



NET2 - 11

Entity Framework

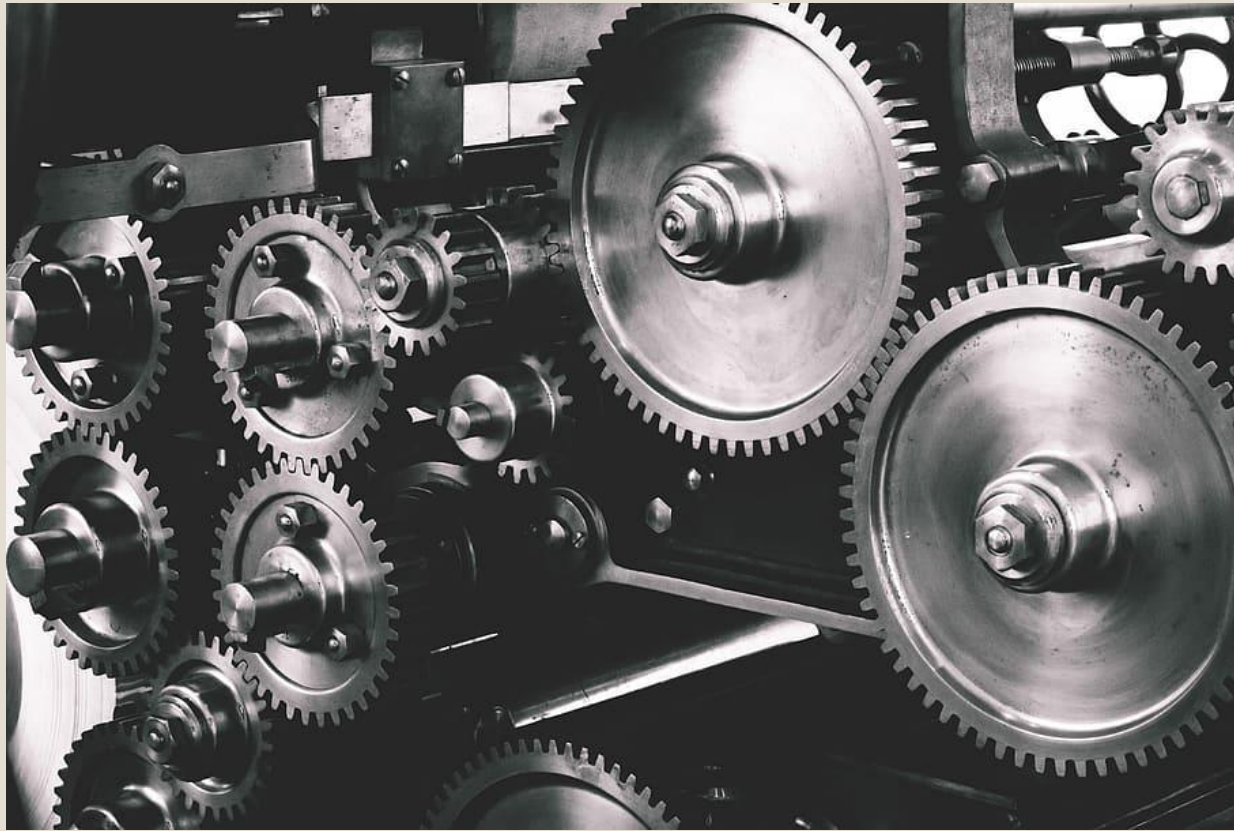
Förra gången

- Redovisning Bokhandeln
- Ansluta till en databas med C#
 - ADO
- Object-relational mapping (ORM)
 - Vad är det?
 - Fördelar och nackdelar
 - Olika ramverk
 - **Dapper**
 - Entity Framework
 - Code-along med Dapper: Parkeringshuset2
 - Övningsuppgift
 - Fortsätta med parkeringshuset

Idag

- Fortsatt arbete med övningen (parkeringsappen)?
- Repetition?
 - Pdf-en?
 - Repetition av Code-along?
- Fredag:
 - Frivillig “redovisning” av parkeringsappen.
 - Entity framework
 - Database First
 - Code First

