

Entity Framework

Niklas Hjelm

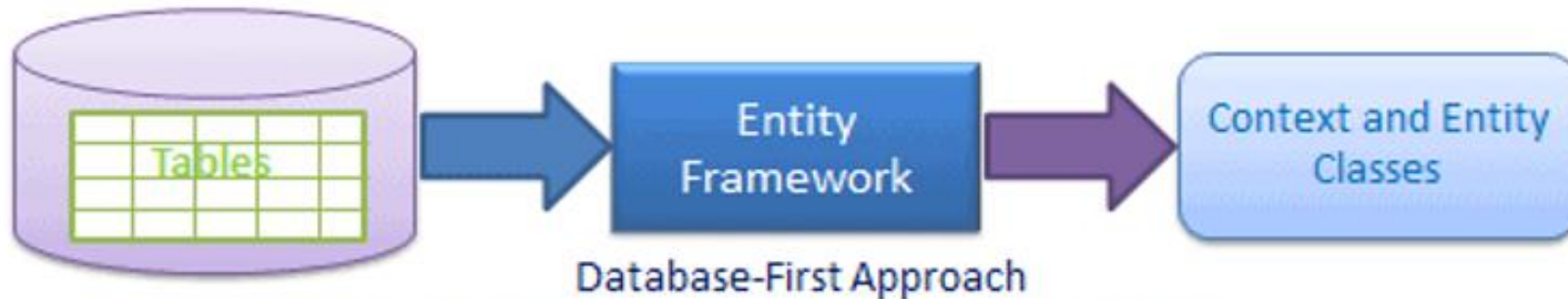


IT-HÖGSKOLAN
GÖTEBORG

Vad är Entity Framework

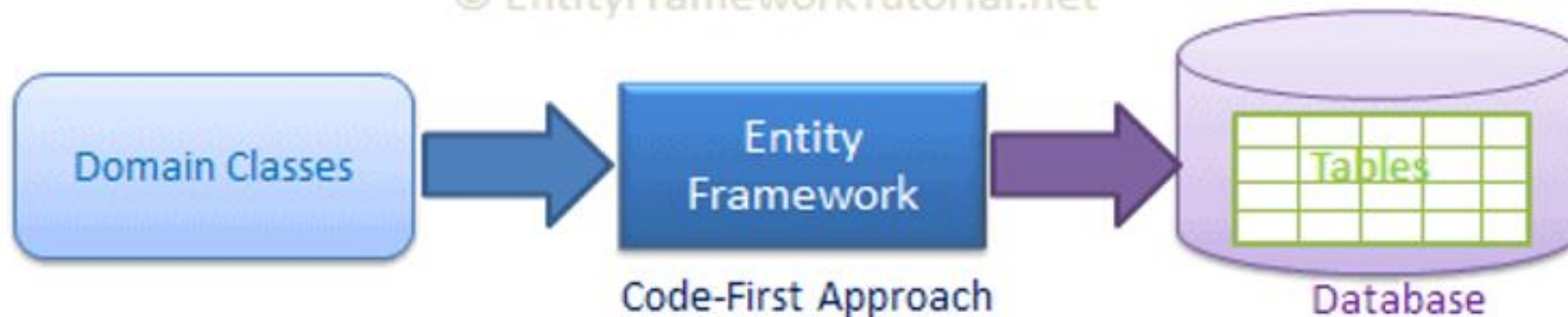
- **Entity Framework** ett sk **ORM**-ramverk för **ADO.NET**.
 - **Object-Relational Mapping**
- **ADO.NET** ger tillgång till databasimplementation i **.NET**.
- **Entity Framework** ger utvecklare möjlighet att...
 - Jobba med domänspecifika objekt och egenskaper.
 - Spåra förändringar i databasmodellen med hjälp av **Migrations**.
 - Använda **LINQ** för att leta efter data.

Code First och Database First



Generate Data Access Classes for Existing Database

© EntityFrameworkTutorial.net



Create Database from the Domain Classes

Code First

- Entity Framework kan skapa databaser och Tabeller efter klasser skrivna i C#.
- Relationer och junction tables skapas efter de relationer vi satt upp mellan våra objekt.
- Möjlighet att ytterligare påverka hur tabeller sätts upp i särskilda konfigurationsklasser

Database First

- Entity Framework kan generera klasser med relevanta relationer utifrån en databas med tabeller.
- Möjlighet att bygga en applikation som kan nyttja en befintlig databas med befintliga tabeller och innehåll.
- Kan i vissa fall kräva vissa förändringar i databasen eller i den genererade koden.

