Efficient data access & Effective object mapping with Dapper & .NET 5

SCOTT DOCKENDORF
CTO, REMOTE OPERATIONS

Goals for this session

- Reinforce ORMs, Micro ORM concepts
- Introduce Dapper as an effective and efficient data access & object mapping solution
- Articulate best fit scenarios
- Provide code examples and resources to empower you to explore use of Dapper in the future

Setting The Stage

ORM VS MICRO ORM

Data Access Options

ADO.NET

SQL Connection
SQL Command
Data Reader/Data Set/Data Tables
Manual Object Mapping

Micro ORM

Queries (POCO)
Commands
Object Mapping

Dapper
Mighty ORM
PetaPoco
FluentData
Massive

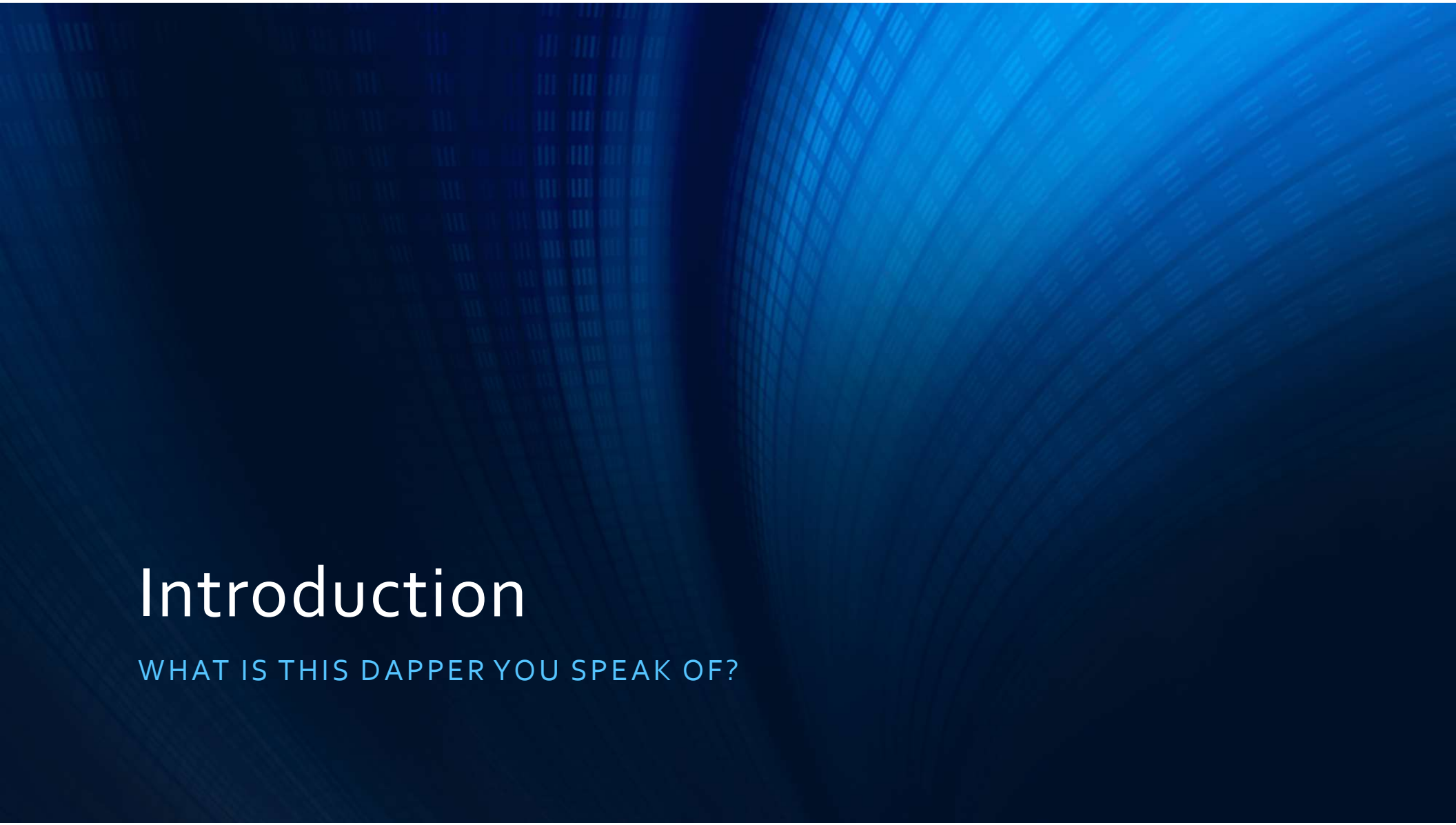
ORM

Code-First approach
Modeling (EDM via POCO)
Visual Designer
Migrations
Object Change Tracking
Querying by LINQ

Entity Framework
nHibernate
LlblGen

ORM vs Micro ORM

	Entity Framework (ORM)	Dapper (Micro ORM)
POCO	Yes	Yes
Queries	Yes (Linq or SQL)	Yes (SQL)
Commands	Yes (Linq or SQL)	Yes (SQL)
Second level cache	Yes	No
Relationships	Yes	No
Object Change Tracking	Yes	No
Designer	Yes	No
Migration	Yes	No


Introduction
WHAT IS THIS DAPPER YOU SPEAK OF?

What is Dapper

- Micro-ORM created in 2011 by Nick Craver and the StackOverflow team
- Works with all ADO.NET Providers including SQL Server, SQLite, Oracle, MySQL, PostgreSQL, Firebird.
- Targets .net Framework, .NET Standard 2.0 & .NET 5
- Licensing: Open Source with Apache-2.0

Dapper's purpose

*"Dapper's simplicity means that many feature that ORMs ship with are stripped out. It worries about the **95% scenario** and gives you the tools you need most of the time. **It doesn't attempt to solve every problem**." ***

- To Solve the following in a performant manner
  - Command execution
  - Query execution
  - Object/collection mapping

*** From Dapper homepage https://github.com/StackExchange/Dapper:*

How Dapper Works