Entity Framework



Entity Framework

- Ett ramverk för att skapa, läsa och skriva till databaser.
- Istället för att skriva SQL så kan vi koda direkt i C#
- Det vanligaste sättet att skapa databasen är med Code-first
 - Vi skapar alla klasser som vanligt, och låter sedan dotnet skapa databasen åt oss.
- De fyra saker vi vanligen kan göra med en databas kallas för CRUD

Entity Framework

- Entity Framework:
 - Entity Framework (EF) är Microsofts ORM för .NET som släpptes för första gången 2008.
 - Tidigare hade EF bara stöd för Microsofts egna databaser som Microsoft SQL Server. Numera finns det paket som kan installeras som ger stöd för många andra typer av databaser.
 - Våldigt mycket händer under huven med Entity Framework.
 - Den skapar konfigurationer,
 - mappar ut tabeller till objekt
 - ...med mera genom att autogenerera kod med hjälp av några knapptryckningar.
 - Detta gör att en utvecklare sparar mer tid och gör det även lättare för en utvecklare att använda sig av ORM:et då en inte behöver lika stor kunskap om hur allting fungerar.
 - EF får dock en del kritik från utvecklare just på grund av att EF skapar kod automatiskt. Man får mindre kontroll på sin egen kod då man måste förlita sig på kod som skapas av en maskin. Det blir till exempel svårare att debugga koden om man själv inte har skrivit den eller kan förstå sig på den

