Eventbrite

07.23.18

Understanding and Applying CQRS

Vinicius Feitosa Pacheco
@ViniciusPach
Agenda

1. What is CQRS?
2. When can we use CQRS?
3. Understanding CQRS
4. Applying CQRS
5. Common Mistakes
What is CQRS?

I have no doubt of that.
Command
Query
Responsibility
Segregation
CQRS is simply the creation of two objects where there was previously only one. The separation occurs based upon whether the methods are a command or a query...

Young, Greg (2010)
When can we use CQRS?
The “Normal” Architecture
First possible solution: Master/Slave
UI / API

Domain

Repository

Model

Eventual Consistency
Second possible solution: Cache
Data Synchronization
When CQRS is util?

- When my storage is a bottleneck.
- When the application has complex queries and these queries could be optimized.
- When a big number of users are updating a small data set and the data could be outdated.
Understanding CQRS
CQRS

- Query Stack – Operations that retrieve information from the data in the application.
- Command Stack – Operations that modify the State of the data in the application.
3.1

QueryStack
QueryStack

- It is simpler than the CommandStack.
- It is a synchronous layer that retrieves data from a denormalized reading.
- Presumes “flat” queries.
3.2

CommandStack
CommandStack

- Potentially asynchronous.
- Has *behavior-centric* approach where all business intention is initially triggered by the client as a use case.
- The Commands are declared in an imperative fashion and are raised asynchronously.
- The handlers returns success or failure.
- A command updates the state of an entity and raises an event that will update the data needed in the database reading.
3.3 Synchronization
Synchronization: Automatic updating

GET

QueryStack

CommandStack

Synchronous
(REST/RPC...)

POST/PUT...
Synchronization: Update possible

GET

QueryStack

CommandStack

Asynchronous (Bus/Queues...)

Event

POST/PUT...
Synchronization: Controlled update

GET

POST/PUT...

QueryStack

CommandStack

Asynchronous (Cron Process...)
Synchronization: Update on demand

GET

QueryStack → B

POST/PUT...

CommandStack → A

Asynchronous (Check-in Process)
3.4 Queueing
Applying CQRS
Common Mistakes
Mistake 1:

CQRS and Event Sourcing must be implemented together
Mistake 2:

CQRS requires eventual consistency
Mistake 3:

CQRS depends on Queues and Message Brokers
Mistake 4:

CQRS is easy
Mistake 5:

CQRS is architecture
CQRS is the best thing since sliced bread!

Pacheco, Vinicius (Today)
Microservice Patterns and Best Practices
Vinicius Feitosa Pacheco
January 2018


@ViniciusPach
May the CQRS be with you

Kenobi, Vinicius (2018)
Thank you