

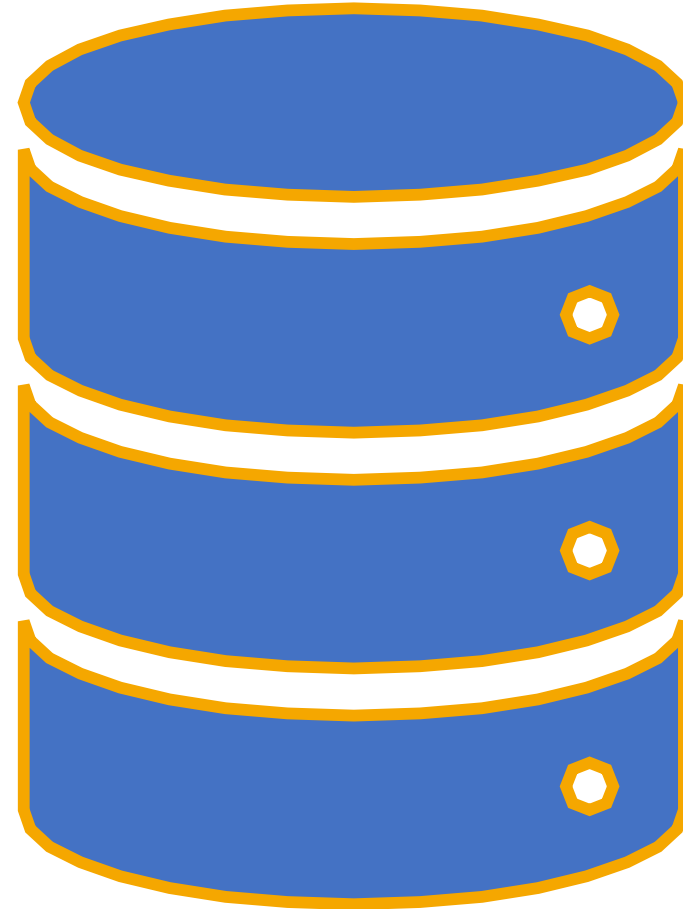
# Entity Framework

Jakub Mielczarek

Politechnika Wrocławska

Koło Naukowe Kredek

CPC 2021-2



# O mnie

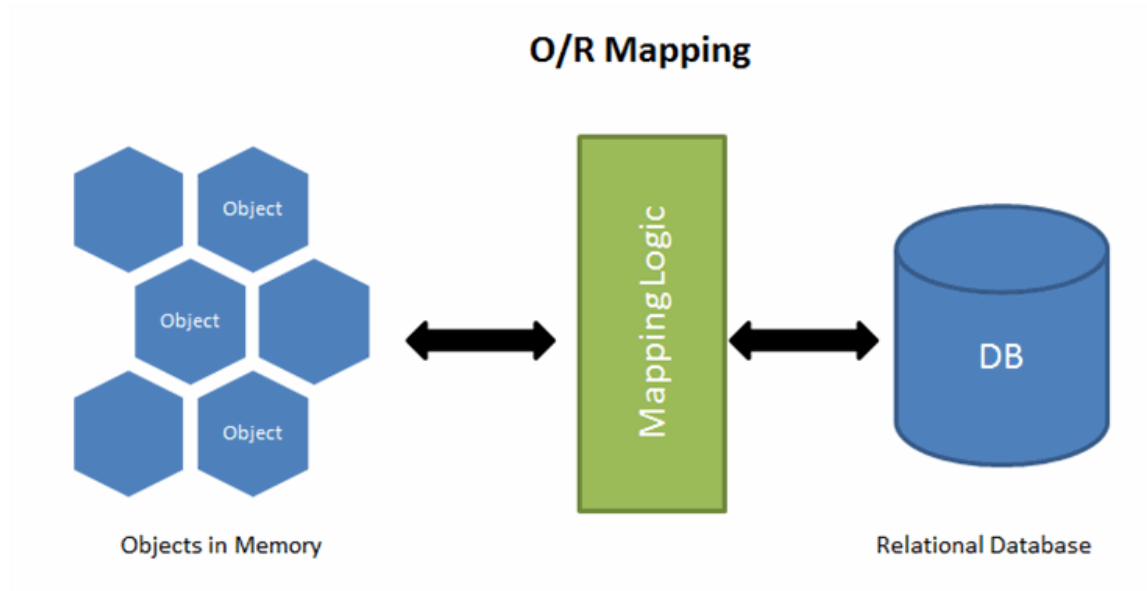
Jakub Mielczarek

- CPC 2020-2
- Członek KN Kredek od 2021
- .NET Developer (DTiQ)
- Technologie:
  - C#
  - .NET Core WebApi
  - MS SQL
  - Entity Framework/Dapper
- Studiuję informatykę na W4 (inż)



<https://www.linkedin.com/in/jakub-mielczarek-633087205>





# Entity Framework + ORM



# Mapowanie obiektowo-relacyjne

Sposób odwzorowania obiektowej architektury systemu informatycznego na bazę danych (lub inny element systemu) o relacyjnym charakterze.

```
public class Movie
{
    public int Id;
    public string Title;
    public int YearOfRelease;
    public int GenreId;
}
```