- Extension methods off IDbConnection
  - `Query(), QueryFirstOrDefault(), QueryMultiple(), QueryAsync(), …`
  - `Execute(), ExecuteAsync(), …`
- Object Mapping based on Object & result set field names
  - Uses Dynamic Method Generation (MSIL) to assign column values to object properties
  - If different, then consider `Dapper.FluentMap` NuGet package
- Connection pooling
- Caches query info (materializes objects & parameters quickly)

Benefits

- Performance
- Reduced custom code required for:
  - Query/Command execution
  - Result set to collection mapping
- Leverage the benefits of SQL and provides more granular control over
  - What SQL is executed
  - What data is necessary for each query
- Flexibility
  - We have used with API, Windows Services, Utility applications, Websites

Read Performance

Sample Size: 10 Sports, 10 teams per sport and 10 players per team (1,000 records)

Avg. Exec. Time	Player by ID (ms)	Roster by Team ID (ms)	Team Rosters by Sport ID (ms)
EF Core w/ Tracking	2.00	1.29	6.45
EF Core w/ No Tracking	1.47	0.69	4.53
Dapper	**0.13**	**0.31**	**1.11**

Source: "Dapper vs EF Core Query Performance Benchmark" by Matthew Jones: https://bit.ly/30LvNET
Code: https://github.com/exceptionnotfound/DapperEFCoreQueryPerformance2019

Read/Write Performance

Sample Size: 1 table with 50 columns and 500,000 rows
Usecases: Select 5,000 rows , Insert a row, Update a row at index 5001

Avg. Exec. Time	Select 5000 rows (ms)	Insert row (ms)	Update Row 5001 (ms)
EF Core Linq	137.1 (AsNoTracking) 148.1	16.20	4.718
EF Core plain SQL	132.6 (AsNoTracking) 149.2	5.645	4.652
Dapper	**129.2**	**5.243**	**4.438**

Source: "Entity Framework Core 2 Vs Dapper Performance Benchmark" by Muhammad Mohsin: https://bit.ly/3n9WIK9
Code: https://github.com/mohsin91/EFCore2VsDapperPerformanceBenchmark

How to get Dapper?

