# NET2 - 14

Mer EF – Många till många

# Förra gången

- Kursinventering
- Blandat
  - Enum
  - Try/Catch
  - Svenska tecken
- EF-Demo
  - Code First
    - Relationer och Constraints

# Idag

- Mer EF
  - Många till många
- Code Along
  - Filmer och skådespelare
  - Bonus: “Fönster” I konsollen

# Inte idag

- Async –> Nästa kurs
  - Enligt kursplanen finns Async med I både denna och nästa kurs, och förra året valde vi att lägga detta helt I nästa kurs istället.
  - En kort demo av async finns, men jag föreslår att vi tar det I nästa kurs.
  - I uppgiften så kan ni därför bortse från de asynkrona delarna.

# Kardinalitet

- En-till-en
- En-till-många
- Många-till-många

# Exempel City.cs (Många till en)

```
public partial class City
{
    public int Id { get; set; }
    public string? Name { get; set; }
    public int? CountryId { get; set; }

    public virtual Country? Country { get; set; }
}
```

# Exempel Country.cs (En till många)

```
public partial class Country
{
    public Country()
    {
        Cities = new HashSet<City>();
    }

    public int Id { get; set; }
    public string? Name { get; set; }

    public virtual ICollection<City> Cities { get; set; }
}
```

# Många till många - EF

- En stad(namnet) kan finnas I många länder, och länder kan ha mångas städer
- En film kan ha många skådespelare, och en skådespelare kan vara med I många filmer
- Ett hus kan ha många ägare, och en person kan äga många hus.

# City.cs och Country.cs (Många till många)

```
public partial class City
{
        public int Id { get; set; }
        public string Name { get; set; }
        public virtual ICollection<Country> Countries { get; set; }
}
```

```
public partial class Country
{
        public int Id { get; set; }
        public string? Name { get; set; }
        public virtual ICollection<City> Cities { get; set; }
}
```

# Junction table

- För att hålla reda på vilka städer som finns I vilka länder, och vilka länder som har visa städer, så behövs en tabell som håller reda på City**s**Id och Country**s**Id.
- Med Entity framework och code-first så skapas den automatiskt, och hålls sedan reda på av EF. Tabellnamnet blir **CityCountry**

# Context-filen

```
 public partial class MyDBContext : DbContext
{
   protected override void OnConfiguring(DbContextOptionsBuilder optionsBuilder)
 {
optionsBuilder.UseSqlServer("Server=.\\SQLExpress;Database=MovieDb;Trusted_Connect
ion=True;TrustServerCertificate=True;");
 }


   public DbSet<City> Citys { get; set; }
   public DbSet<Country> Countrys { get; set; }
}
```

# Programmet

```
foreach (var country in myDb.Countrys.Include(a => a.Citys))
{
    Console.WriteLine($"\nLand: {country.Name}");

    foreach (var city in country.Citys)
    {
        Console.WriteLine($"  Stad: " + city.Name);
    }
}
```

# Code - Along

- EFDemo-Many2Many

- Bonus
  - "Fönster" I consollen

# Övning - Receptdatabasen

- Skapa en application som håller reda på recept
- Två tabeller:
  - Maträtt
  - Ingrediens

- En maträtt innehåller många ingredienser, och en ingrediens kan finnas I flera maträtter.
- Använd EF code-first
- Lägg in minst tre maträtter och tio ingredienser
- Visa två listor:
  - Alla maträtter med alla ingredienser
  - Ingredienser, och I vilka maträtter de används

# Idag…

- Övningar: Receptdatabasen (Frivillig)
- Gruppuppgiften: Webbshoppen
  - Fundera först på hur en webshop är uppbyggd
    - Titta på ert gamla arbete I kursen Agil utveckling
    - Fundera på vilka objekt och relationer som kan finnas
    - Anteckna era tankar, börja planera.
    - Lucidchart!
    - Fundera på om det behövs en många-till-många-tabell…
- Prova Fönster-lösningen
- Lite längre lunch, men handledning till 16.45.

# Länkar

- Entity Framework Core 5.0 — Many-to-Many Relationships | by Henrique Siebert Domareski | Medium