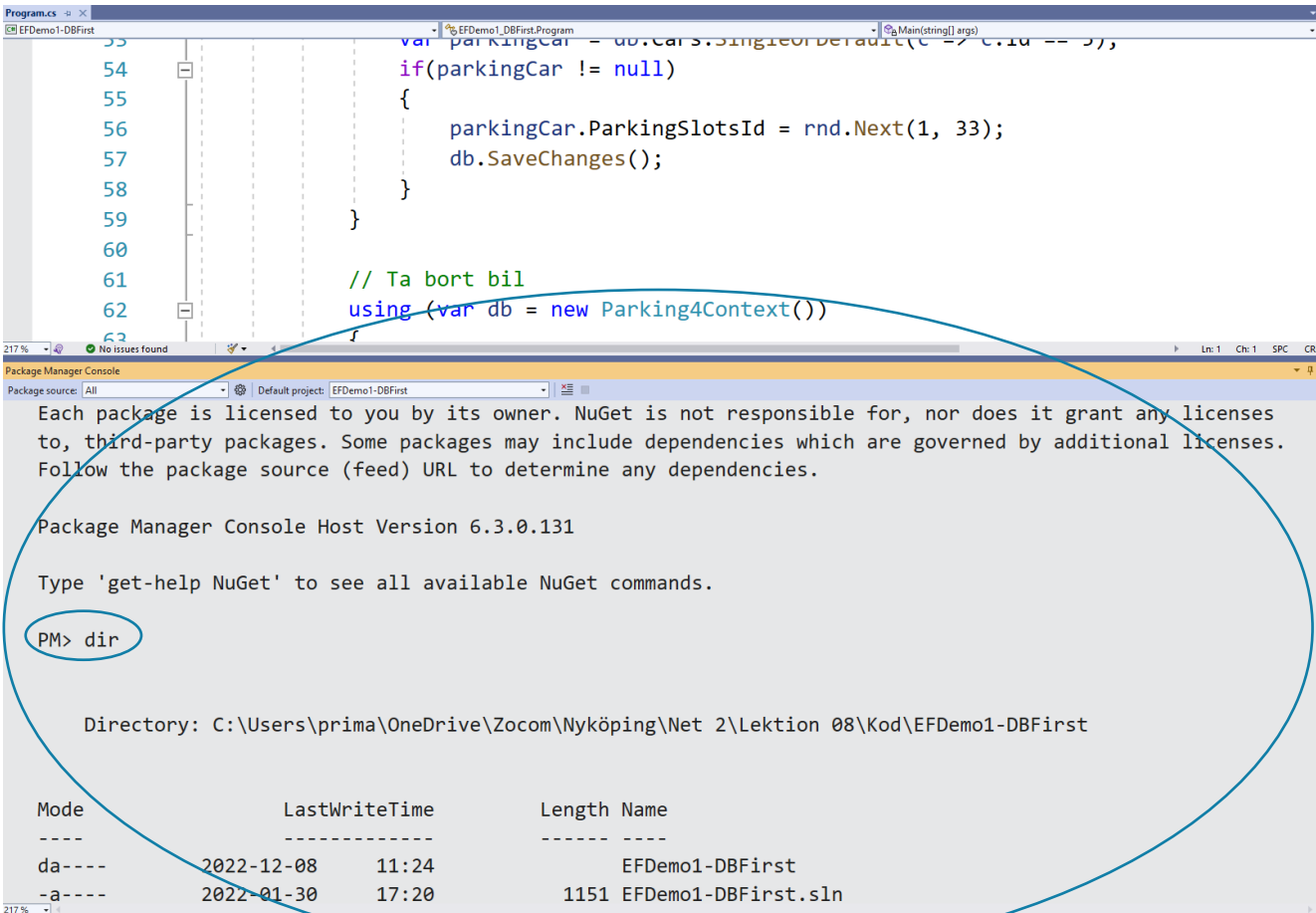
Database first med Entity Framework

- Ofta finns database redan, och vi vill kunna använda den i vår egen application.
- Vi vill då få tabellerna och dess kolumner att bli tillgängliga i vår kod, som objekt.
- Detta gör vi med en sk scaffolding – byggnadsställning.
 - Detta finns inbyggt i EF, och utförs med ett enkelt commando i Packet Manager Console.



Packet manager console

- Package Manager-konsolen i Visual Studio använder PowerShell-kommandon för att interagera med NuGet-paket. Du kan använda konsolen när det inte finns något sätt att utföra en åtgärd via Package Manager-användargränssnittet. Du kan också använda dotnet CLI- eller NuGet CLI-kommandon i konsolen
 - Enkelt förklarat, en kommando-prompt inbyggt i Visual studio.



```
Program.cs
EFDemo1-DBFirst
54
55
56
57
58
59
60
61
62
63

var parkingCar = db.Cars.SingleOrDefault(c => c.Id == 5);
if(parkingCar != null)
{
    parkingCar.ParkingSlotsId = rnd.Next(1, 33);
    db.SaveChanges();
}

// Ta bort bil
using (var db = new Parking4Context())
```

```
Package Manager Console
Package source: All
Default project: EFDemo1-DBFirst

Each package is licensed to you by its owner. NuGet is not responsible for, nor does it grant any licenses to, third-party packages. Some packages may include dependencies which are governed by additional licenses. Follow the package source (feed) URL to determine any dependencies.

Package Manager Console Host Version 6.3.0.131

Type 'get-help NuGet' to see all available NuGet commands.

PM> dir

Directory: C:\Users\prima\OneDrive\Zocom\Nyköping\Net 2\Lektion 08\Kod\EFDemo1-DBFirst

Mode                LastWriteTime         Length Name
----                -
da----          2022-12-08    11:24             EFDemo1-DBFirst
-a----          2022-01-30    17:20          1151 EFDemo1-DBFirst.sln
```


Paket: Entity Framework

The image shows the NuGet Package Manager and Solution Explorer in Visual Studio. The NuGet Package Manager is displaying the search results for 'entityframework'. The package 'Microsoft.EntityFrameworkCore' is selected, and the version '6.0.11' is highlighted with a red circle. The Solution Explorer shows the project 'EF-Demo-DatabaseFirst1' with the package 'Microsoft.EntityFrameworkCore (6.0.11)' installed.

NuGet Package Manager: EF-Demo-DatabaseFirst1

Package source: [nuget.org](https://www.nuget.org)

Microsoft.EntityFrameworkCore by Microsoft, 561M downloads

Entity Framework Core is a modern object-database mapper for .NET. It supports LINQ queries, change tracking, updates, and schema migrations. EF Core works with SQL Server, Azure SQL Database, SQLite, Azure Cosmos DB, MySQL, PostgreSQL, and other databases t...

