Entity Framework Core: Is It Ready Yet?

Tomas Herceg
Microsoft MVP
Microsoft Regional Director

tomas.herceg@riganti.cz
@hercegtomas

riganti

DOTVVM

Agenda

- Why New Version
- What's New
- Demos
- Roadmap

Entity Framework Core

- Lightweight
- Extensible
- Cross Platform
  - Full .NET (Windows Only)
  - .NET Core (All Platforms)

- Renamed from „Entity Framework 7“
- Revolution instead of evolution

Why New Version?

- Old codebase
  - Not portable easily

- Two APIs
  - `DbContext` (Code First)
  - `ObjectContext` (legacy)

  - Some features only available using the old API
    - `((IObjectContextAdapter)dbContext).ObjectContext`

Why New Version?

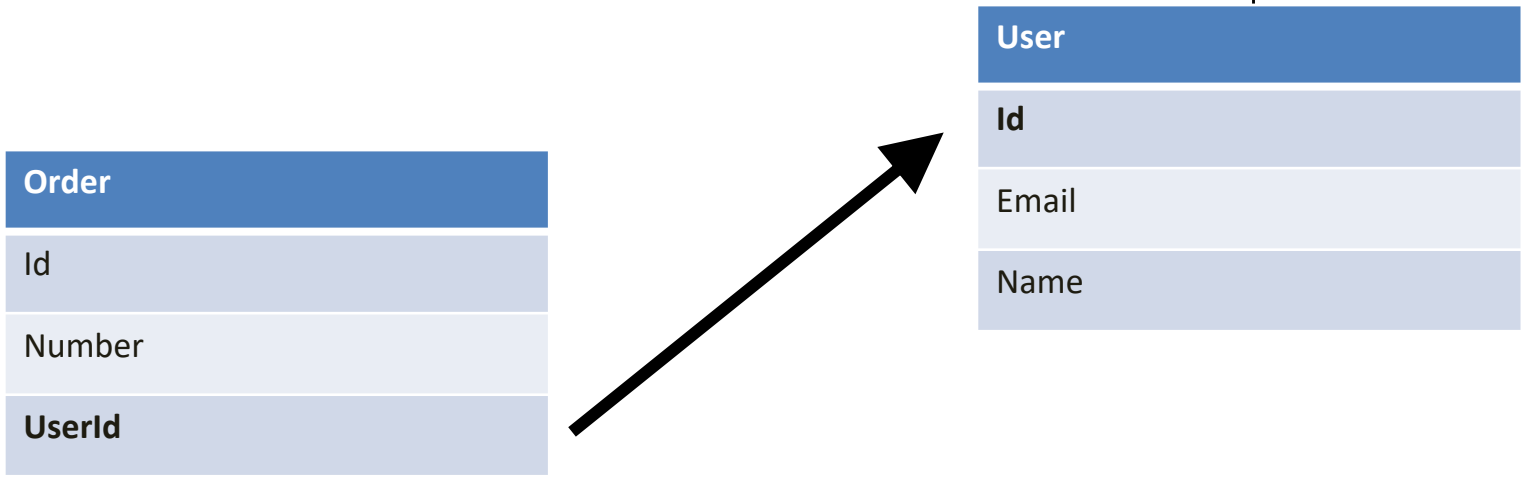
- Monolithic design
  - Difficult to extend (e.g. Hierarchy ID support)
    - Many switch and if statements to change to support new types
- High memory footprint
  - Problem on mobile devices

New Features

- Batching
  - INSERT, UPDATE & DELETE done in batches
- Non-relational Data Store Support Announced
  - Azure Table Storage
  - Redis
- In-Memory Store for Testing