Saker vi behöver känna till i C#


- **Attribut** – "taggar" som ger information till kompilatorn om ytterligare egenskaper hos en Property eller Metod.
- **LINQ** – Centralt för att ställa queries till en relationsdatabas i C#
- **IEnumerable<T>** - Grundtypen för listor i EntityFramework.
- **Lambda** - Används i LINQ för queries.
- **Klasser och Objekt** - Det är detta som används för att representera data från databasen


Entity Model

- Till höger ser vi exempel på två klasser som vi vill skapa en relation emellan:

```
public class Student
{
    public int StudentId { get; set; }
    public string StudentName { get; set; }
}
```

```
public class Grade
{
    public int GradeId { get; set; }
    public string GradeName { get; set; }
    public string Section { get; set; }
}
```

<div>  Student </div>	
PK	<u>Id</u>
	StudentName

<div>  Grade </div>	
PK	<u>Gradeld</u>
	GradeName
	Section

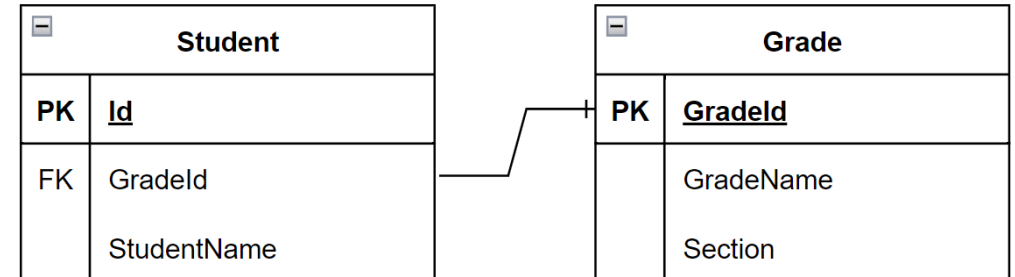
One-to-One

```
public class Student
{
    public int Id { get; set; }
    public string Name { get; set; }

    public Grade Grade { get; set; }
}

public class Grade
{
    public int GradeId { get; set; }
    public string GradeName { get; set; }
    public string Section { get; set; }
}
```

Genom att lägga till en property av typen Grade i klassen Student så kommer Entity Framework skapa en relation mellan tabellerna.



DbContext

- DbContext är en basklass som låter oss skapa klasser där vi kan göra följande:
 - Ansluta mot en databas
 - Konfigurera modeller och relationer
 - Göra Queries till databasen
 - Spara data i databasen
 - Ställa in spårning av förändringar
 - Transaktionshantering

Attribut

- En typ av tagg som ger kompilatorn ytterligare information om hur en klass, property eller metod ska användas.
- Data Annotations är ett namespace som har många sådana för EntityFramework.
- Detta kan användas för att konfigurera en entitet.
- Syntax: **[AttributNamn]**

```
public class Book
{
    [Key]
    [StringLength(13)]
    0 references
    public string ISBN { get; set; }
```