Version: **6.0.11** Install

Options

Description

Entity Framework Core is a modern object-database mapper for .NET. It supports LINQ queries, change tracking, updates, and schema migrations. EF Core works with SQL Server, Azure SQL Database, SQLite, Azure Cosmos DB, MySQL, PostgreSQL, and other databases through a provider plugin API.

Commonly Used Types:

- Microsoft.EntityFrameworkCore.DbContext
- Microsoft.EntityFrameworkCore.DbSet

Version: 6.0.11

Author(s): Microsoft

License: MIT

Date published: Monday, November 7, 2022 (11/7/2022)

Project URL: <https://docs.microsoft.com/ef/core/>

Report Abuse: <https://www.nuget.org/packages/>

Solution Explorer

Search Solution Explorer (Ctrl+)

Solution 'EF-Demo-DatabaseFirst1' (1 of 1 project)

- EF-Demo-DatabaseFirst1
 - Dependencies
 - Analyzers
 - Frameworks
 - Packages
 - Microsoft.EntityFrameworkCore (6.0.11)
 - Program.cs

Paket: Entity Framework Tools

The screenshot displays the NuGet Package Manager interface within a Visual Studio environment. The top bar shows the current project as 'Program.cs' and the package source as 'nuget.org'. The search bar contains 'entityframeworkcore'. The results list several packages, with 'Microsoft.EntityFrameworkCore.Tools' highlighted. The right-hand pane shows the details for this package, including its version (6.0.11), author (Microsoft), license (MIT), and a list of commands it enables. The Solution Explorer at the bottom right shows the project structure, including the 'EF-Demo-DatabaseFirst1' project and its dependencies.

NuGet Package Manager: EF-Demo-DatabaseFirst1

entityframeworkcore ☐ Include prerelease

Package source: **nuget.org**

Package Name	Author	Downloads	Version
Microsoft.EntityFrameworkCore.Relational	by Microsoft	550M	7.0.0
Microsoft.EntityFrameworkCore.Abstractions	by Microsoft	521M	7.0.0
Microsoft.EntityFrameworkCore.Analyzers	by Microsoft	504M	7.0.0
Microsoft.EntityFrameworkCore.Design	by Microsoft	262M	7.0.0
Microsoft.EntityFrameworkCore.SqlServer	by Microsoft	272M	7.0.0
Microsoft.EntityFrameworkCore.Tools	by Microsoft	184M	7.0.0
Microsoft.EntityFrameworkCore.InMemory	by Microsoft	137M	7.0.0

Microsoft.EntityFrameworkCore.Tools

Version: **6.0.11**

Options

Description

Entity Framework Core Tools for the NuGet Package Manager Console in Visual Studio.

Enables these commonly used commands:

- Add-Migration
- Bundle-Migration
- Drop-Database
- Get-DbContext
- Get-Migration
- Optimize-DbContext
- Remove-Migration
- Scaffold-DbContext
- Script-Migration
- Update-Database

Version: 6.0.11
Author(s): Microsoft
License: MIT
Date published: Monday, November 1, 2021
Project URL: <https://docs.microsoft.com/en-us/ef/core/tools/>

Solution Explorer

Search Solution Explorer (Ctrl+)

Solution 'EF-Demo-DatabaseFirst1' (1 of 1 project)

- EF-Demo-DatabaseFirst1
 - Dependencies
 - Analyzers
 - Frameworks
 - Packages
 - Microsoft.EntityFrameworkCore (6.0.11)
 - Microsoft.EntityFrameworkCore.Tools (6.0.11)
 - C# Program.cs

Paket: SQL Server

The screenshot displays the Visual Studio interface with the NuGet Package Manager and Solution Explorer open.