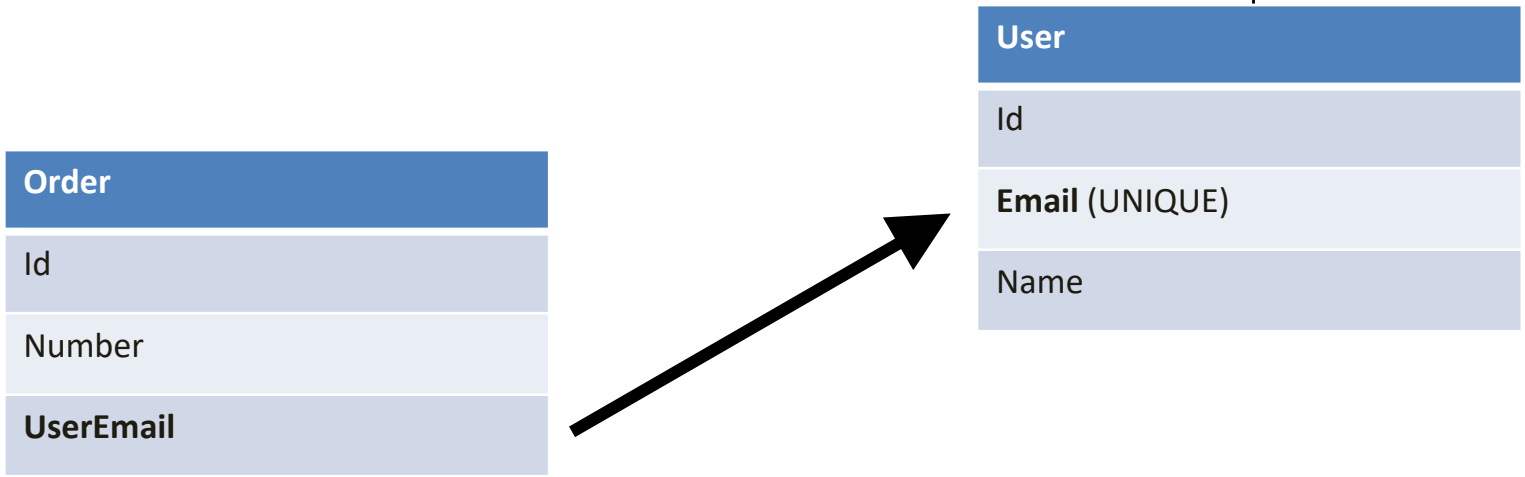
New Features

- Alternate Keys
  - Can be used as one end of the relationship



New Features

- Alternate Keys
  - Can be used as one end of the relationship



New Features

- More Complex Querying Mechanisms
  - In EF6, the whole LINQ query must be translated to SQL
  - In EF Core, the provider can decide what will be translated and what cannot

Client Evaluation

```
from p in db.Products
where !p.Discontinued
select new
{
  Name = p.Name,
  ItemsOnStock = GetDisplayText(p.ItemsOnStock)
}
```

- EF Core can evaluate the **GetDisplayText** on the client

Client Evaluation

- Considerations
  - It can slow down the query!
  - Produces warnings

- https://docs.efproject.net/en/latest/querying/client-eval.html

Code First

- Not a really good name
- „Code-Based Modeling“

- The model can be written from scratch, or generated from database

Code First

- The only supported way for Entity Framework Core
  - No EDMX
  - No designer in Visual Studio

- Scaffolding model from existing database
  - CLI
    - `dotnet ef dbcontext scaffold [connectString] [provider]`
  - Package Manager Console
    - `Scaffold-DbContext –connection … -provider … -outputDir …`

What's Missing

- Dropped
  - Old ObjectContext API
- Not Implemented Yet
  - TPC and TPT Inheritance types
  - Many to many without join entity
  - Spatial data
  - Lazy Loading
  - Stored Procedures

Things That Remain The Same

- LINQ
- DbSet<TEntity>
- DbSet.Add
- DbSet.Remove
- DbContext
- DbContext.Database
- DbContext.ChangeTracker

```
using (var db = new BlogContext())
{
    db.Blogs.Add(new Blog
    {
        Url = "..."
    });
    db.SaveChanges();

    var blogs = from b in db.Blogs
                            .Include(b => b.Posts)
                orderby b.Name
                select b;

    foreach (var blog in blogs)
    {
        Console.WriteLine(blog.Name);
        foreach (var post in blog.Posts)
        {
            Console.WriteLine(post.Title);
        }
    }
}
```

DEMO

Entity Framework Core

Database Schema Changes

- EDMX: Update Model From Database
  - Not very reliable
  - Changes to the model sometimes dissapear
  - Difficult merges

- Code First Migrations in EF6
  - Problems in team environment
  - Merge conflicts in RESX files

Database Schema Changes

- EF Core Migrations
  - CLI
    - `dotnet ef migrations add {name}`
    - `dotnet ef database update`
  - Package Manager Console
    - `Add-Migration`
    - `Update-Database`

DEMO

Entity Framework Core – Migrations

Providers

- Microsoft SQL Server
- SQLite
- PostgreSQL
- MySQL
- Microsoft SQL Server Compact
- IBM Data Servers
- InMemory
- Devart (MySQL, Oracle, PostgreSQL, SQLite, DB2, SQL Server and more)
- Oracle (Coming Soon)

DEMO

Entity Framework Core on Linux

EF Core 1.1

- Improved LINQ translation
- Connection Resiliency
- Explicit Loading

- `DbSet.Find`
- EntityEntry API
  - `Reload`
  - `GetModifiedProperties`
  - `GetDatabaseValues`

EF Core 2.0

- Improved LINQ querying
  - Unnecessarily create nested subqueries
  - Switch prematurely to client-side evaluation
  - Retrieve all columns of a table when only a few were requested
  - Translate a single LINQ query into N+1 queries, sometimes without appropriate filters

EF Core 2.0

- `EF.Functions` class with useful functions (`Like` etc.)
- Owned Entities and Table Splitting
  - Managing identity for complex types without keys
- Global Query Filters
  - ```
    modelBuilder.Entity<Order>()
        .HasQueryFilter(o => o.UserId == CurrentUserId);
    ```
- DbContext pooling
- Manual compiled queries using `EF.CompileQuery`
    ```

DEMO

Query Filters

To upgrade or not to upgrade?

- New platforms (UWP, .NET Core on Linux)
  - Try it
- Existing applications with EF6
  - No need to upgrade
- Review the feature comparison
  - https://docs.microsoft.com/en-us/ef/efcore-and-ef6/features

Q&A

Tomas Herceg
Microsoft MVP
Microsoft Regional Director

tomas.herceg@riganti.cz
@hercegtomas

riganti

DOTVVM