und Entity Framework ist einer davon.

Im Kurs lernen wir erheblich mehr über die Funktionen aber zu einem ORM versteht man heute mehr als nur das Mapping:

- Verwalten von Datenbanksessions
- Verwalten von Änderungsmanagement der Entitäten
- Generieren von SQL
- Verwalten vom Datenbankschema

Und EF übernimmt alle diese Funktionen und noch mehr.

Nachteile und Alternativen

Nachteile hat jede Technologie und auch EF ist davon nicht ausgenommen:

- Performance
  - Abstraktion bringt immer Performanceeinbußen mit sich
  - Nicht alle generierten Joins sind perfekt (insebsondere bei tiefen dependencies von Objekten)
  - SQL Optimierungen hängen stark vom Datenbank Provider ab und wie diese für EF implementiert sind
- Initiales Setup kann sehr aufwändig sein
  - Einrichten der Entities
  - Migrationen verwalten und schreiben
- Technologie und Konzepte müssen verstanden werden
  - DbContext und seine Funktionen
  - Changetracking und Queries
  - Limitationen im Datenbankschema
  - Verwendung des jeweiligen Providers
- Oft wird EF missverstanden
  - Nutzen als SQL Ersatz (das man es nicht lernen muss damit)
  - Queries und Setup ohne die DB im Hinterkopf zu behalten

Alternativen (nicht alle Kostenlos!)

- LINQ to Db
- DevExpress XPo
- SQL+.Net
- NHibernate
- Micro ORMs (nur das Mapping, Rest der Features wird weggelassen, auch weniger verwaltungsaufwand dann)
  - Dapper (sehr beliebt)
  - Massive

Ich persönlich empfehle immer EF core aus den folgenden Gründen:

- Weit verbreitet
- Open source, Microsoft gepflegt
- Herausragendes Feature set
- Flexibel
- Einfach zu verstehen und durchdacht

Ansonsten kann ich aus persönlicher Erfahrung noch zu Dapper raten, wenn weniger overhead gewünscht ist und SQL proficiency kein Problem darstellt.

Mehr zu diesem Thema in meinem (englischsprachigen Kurs zu Datenzugrifsstechnologien mit .Net)

Entity Framework Core effektiv einsetzen

Grundlagen