PREGRADO

UNIDAD 3 | WEB SERVICES

ENTITY FRAMEWORK CORE

DATA ANNOTATIONS & FLUENT API

SI653 | Aplicaciones Web

Al finalizar la semana, el estudiante comunica resultados y proceso ágil colaborativo aplicado para la implementación de componentes para las capas de interfaz y servicio, de una aplicación de lado servidor, con características innovadoras, bajo una arquitectura orientada a servicios y aplicando los principios RESTful utilizando el lenguaje C# y Microsoft .NET Framework.

AGENDA

INTRO

CONFIGURATION METHODS

COMPARISON

Introducción

Entity Framework (EF) Core es una versión ligera, extensible, open source y multi-plataforma de la tecnología de acceso a datos Entity Framerwork.

EF Core sirve como object-relational mapper (O/RM), permitiendo utilizar .NET Objects para trabajar con bases de datos, eliminando la necesidad de describir código de acceso a datos.

Cómo opera EF Core

Utiliza un conjunto de **convenciones** para compilar un modelo basado en la forma de las clases de entidad. Puede especificar una configuración adicional para complementar o reemplazar lo que se ha detectado por convención.

La configuración se puede aplicar a un modelo para cualquier almacén de datos y que se puede aplicar al elegir como destino cualquier base de datos relacional. Los proveedores también pueden habilitar la configuración específica de un almacén de datos determinado.

Data Providers

NuGet Package	Supported database engines	Maintainer / Vendor	Notes / Requirements
Microsoft.EntityFrameworkCore.SqlServer	SQL Server 2008 onwards	EF Core Project(Microsoft)	
Microsoft.EntityFrameworkCore.Sqlite	SQLite 3.7 onwards	EF Core Project(Microsoft)	
Microsoft.EntityFrameworkCore.InMemory	EF Core in-memory database	EF Core Project(Microsoft)	For testing only
Microsoft.EntityFrameworkCore.Cosmos	Azure Cosmos DB SQL API	EF Core Project(Microsoft)	Preview only
Npgsql.EntityFrameworkCore.PostgreSQL	PostgreSQL	Npgsql Development Team	
Pomelo.EntityFrameworkCore.MySql	MySQL, MariaDB	Pomelo Foundation Project	
Pomelo.EntityFrameworkCore.MyCat	MyCAT Server	Pomelo Foundation Project	Prerelease only
EntityFrameworkCore.SqlServerCompact40	SQL Server Compact 4.0	Erik Ejlskov Jensen	.NET Framework
EntityFrameworkCore.SqlServerCompact35	SQL Server Compact 3.5	Erik Ejlskov Jensen	.NET Framework
FirebirdSql.EntityFrameworkCore.Firebird	Firebird 2.5 and 3.x	Jiří Činčura	
EntityFrameworkCore.FirebirdSql	Firebird 2.5 and 3.x	Rafael Almeida	
MySql.Data.EntityFrameworkCore	MySQL	MySQL project (Oracle)	
Oracle.EntityFrameworkCore	Oracle DB 11.2 onwards	Oracle	Prerelease
IBM.EntityFrameworkCore	Db2, Informix	IBM	Windows version
IBM.EntityFrameworkCore-lnx	Db2, Informix	IBM	Linux version
IBM.EntityFrameworkCore-osx	Db2, Informix	IBM	macOS version
EntityFrameworkCore.Jet	Microsoft Access files	Bubi	.NET Framework
EntityFrameworkCore.OpenEdge	Progress OpenEdge	Alex Wiese	
Devart.Data.Oracle.EFCore	Oracle DB 9.2.0.4 onwards	DevArt	Paid
Devart.Data.PostgreSql.EFCore	PostgreSQL 8.0 onwards	DevArt	Paid
Devart.Data.SQLite.EFCore	SQLite 3 onwards	DevArt	Paid
Devart.Data.MySql.EFCore	MySQL 5 onwards	DevArt	Paid

AGENDA

INTRO

CONFIGURATION METHODS

COMPARISON

Configurar un modelo

Existen 2 métodos de configuración de un modelo:

- **Fluent API** : Es el método más eficaz de configuración y permite especificar la configuración sin modificar las clases de entidad. La configuración de Fluent API tiene la prioridad más alta y reemplaza las anotaciones de datos y las convenciones.
- **DataAnnotation attributes:** Puede aplicar atributos (conocidos como anotaciones de datos) a las clases y las propiedades. Las anotaciones de datos reemplazarán a las convenciones, pero la configuración de la Fluent API también las reemplazará.