- Prerequisites (for use with SQL Server)
  - For .NET Core/.NET 5 – Add Dependency to Microsoft.Data.SqlClient
  - For .NET Framework – Use System.Data.SqlClient
- Add Dapper NuGet package
- …. That's it!

- Let's dive in!

DEMO

QUERIES, COMMANDS, TRANSACTIONS AND MORE

SAMPLE DATABASE FROM SQLSERVERTUTORIAL.NET

Demo In Review

- Queries
  - List (with in-line SQL & stored procedures)
  - Get (with in-line SQL & stored procedures)
  - Get Parent/Child objects
  - Different Model mapping
- Commands
  - Execute
  - Transactions
  - Return values
  - Execute command multiple times
- Exception Handling
- Design Approach
  - Query/Command slices

Best Fit Scenarios

ORM VS MICRO ORM

No Civil War necessary

Both ORM & Micro-ORMs are functional data access mechanisms

Approach boils down to two factors

(a) functional & technical requirement alignment

(b) environmental fit (Self, Team, Department)

Environments/Teams/Projects are unique

SOME

- New project, new database
- Development team can manage all aspects of the project (architecture, database , front-end, back-end, API)
- Can build from scratch

OTHERS

- Supporting & enhancing existing codebase with performance issues
- Partner with DBA team
  - … that prefers abstraction layer over development speed (for performance tuning)
- Existing database
  - … that contains business logic in stored procedures or views
  - … that requires complex joins with tables that have millions of rows (that may present performance challenges with ORM solution)

Overall Guidance

ORM

- Brand new project
- Development team is responsible for all layers
- Team is well versed in C# over SQL and can fully support (& tune performance in) C#

MICRO ORM

- New or existing database
- New or existing project
- Complex database with business logic
- Partnering with DBA team who prefers abstraction
- Want more flexibility with hands-on SQL development

"When considering approach, **does your team**:" (from Tim Corey)

- … understand how to use {o} well?
- … know how to diagnose issues with {o} code?
- … know how to diagnose under-performing {o} queries and fix them?
- … understand the code that {o} wrote for you well enough to know what it is doing?
- … know how to protect the {o} credentials on client machines?

Where {o} == Dapper, Entity Framework, etc…

Src: https://www.iamtimcorey.com/blog/137806/entity-framework

Re: Dapper -
"When considering approach, does your team:"

- … understand how to use {0} well?
  - Easy to learn, low barriers to use, simple code solution
- ~~… know how to diagnose issues with {0} code?~~
  - Normal debugging techniques apply, no generated code
- … know how to diagnose under-performing {0} queries and fix them?
  - Dapper uses SQL that you (or your team) wrote, so someone should be able to help
- ~~… understand the code that {0} wrote for you well enough to know what it is doing?~~
  - Dapper does not contain any generated code
- … know how to protect the {0} credentials on client machines?
  - No different than Raw or ORM/EF.
  - Encrypted, NOT in source code but in UserSecrets OR Server Variables or Cloud Configuration

Src: https://www.iamtimcorey.com/blog/137806/entity-framework

Advanced Topics

EXTENSIONS, FAULT TOLERANCE &
INFRASTRUCTURE ARCHITECTURE

Extending Dapper with NuGet Packages

	Entity Framework (ORM)	Dapper (Micro ORM)	Dapper Extensions
POCO	Yes	Yes	Yes (with Dapper.FluentMap for different named properties)
Queries	Yes (Linq or SQL)	Yes (SQL)	n/a
Commands	Yes (Linq or SQL)	Yes (SQL)	n/a
Second level cache	Yes	No	n/a
Relationships	Yes	No	Yes (with Dapper.Mapper)
Object Change Tracking	Yes	No	Yes (with Dapper.Contrib)
Designer	Yes	No	
Migration	Yes	No	