	Column Name	Data Type	Allow Nulls
🔑	Id	int	<input type="checkbox"/>
	Title	nvarchar(50)	<input type="checkbox"/>
	YearOfRelease	int	<input type="checkbox"/>
▶	GenreId	int	<input type="checkbox"/>

```
Movie movie = new Movie();
movie.Id = 1;
movie.Title = "Incepcja";
movie.YearOfRelease = 2010;
movie.GenreId = 5;
```



	Id	Title	YearOfRelease	GenreId
	1	Incepcja	2010	5





# Entity Framework

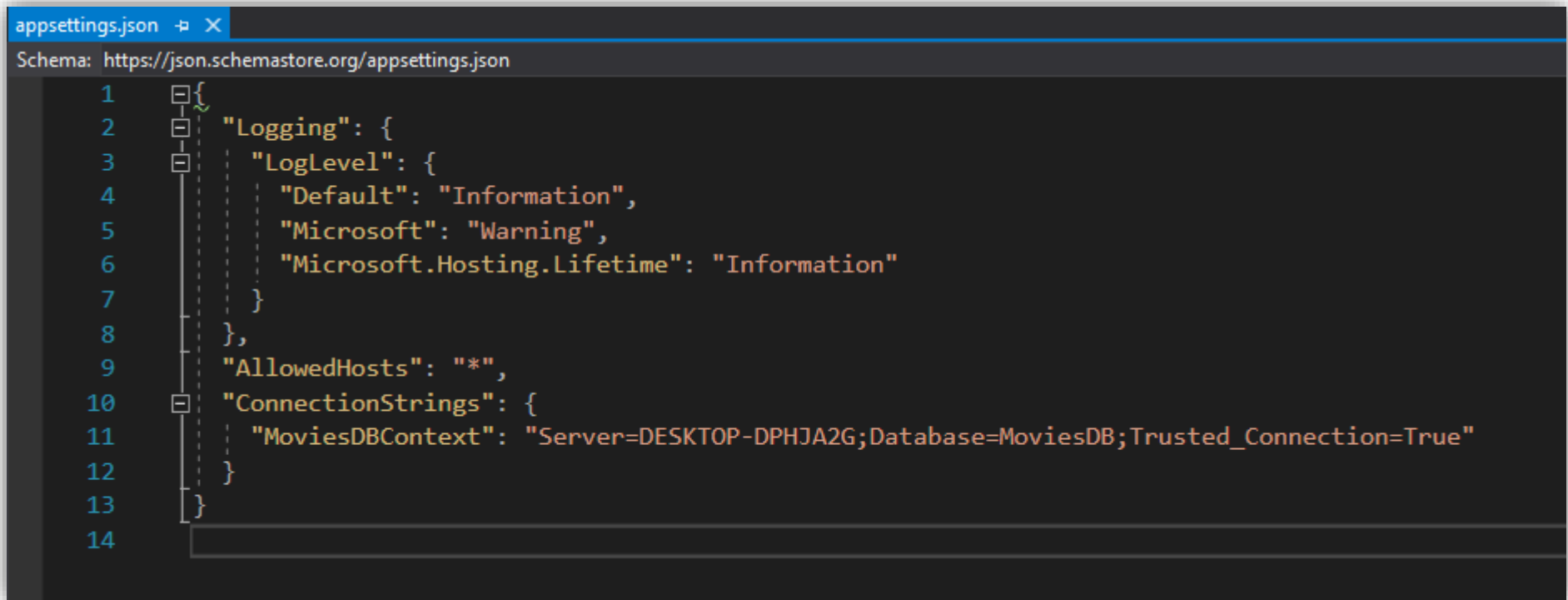
Narzędzie służące do mapowania obiektowo-relacyjnego dla platformy .NET, umożliwia 3 podejścia do pracy z bazą danych:

- Database first
- Model first
- Code first



# Code first - kolejne kroki

- ConnectionString w pliku appsettings.json



The screenshot shows a code editor window titled 'appsettings.json' with a schema link 'https://json.schemastore.org/appsettings.json'. The JSON content is as follows:

```
1 {
2   "Logging": {
3     "LogLevel": {
4       "Default": "Information",
5       "Microsoft": "Warning",
6       "Microsoft.Hosting.Lifetime": "Information"
7     }
8   },
9   "AllowedHosts": "*",
10  "ConnectionStrings": {
11    "MoviesDBContext": "Server=DESKTOP-DPHJA2G;Database=MoviesDB;Trusted_Connection=True"
12  }
13 }
14
```

# Code first - kolejne kroki

- Stworzenie modelu bazy danych:
  - klasy z adnotacjami (tabele z opisem kolumn/relacji)
  - klasa DbContext (zbiór tabel - baza danych)

```
public class Movie
{
    public int Id;
    public string Title;
    public int YearOfRelease;
    public int GenreId;
}
```