DataAnnotation attributes

System.ComponentModel.DataAnnotations.Schema attributes

Attribute	Description
Table	The database table and/or schema that a class is mapped to.
Column	The database column that a property is mapped to.
ForeignKey	Specifies the property is used as a foreign key in a relationship.
DatabaseGenerated	Specifies how the database generates values for a property.
NotMapped	Applied to properties or classes that are to be excluded from database mapping.
InverseProperty	Specifies the inverse of a navigation property
ComplexType	Denotes that the class is a complex type. *Not currently implemented in EF Core.

DataAnnotation attributes

System.ComponentModel.Annotations attributes

Attribute	Description
Key	Identifies one or more properties as a Key
Timestamp	Specifies the data type of the database column as rowversion
ConcurrencyCheck	Specifies that the property is included in concurrency checks
Required	Specifies that the property's value is required
MaxLength	Sets the maximum allowed length of the property value (string or array)
StringLength	Sets the maximum allowed length of the property value (string or array)

AGENDA

INTRO

CONFIGURATION METHODS

COMPARISON

Ejemplo: Uso de Fluent API para configurar un modelo

```
using Microsoft.EntityFrameworkCore;

namespace EFModeling.Configuring.FluentAPI.Samples.Required
{
    class MyContext : DbContext
    {
        public DbSet<Blog> Blogs { get; set; }
        protected override void OnModelCreating(ModelBuilder modelBuilder)
        {
            modelBuilder.Entity<Blog>()
                .Property(b => b.Url)
                .IsRequired();
        }
    }
    public class Blog
    {
        public int BlogId { get; set; }
        public string Url { get; set; }
    }
}
```

Ejemplo: Uso de DataAnnotation attributes para configurar un modelo

```
using Microsoft.EntityFrameworkCore;
using System.ComponentModel.DataAnnotations;

namespace EFModeling.Configuring.DataAnnotations.Samples.Required
{
    class MyContext : DbContext
    {
        public DbSet<Blog> Blogs { get; set; }
    }
    public class Blog
    {
        public int BlogId { get; set; }
        [Required]
        public string Url { get; set; }
    }
}
```

Inclusión y exclusión de tipos

DataAnnotation attributes

```
class MyContext : DbContext
{
    public DbSet<Blog> Blogs { get; set; }
}
public class Blog
{
    public int BlogId { get; set; }
    public string Url { get; set; }
    public BlogMetadata Metadata { get; set; }
}
[NotMapped]
public class BlogMetadata
{
    public DateTime LoadedFromDatabase { get; set; }
}
```

Fluent API

```
class MyContext : DbContext
{
    public DbSet<Blog> Blogs { get; set; }

    protected override void OnModelCreating(ModelBuilder modelBuilder)
    {
        modelBuilder.Ignore<BlogMetadata>();
    }
}
```

Inclusión y exclusión de propiedades

DataAnnotation attributes

```
class MyContext : DbContext
{
    public DbSet<Blog> Blogs { get; set; }
}
public class Blog
{
    public int BlogId { get; set; }
    public string Url { get; set; }

    [NotMapped]
    public DateTime LoadedFromDatabase { get; set; }
}
```

Fluent API

```
class MyContext : DbContext
{
    public DbSet<Blog> Blogs { get; set; }

    protected override void OnModelCreating(ModelBuilder modelBuilder)
    {
        modelBuilder.Entity<Blog>()
            .Ignore(b => b.LoadedFromDatabase);
    }
}
```

Claves (principal)

Convention

```
class Car
{
    public string Id { get; set; }
    public string Make { get; set; }
    public string Model { get; set; }
}
```

```
class Car
{
    public string CarId { get; set; }
    public string Make { get; set; }
    public string Model { get; set; }
}
```

DataAnnotation attributes

```
class Car
{
    [Key]
    public string LicensePlate { get; set; }
    public string Make { get; set; }
    public string Model { get; set; }
}
```

Fluent API

```
class MyContext : DbContext
{
    public DbSet<Car> Cars { get; set; }
    protected override void OnModelCreating(ModelBuilder modelBuilder)
    {
        modelBuilder.Entity<Car>()
            .HasKey(c => c.LicensePlate);
    }
}
```

```
class Car
{
    public string LicensePlate { get; set; }
    public string Make { get; set; }
    public string Model { get; set; }
}
```

Sin generación de valor

DataAnnotation attributes

```
public class Blog
{
    [DatabaseGenerated(DatabaseGeneratedOption.None)]
    public int BlogId { get; set; }
    public string Url { get; set; }
}
```