Dapper Extensions

- Dapper (54.3M NuGet downloads)
- Dapper.Contrib (4.19M) – CRUD Helpers, entity lists & IsDirty entity tracking
- Dapper.SQLBuilder (1.23M) – Dynamic building of SQL commands
- Dapper.FluentMap (1.08M) – Helper for object mapping (different property names, append text to object properties, etc..)
- DapperExtensions (1.06M) – CRUD & predicate system
- Dapper.Mapper (254k) – IDs and automatic object/collection assignments objects based on relationships
- …

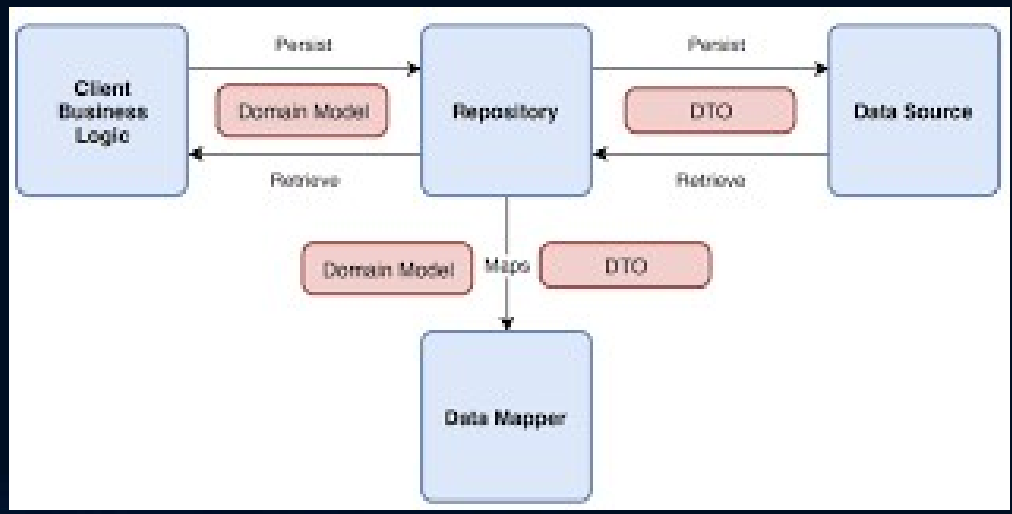
Advanced Topic: Transient Errors

- Especially important in PaaS-based applications & DaaS environments
  - Temporary service interruptions, many are self-healing
  - Database connections drop as resources are dynamically allocated
  - Jumping through multiple load balancers, multi-tenant services throttled due to activity
- How to resolve? Use smart retry/back-off logic
- Enter Polly.net - https://github.com/App-vNext/Polly
  - Resilience and transient-fault-handling library for Retry, Circuit Breaker, Timeout, Bulkhead Isolation, and Fallback policies in a fluent and thread-safe manner.
  - Open Source
  - Targets
    - .NET Standard 1.1 (coverage: .NET Core 1.0, Mono, Xamarin, UWP, WP8.1+)
    - .NET Standard 2.0+ (coverage: .NET Core 2.0+, .NET Core 3.x, and later Mono, Xamarin and UWP targets).
    - The nuget package also includes direct targets for .NET Framework 4.6.1 and 4.7.2.

