

.NET Core



Efficient data access & Effective object mapping with **Dapper & .NET Core**

SCOTT DOCKENDORF

CTO, REMOTE OPERATIONS

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

Setting The Stage

DATA ACCESS & ORM APPROACH

Data Access Options

.net/.NET Core

- SQL Connection/Command
- DataReader
- DataSet/DataTables
- Manual object mapping

Micro ORM

- Queries
- Commands
- Object Mapping

Dapper
nPoco

Object Relational Mapping

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

Entity Framework
nHibernate
LlblGen

No Civil War necessary

Both ORM & Micro-ORMs are valid data access mechanisms

Approach boils down to (a) functional & technical requirement match & (b) environmental fit

- Pure ORM may be a good fit:
 - 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 (like Dapper) may be a good fit:
 - New or existing database
 - Complex database with business logic
 - Partnering with DBA team who prefers abstraction
 - Want more flexibility with hands-on SQL development



Environment is important

UNICORN

- New project, new database
- Our team can manage all aspects of the project
- Can build from scratch

REALITY

- 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)

“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>

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
- 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>:*

Benefits

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

How Dapper Works

- Extension methods off IDbConnection
 - `Query()`, `QueryFirstOrDefault()`, `QueryMultiple()`, `QueryAsync()`, ...
 - `Execute()`, `ExecuteAsync()`, ...
- Object Mapping based on Object & result set field names
 - If different, then consider `Dapper.FluentMap` NuGet package
- Connection pooling
- Caches query info (materializes objects & parameters quickly)

Re: Dapper - “When considering approach, does your team:”

- ... understand how to use {o} well?
 - You will see in a second, but the code is simple and straight forward
- ~~... know how to diagnose issues with {e} code?~~
 - Dapper does not contain any generated code
- ... know how to diagnose under-performing {o} queries and fix them?
 - Dapper uses SQL that you (or your team) wrote, so someone should be able to help
- ~~... understand the code that {e} wrote for you well enough to know what it is doing?~~
 - Dapper does not contain any generated code
- ... know how to protect the {o} 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>

How to get Dapper?

- Prerequisites
 - For .NET Core – 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

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 Topics

- Transient Errors
- Design Architecture
 - Repository Pattern
 - CQRS with MediatR & AutoMapper



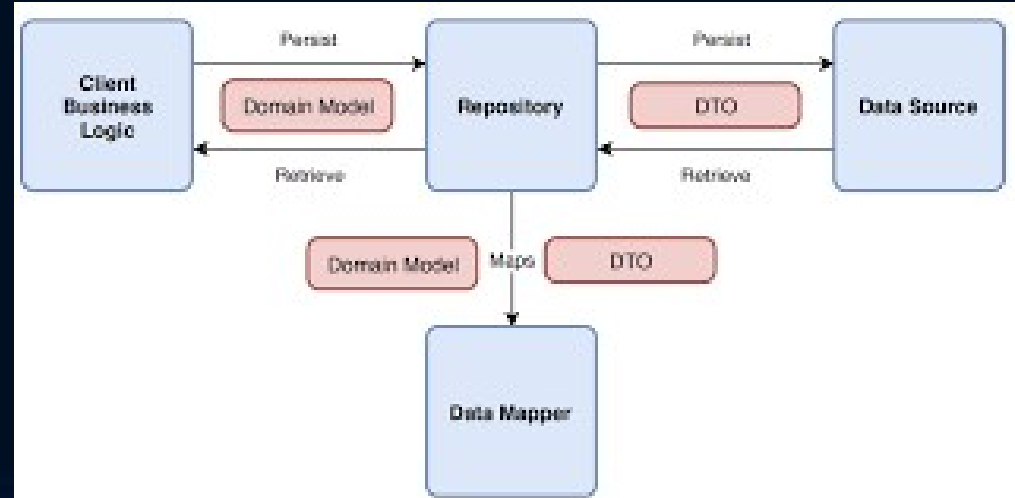
Advanced Topic: Transient Errors

- 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 3.0, 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/32xIXhg>

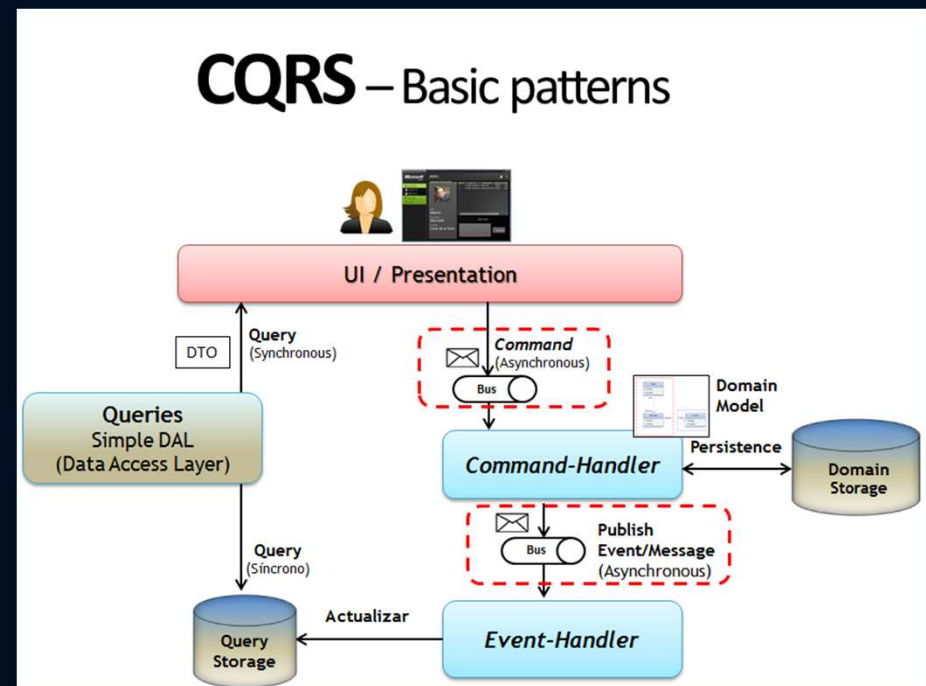
Advanced Topic – Repository Pattern

- 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

The background is a deep blue gradient. On the left, there's a faint grid of small squares. On the right, there are several concentric, curved lines that create a sense of depth and movement, resembling a tunnel or a stylized eye.

Want to dive in more?



[https://github.com/
scottdock/DapperTalk](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/>

The background is a deep blue gradient. On the left, there are faint, vertical, wavy lines. On the right, there are more pronounced, curved, wavy lines that create a sense of depth and movement. A subtle grid pattern is visible in the center, where the wavy lines intersect.

Q&A

The background is a dark blue gradient. On the left, there is a faint, light blue grid pattern. On the right, there is a curved, light blue effect that resembles a tunnel or a stylized 'C' shape, creating a sense of depth and movement.

Thank you!