	Column Name	Data Type	Allow Nulls
🔑	Id	int	<input type="checkbox"/>
	Title	nvarchar(50)	<input type="checkbox"/>
	YearOfRelease	int	<input type="checkbox"/>
▶	GenreId	int	<input type="checkbox"/>



<https://www.entityframeworktutorial.net/code-first/dataannotation-in-code-first.aspx>

# Code first - kolejne kroki

- Zarejestrowanie klasy DbContext w metodzie ConfigureServices klasy Startup.cs

```
public void ConfigureServices(IServiceCollection services)
{
    services.AddControllersWithViews();

    services.AddDbContext<DataBaseContext>(options =>
        options.UseSqlServer(Configuration.GetConnectionString("MoviesDBContext")));
}
```





# Code first - kolejne kroki

- Migracje:
  - Add-migration
  - Update-database

```
PM> add-migration
cmdlet Add-Migration at command pipeline position 1
Supply values for the following parameters:
Name: Initial-migration
Build started...
Build succeeded.
To undo this action, use Remove-Migration.
PM> update-database
Build started...
Build succeeded.
Done.
PM>
```

121 %  
Error List Output Package Manager Console



<https://docs.microsoft.com/en-us/ef/core/managing-schemas/migrations/?tabs=dotnet-core-cli>

# LINQ

- Zestaw metod ułatwiający wykonywanie zapytań np. do baz danych, plików XML lub tablic.

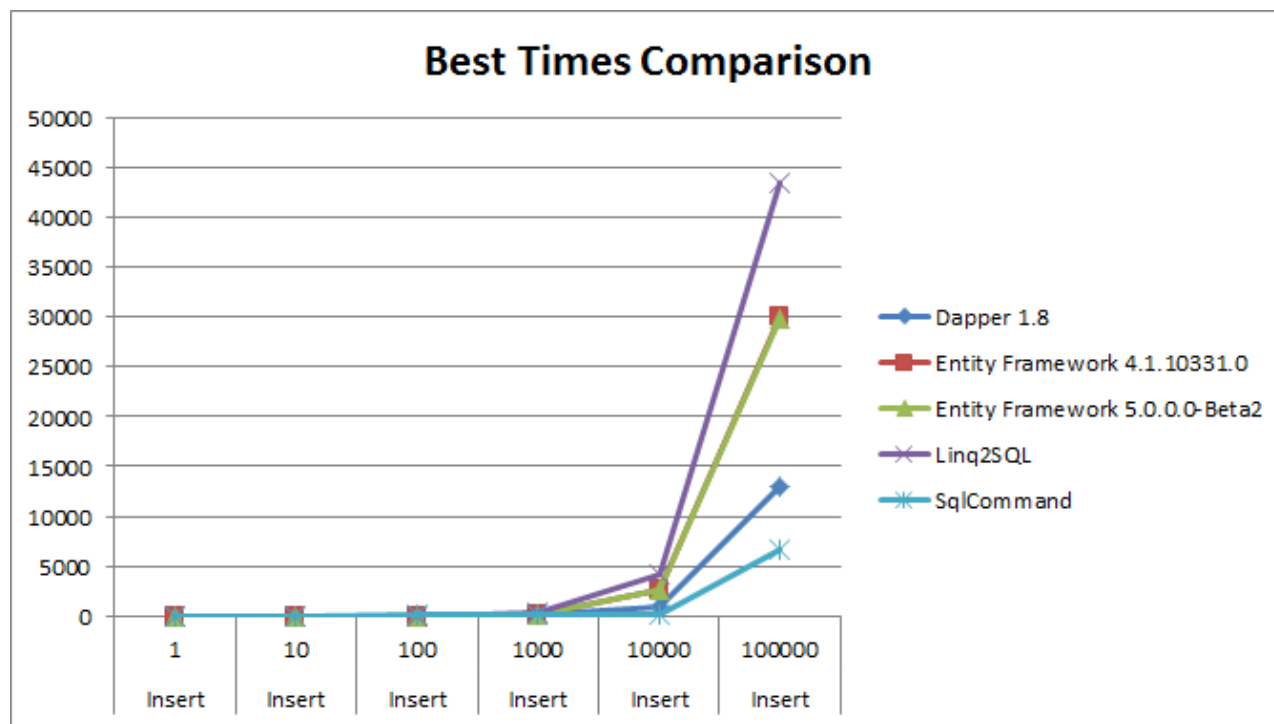
```
var movie = _context.Movies.FirstOrDefault(movie => movie.Id.Equals(3));
```

```
SELECT * FROM Movies WHERE Id = 3
```

# LINQ - wybrane metody

- First() - zwraca pierwszy element spełniający warunek (wyjątek jeśli nie istnieje)
- Single() - zwraca jedyny element spełniający warunek (może występować tylko jeden, inaczej wyrzuci wyjątek)
- FirstOrDefault() - tak jak First, ale zwraca null jeśli element nie występuje

# Wydajność



<http://www.tiernok.com/posts/evaluating-orms-for-batch-data.html>