For more information, please visit: https://bit.ly/32xlXhg

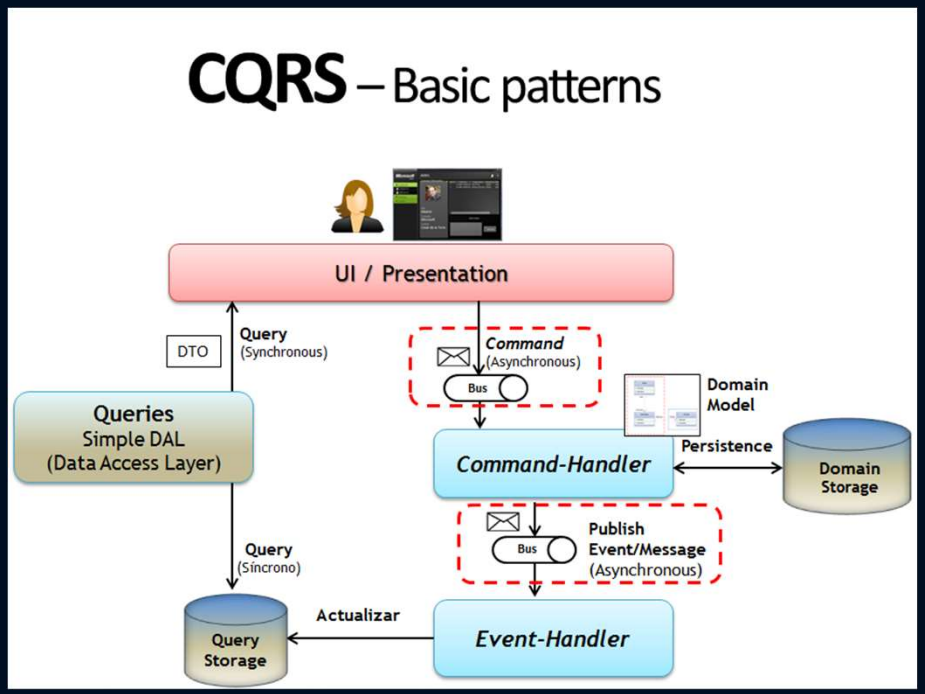
Advanced Topic – Repository Pattern

- Alex Calingasan (@alexcodetuts)
  - Article: https://bit.ly/2FAhku6
  - Video: https://bit.ly/2FpfjBg
  - Source: https://bit.ly/3mhkBiC
- Mukesh Murugan
  - Article: https://www.codewithmukesh.com/blog/dapper-in-aspnet-core/
- Mukesh Kumar
  - https://www.c-sharpcorner.com/article/dapper-and-repository-pattern-in-web-api/
- Dapper Tutorial
  - https://dapper-tutorial.net/knowledge-base/52418496/generic-repository-pattern-for--net-core-with-dapper



Advanced Topic – Mediator with CQRS

- Contoso how Jimmy Bogard would write it
  - Command & Query Responsibility Segregation (CQRS) & MediatR
  - AutoMapper
  - Vertical slices
  - Razor Pages
  - Fluent Validation
  - HTML Tags
  - EF Core
- For more information:
  - https://github.com/jbogard/ContosoUniversityDotNetCore-Pages



Goals for this session

- ✓Introduce Dapper as an effective and efficient data access & object mapping solution
- ✓Articulate best fit scenarios
- ✓Provide code examples and resources to empower you to explore use of Dapper in the future

Want to dive in more?

https://github.com/scottdock/DapperTalk

DOWNLOAD SAMPLE CODE & SLIDES

References

- Project Homepage: https://github.com/StackExchange/Dapper
- Nuget package: https://www.nuget.org/packages/Dapper
- Dapper Training:
  - https://dapper-tutorial.net/
  - https://www.pluralsight.com/courses/getting-started-dapper (by Steve Michelotti)
- Fellow Dapper content creators
  - Tim Corey - https://www.youtube.com/watch?v=eKkh5XmoOlU
  - Kevin Griffin @ Jetbrains - https://www.youtube.com/watch?v=glhRrF49M-Q

Good Articles

- Should I use Entity Framework (Tim Corey) https://www.iamtimcorey.com/blog/137806/entity-framework
- Dapper vs EF Core Query Performance Benchmarking (Matthew Jones) https://exceptionnotfound.net/dapper-vs-entity-framework-core-query-performance-benchmarking-2019/

Q&A


Thank you!