Fluent API

```
protected override void OnModelCreating(ModelBuilder modelBuilder)
{
    modelBuilder.Entity<Blog>()
        .Property(b => b.BlogId)
        .ValueGeneratedNever();
}
```

Longitud máxima

DataAnnotation attributes

```
class MyContext : DbContext
{
    public DbSet<Blog> Blogs { get; set; }
}

public class Blog
{
    public int BlogId { get; set; }
    [MaxLength(500)]
    public string Url { get; set; }
}
```

Fluent API

```
class MyContext : DbContext
{
    public DbSet<Blog> Blogs { get; set; }

    protected override void OnModelCreating(ModelBuilder modelBuilder)
    {
        modelBuilder.Entity<Blog>()
            .Property(b => b.Url)
            .HasMaxLength(500);
    }
}
public class Blog
{
    public int BlogId { get; set; }
    public string Url { get; set; }
}
```

Relaciones

Conventions

```
public class Blog
{
    public int BlogId { get; set; }
    public string Url { get; set; }

    public List<Post> Posts { get; set; }
}
```

```
public class Post
    {
        public int PostId { get; set; }
        public string Title { get; set; }
        public string Content { get; set; }

        public int BlogId { get; set; }
        public Blog Blog { get; set; }
    }
```

DataAnnotation attributes

```
public class Blog
{
    public int BlogId { get; set; }
    public string Url { get; set; }

    public List<Post> Posts { get; set; }
}
```

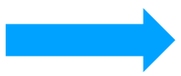
```
public class Post
    {
        public int PostId { get; set; }
        public string Title { get; set; }
        public string Content { get; set; }

        public int BlogForeignKey { get; set; }

        [ForeignKey("BlogForeignKey")]
        public Blog Blog { get; set; }
    }
```

Relaciones

Fluent API



```
class MyContext : DbContext
{
    public DbSet<Blog> Blogs { get; set; }
    public DbSet<Post> Posts { get; set; }

    protected override void OnModelCreating(ModelBuilder modelBuilder)
    {
        modelBuilder.Entity<Post>()
            .HasOne(p => p.Blog)
            .WithMany(b => b.Posts);
    }
}

public class Blog
{
    public int BlogId { get; set; }
    public string Url { get; set; }

    public List<Post> Posts { get; set; }
}
public class Post
{
    public int PostId { get; set; }
    public string Title { get; set; }
    public string Content { get; set; }

    public Blog Blog { get; set; }
}
```

Índices

Fluent API

```
class MyContext : DbContext
{
    public DbSet<Blog> Blogs { get; set; }

    protected override void OnModelCreating(ModelBuilder modelBuilder)
    {
        modelBuilder.Entity<Blog>()
            .HasIndex(b => b.Url)
            .IsUnique();
    }
}
public class Blog
{
    public int BlogId { get; set; }
    public string Url { get; set; }
}
```

```
class MyContext : DbContext
{
    public DbSet<Person> People { get; set; }

    protected override void OnModelCreating(ModelBuilder modelBuilder)
    {
        modelBuilder.Entity<Person>()
            .HasIndex(p => new { p.FirstName, p.LastName });
    }
}
public class Person
{
    public int PersonId { get; set; }
    public string FirstName { get; set; }
    public string LastName { get; set; }
}
```

RESUMEN

Recordemos

Entity Framework Core

- Conventions
- DataAnnotation attributes
- API Fluent

REFERENCIAS

Para profundizar

https://docs.microsoft.com/es-es/ef/#pivot=entityfmwk&panel=entityfmwk1

https://www.entityframeworktutorial.net/efcore/entity-framework-core.aspx

https://docs.microsoft.com/es-es/aspnet/mvc/overview/getting-started/getting-started-with-ef-using-mvc/creating-a-more-complex-data-model-for-an-asp-net-mvc-application

https://learn.microsoft.com/en-us/aspnet/core/data/ef-mvc/intro?view=aspnetcore-6.0

https://www.learnentityframeworkcore.com/configuration/data-annotation-attributes

https://github.com/aspnet/EntityFrameworkCore

PREGRADO

Ingeniería de Software

Escuela de Ingeniería de Sistemas y Computación | Facultad de Ingeniería

UPC
Universidad Peruana
de Ciencias Aplicadas

Prolongación Primavera 2390,
Monterrico, Santiago de Surco
Lima 33 - Perú
T 511 313 3333
https://www.upc.edu.pe

exígete, innova