NuGet Package Manager: EF-Demo-DatabaseFirst1

Package source: nuget.org

Search: ☐ Include prerelease

Package Name	Author	Downloads	Version
Microsoft.EntityFrameworkCore.SqlServer	by Microsoft	272M	7.0.0
EntityFrameworkCore.SqlServer.HierarchyId	by Brice Lambson, et al.	3,46M	4.0.0
EntityFrameworkCore.SqlServer.Abstractions	by Brice Lambson, et al.	3,47M	4.0.0
EntityFrameworkCore.SqlServer.Design	by Microsoft	9,74M	1.1.6
EntityFrameworkCore.SqlServer.TopologySuite	by Microsoft	6,9M	7.0.0
EntityFrameworkCore.SqlServer.Tools	by Jon Davies	81,7K	6.0.9
EntityFrameworkCore.SqlServer.WorkCore	by ErikEJ	94,7K	2.2.0.7

Microsoft.EntityFrameworkCore.SqlServer (7.0.0)

Version:

Options

Description
Microsoft SQL Server database provider for Entity Framework Core.

Version: 6.0.11
Author(s): Microsoft
License: MIT
Date published: Monday, November 7, 2022 (11/7/2022)
Project URL: <https://docs.microsoft.com/ef/core/>
Report Abuse: <https://www.nuget.org/packages/Microsoft.EntityFrameworkCore.SqlServer/6.0.11/ReportAbuse>

Tags: Entity, Framework, Core, entity-framework-core, EF, Data, O/RM, EntityFramework, EntityFrameworkCore, EFCore, SQL, Server

Dependencies

- net6.0
- Microsoft.EntityFrameworkCore.Relational (>= 6.0.11)
- Microsoft.Data.SqlClient (>= 2.1.4)

Solution Explorer

Solution 'EF-Demo-DatabaseFirst1' (1 of 1 project)

- EF-Demo-DatabaseFirst1
 - Dependencies
 - Analyzers
 - Frameworks
 - Packages
 - Microsoft.EntityFrameworkCore (6.0.11)
 - Microsoft.EntityFrameworkCore.SqlServer (6.0.11)
 - Microsoft.EntityFrameworkCore.Tools (6.0.11)
 - C# Program.cs

DbContext och Model

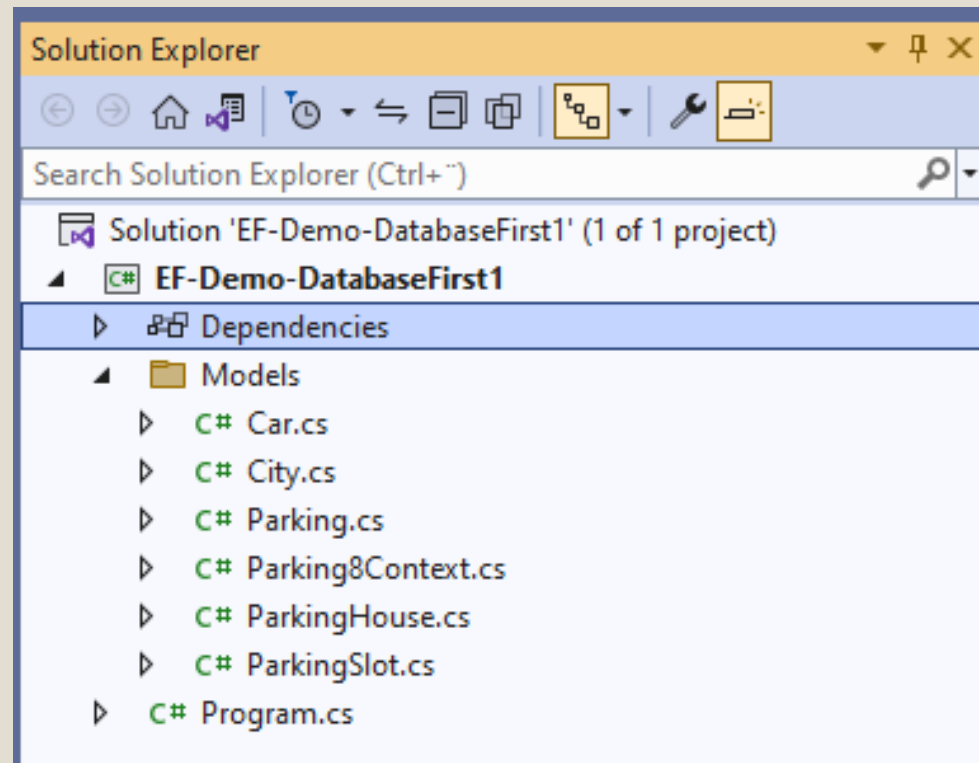
- En DbContext-instans representerar en session med databasen och kan användas för att fråga och spara instanser av dina entiteter.
- DbContext-objektet innehåller hela databasen, och den kan vi använda i vår kod.
- Model är ett annat namn på de klasser vi använder för att beskriva vår verklighet, t ex städer, bilar eller personer.