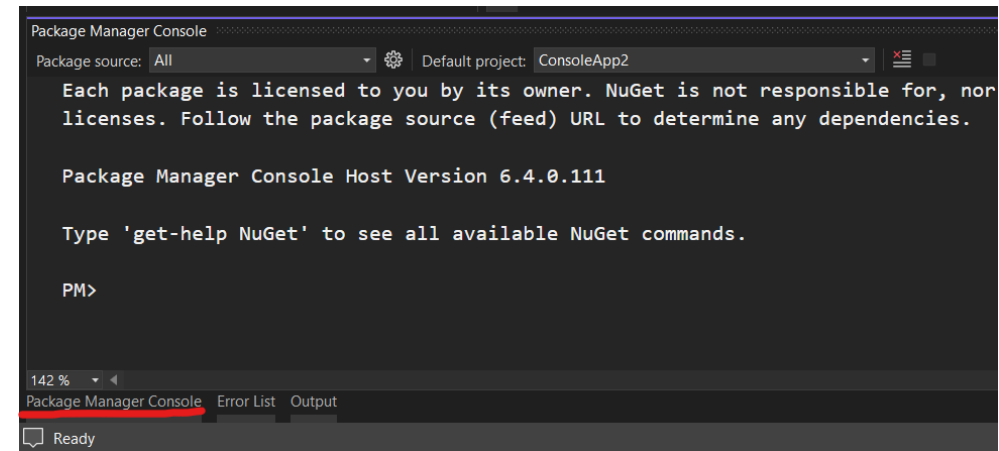
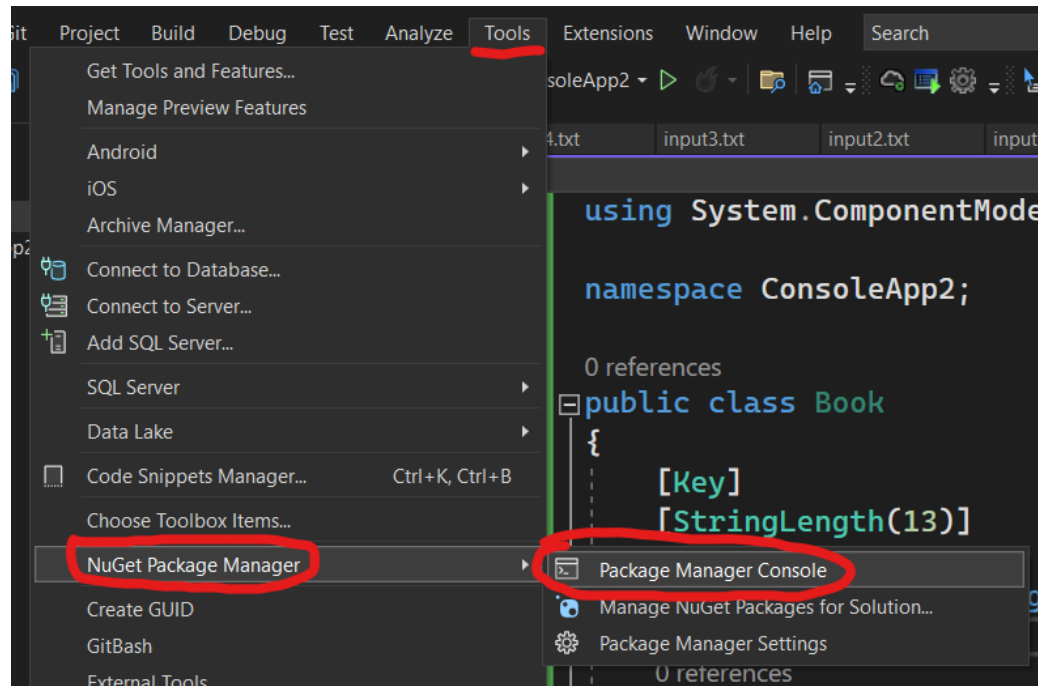
Fluent API

- Fluent API är ett sätt som man kan konfigurera databasmodeller med som ger mer valmöjligheter och kontroll över detaljer än om man använder sig utav Attribut.
- Detta görs i en s.k. ModelBuilder inne i DbContext.

CLI – Command Line Interface

- Ett CLI kan användas för att interagera med olika program genom en terminal.
 - cmd
 - bash
 - powershell
- CLI:er som är värda att känna till:
 - Git
 - Dotnet
 - NuGet Package Manager Console

NuGet Package Manager Console



Kommandon för Entity Framework

//Create first migration called "init"

add-migration "init"

//if multiple contexts exist

add-migration "init" -context [Ditt context name]

//For database first

Scaffold-DbContext [ConnectionString]

Microsoft.EntityFrameworkCore.SqlServer

-OutputDir [Mapp för databasfiler]

Connection String

- `[Data Source|Server]=[ServerNamn];`
- `Database=[DatabasNamn]`
- `Integrated Security=[True|False];`
- `Connect Timeout=[integer];`
- `Encrypt=[True|False];`
- `TrustServerCertificate=[True|False];`
- `ApplicationIntent=[ReadWrite|Read|Write];`
- `MultiSubnetFailover=[True|False]`

Resten av dagen

- Paus 15 min
- Demo Database First
- Lunch
- Mer demo Database First
- Paus
- Intro Labb 2