**Výběr ORM knihovny pro platformu .NET**

# **Úvod**

K původnímu frameworku EFC jsem si vybral poměrně dosti odlišnou knihovnu/framework s názvem [Dapper](https://dapper-tutorial.net/). Kvůli patřičným odlišnostem frameworků a také kvůli nedostatku času jsem složitost struktury dat, se kterými v řešení pracuji, mírně zjednodušil. Odstranil jsem vztahy mezi entitami typu „many-to-many“ a pracují pouze s ostatními vztahy. V projektu jsem pracoval s EFC verze 6.0.10 a s frameworkem Dapper verze 2.0.123.

# **Entity Framework Core (EFC)**

### Stručný popis

EFC je moderní komplexní framework od firmy Microsoft sloužící k usnadnění objektově-relačnímu mapování dat pro platformu .NET. Uživatele naprosto odstíní od jazyka SQL a uživateli postačí pouze základní znalosti práce s databází. Navíc umožňuje později měnit schéma databáze pomocí migrací a také podporuje práci s LINQ. Dále podporuje i líné vyhodnocování.

## Dokumentace

Dokumentace pro EFC je vskutku obsáhlá ale velice přehledná. Obsahuje spousty ukázkových příkladů, které uživateli pomůžou při seznamování se s frameworkem. Kdykoli jsem v projektu narazil na problém týkající se EFC, našel jsem jeho řešení právě v dokumentaci. Po zkušenostech s ostatními dokumentaci je tato dokumentace jedna z nejlepších, se kterými jsem se osobně setkal.

## Intuitivnost

Intuitivnost dá se říct jde ruku v ruce s dokumentací tohoto frameworku. Jakmile jsem narazil na problém, téměř vždy jsem jej vyřešil pomocí dokumentace. Za velké plus považuji způsob definování vztahů mezi daty. EFC poskytuje definování pomocí anotací, Fluent API a v poslední řadě je sám schopný (zřejmě pomocí vhodných názvů) vztahy odvodit. Uživatel si tedy může vybrat, který způsob mu vyhovuje nejvíce.

### Hodnocení

Jelikož jsem s EFC již pracoval, bylo pro mě jeho použití značně jednoduší. Celkově se jedná o knihovnu, která uživateli značným způsobem zjednoduší práci. Uživatel taktéž nemusí mít hluboké znalosti jazyka SQL a práci s databází tak poměrně lehce zvládne. To si ale vybírá daň na výkonu této knihovny. V porovnání s knihovnou Dapper je značně pomalejší. Což můžeme vidět jako výsledek projektu. Osobně bych knihovnu doporučil pro práci s daty, jejíchž struktura je složitější.

# **Dapper**

### Stručný popis

Dapper je daleko menší, jednodušší knihovnou, která také slouží k objektově-relačnímu mapování dat. Dokonce jsem našel pro tuto knihovnu označení „King of Micro ORM“. Na rozdíl od EFC upřednostňuje tato knihovna výkon před uživatelskou přívětivostí. Knihovna poskytuje pouze pár základních funkcí přistupujících k databázi. Téměř všechny ale vyžadují SQL dotaz, který specifikuje, co chceme s databází vykonávat. Tedy opět na rozdíl od EFC, zde musí uživatel mít alespoň mírně pokročilé znalosti jazyka SQL. Zároveň ale existují knihovny třetí strany (DapperPlus, DapperContrib), které poskytují základní CRUD metody, čímž opět odstiňuje uživatele od SQL dotazů. V řešení jsem však použil pouze základní knihovnu Dapper. Knihovna také podporuje transakce v jazyce SQL.

### Dokumentace

Dokumentace knihovny Dapper je opět menší oproti dokumentaci EFC. Opět je ale přehledná a „výřečná“. Neobsahuje tolik ukázkových příkladů jako EFC ale opět funkcionalitu vysvětluje podrobně a jasně. Během řešení projektu se občas stalo, že oficiální stránka s dokumentací byla nedostupná. Dokumentace také obsahuje i popis funkcionality knihoven třetích stran jako je DapperPlus, DapperContrib a DapperTransaction. Ze zvědavosti jsem procházel dokumentaci pro knihovnu DapperPlus, která mi přišla daleko méně přehlednějši, než dokumentace základní knihovny Dapper.

### Intuitivnost

Oproti EFC je práce s knihovnou Dapper složitější. V řešení jsem pro usnadnění práce vytvořil třídu reprezentující repozitář, ve které jsem pomocí jazyka SQL a funkcím knihovny Dapper definoval základní metody CRUD. Zdrojový kód, je díky SQL patřičně obsáhlejší ale zase uživateli umožňuje větší přehled nad mapováním a tím umožňuje uživateli dosáhnout lepšího výkonu. Například díky vlastní správě id atributů jednotlivých entit. Mapování směrem z relace do objektu je pak velice jednoduchá a intuitivní.

### Hodnocení

Oproti EFC je použití knihovny Dapper daleko pracnější i mírně složitější. Zároveň se očekává znalost jazyka SQL, která je klíčová. Na druhou stranu je Dapper daleko rychlejší co se týče výkonu. Není zde ale podpora líného vyhodnocování a používání LINQ.

## **Závěrečné doporučení**

Knihovnu Dapper bych osobně upřednostnil, pokud je pro naši firmu výkon klíčový. Pro mapování jednodušších dat je to skvělé, rychlé a po krátkém seznámení s knihovnou i jednoduché řešení. Jedinou nevýhodou je psaní vlastních SQL dotazů, které je zdlouhavé. Pro složitější strukturu dat, například se vztahy many-to-many, bych osobně preferoval knihovnu EFC, protože značně usnadňuje uživateli práci. Je zde jednoduše a intuitivně řešený popis vztahů mezi daty. Navíc zdrojový kód potřebný pro uskutečnění mapování je minimální. Po krátkém průzkumu pár open-source projektů jsem ale viděl, že programátoři často používají obě knihovny dohromady. Osobně mi EFC přišla přehlednější a intuitivnější.