Skapa DbContext och Models/Klasser

- När vi ska skapa en DbContext och hämta modellerna/tabellerna som finns i databasen så använder vi funktionen "Scaffold-DbContext", som vi kör i Packet Manager Consol
- Den har några inparametrar:
 - **"Server=.\SQLEXPRESS;Database=Parking8;Trusted_Connection=True;"**
 - Server = Vilken databas vi vill scaffolda. Och hur vi ska logga in.
 - Trusted_Connection = windowsinloggningen
- Sedan ska vi ange vilken databastyp
 - **Microsoft.EntityFrameworkCore.SqlServer**
- Sist anger vi i vilken mapp i projektet vi vill spara alla klasser
 - **-OutputDir Models**
- Exempel:
 - Scaffold-DbContext "Scaffold-DbContext
"Server=.\SQLEXPRESS;Database=Parking10;Trusted_Connection=True;TrustServerCertificate=true"
Microsoft.EntityFrameworkCore.SqlServer -
OutputDir Models

```
PM> Scaffold-DbContext "Server=.\SQLEXPRESS;Database=Parking10;Trusted_Connection=True;TrustServerCertificate=true"  
Microsoft.EntityFrameworkCore.SqlServer -OutputDir Models
```

Målet



Använda databasen

- “Ansluta” till databasen:
 - `var db = new Parking8Context();`
 - I db finns nu en instans av hela databasen, men bara kopplingen till databasen, inte data.
- Hämta innehållet i en tabell
 - `var carList = db.Cars;`
 - I carList finns nu en kollektion av alla bilar
- Loopa igenom alla bilar:

```
foreach (var car in carList)
{
    Console.WriteLine(car.Make + "\t" + car.Color + "\t" + car.Plate);
}
```

using

- Genom att använda using så ser vi till att .NET tar hand om alla stängningar, och att variabler, anslutningar och listor hanteras rätt, när de inte behövs längre (garbage collector).
- Skriv därför såhär:

```
using (var db = new Parking8Context())
{
    var carList = db.Cars;

    foreach (var car in carList)
    {
        Console.WriteLine(car.Make + "\t" + car.Color + "\t" + car.Plate);
    }
}
```

EF CRUD

- Fungerar som C#

- Create:

```
carList.Add(newCar);  
Db.SaveChanges();
```

- Read

```
foreach (var car in carList)  
{  
    Console.WriteLine(car.Make + "\t" + car.Color + "\t" + car.Plate);  
}
```

- Update:

```
parkingCar.ParkingSlotsId = 23;  
db.SaveChanges();
```

- Delete

```
db.Cars.Remove(scrapCar);  
db.SaveChanges();
```

Lite om LINQ

- I kursen ska vi prata om mer avancerade programtekniska saker, såsom LINQ
- Men, för att kunna göra lite vettiga urval behöver vi lite smakprov på detta redan nu.
- I den här kursen använder vi ett handfull LINQ-uttryck för urval.
- Behöver vi göra mer avancerade saker kan vi göra det med SQL via Dapper.
- LINQ-Exempel, för att plocka ut en viss bil:
 - `var theCar = (from c in db.Cars`
 - `where c.Id == 5`
 - `select c).SingleOrDefault();`
- Nästa lektion ska vi prata mer LINQ

Code along

- EF-Demo-Databasefirst



Code first

- Med Code first så skapar vi våra klasser som vanligt, och låter sedan Entity Framework bygga databaserna åt oss,
- Bygg applikationen som vanligt, med de klasser/models du behöver.
- När du behöver en databas för din application så gör du en "migration", att flytta.

Skapa klass

- `class Product`
- `{`
- `public int Id { get; set; }`
- `public string Name { get; set; }`
- `public double Price { get; set; }`
- `}`

Skapa dbcontext-klassen

```
class ProductContext : DbContext
{
    public DbSet<Product> Products { get; set; }

    protected override void OnConfiguring(DbContextOptionsBuilder optionsBuilder)
    {
        optionsBuilder.UseSqlServer(@"Server=.\SQLEXPRESS;Database=WebShop;Trusted_Connection=True;");
    }
}
```

Klassen du vill göra en databastabell av

Detta blir tabellnamnet

Namnet på databasen du vill skapa en tabell i.
(Skapas i förväg)

Skapa databastabell

- I Packet Manager
 - Skriv följande:
 - **Add-migration valfri_egen_beskrivning**
 - En mapp skapas, som håller reda på hur databasen ska skapas.
 - Skriv därefter:
 - **Update-database**
- Klart! Nu kan du koda på vanligt sätt.

Paket

NuGet: EF-Demo-CodeFirst1 20221208162932_FirstInit.cs MyDbContext.cs Product.cs Program.cs

[Browse](#) [Installed](#) [Updates](#) **3**

☐ Include prerelease

Microsoft.EntityFrameworkCore by Microsoft
Entity Framework Core is a modern object-database mapper for .NET. It supports LINQ queries, change tracking, updates, and schema migrations. EF Core works with SQL Server, Azure SQL Database, SQLite, Azure Cosmos DB, MySQL, PostgreSQL, and oth...

6.0.11
7.0.0

Microsoft.EntityFrameworkCore.SqlServer by Microsoft
Microsoft SQL Server database provider for Entity Framework Core.

6.0.11
7.0.0

Microsoft.EntityFrameworkCore.Tools by Microsoft
Entity Framework Core Tools for the NuGet Package Manager Console in Visual Studio.

6.0.11
7.0.0

Code Along

- Dokument: EFDemo-CodeFirst1

Övningar

- Övningarnas fokus är att få igång och använda Entity Framework
 - För varje övning, skapa en helt ny console-app, så du repeterar vilka bibliotek du ska lägga till.

Övningar – Database First

- Skapa en app som läser och visar data från Everloop – Products
 - Gör en scaffold av databasen.
 - Gör så att du kan visa produkterna och ändra pris på produkterna. (strunta i att lägga till och ta bort)
- Skapa en app som visar en lista på städer, och vilka parkeringshus som finns i varje stad
 - Gör en scaffold av Parking-databasen.
 - Stockholm
 - Citygaraget
 - Parkaden
 - Göteborg
 - ...
 - Gör INTE övningen med JOINS, utan läs data från de båda tabellerna, och använd två foreach-loopar + LINQ(WHERE)
 - När du loopar två tabbeller i varandra behöver de vara listor, så använd följande: `var cityList = db.Cities.ToList()`
 - Gör ett enkelt admin för att kunna lägga till/ändra städer med hjälp av EF och C#.ul> - När vi vill ändra i databaser får de INTE vara listor. Använd: `var cityList2 = db.Cities`

Övning – Code first

- Code first
 - Todo-appen:
 - Skapa ett object för en Todo-lista med posterna Namn, Ansvarig och Fixat
 - **Med hjälp av Code-first-upplägget låter du EF skapa tabellen, i en ny databas. (Migration: Add-migration och Update-database)**
 - Bygg ett enkelt admin för att kunna lägga till poster i Todo-listan
 - Gör också så att du kan ta bort todo-poster, genom att ange ett Id.
 - Gör också så att du kan ange ett id för att sätta todo-posten som Fixad
 - När appen är klar, lägg till en ny property, med namnet Poäng (varje sak man orkar göra ska ge poäng).
 - Upprepa Migration och update.



Länkar

- [Getting Started with Entity Framework Core: Database-First Development - TechNet Articles - United States \(English\) - TechNet Wiki \(microsoft.com\)](#)
- [Entity Framework Core with Existing Database \(entityframeworktutorial.net\)](#)
- [Install Entity Framework Core \(entityframeworktutorial.net\)](#)
- [First EF Core Console Application \(entityframeworktutorial.net\)](#)