Hands-On Online Training by three dots labs

GO EVENT DRIVEN

Learn building **Extremely Scalable & Resilient** Go backends

Event-Driven Architecture is a modern and well-respected approach to building backend systems. However, developers who know the patterns well are rare. Many businesses get stuck with software that's slow and difficult to maintain.

We designed this training as a solid foundation for Event-Driven patterns. Your teams will learn the principles and gain hands-on experience building scalable and resilient Go projects.

Benefits for Your Organization:

- **Minimize production issues and downtime.**
- **Build high-performance systems that scale as the product grows.**
- **Improve productivity within teams with clear boundaries and contracts.**

Thanks to our unique hands-on format:

- **Trainees learn the patterns in practice by coding and can apply the same ideas in work projects.**
- **Available instantly and with no time limit. The training can be finished anytime — during or after work hours.**
- **We provide the playground and verify the solutions. No risk in experimenting on your product.**

OVER 600 DEVELOPERS HAVE JOINED THE FIRST EDITION OF GO EVENT-DRIVEN.

SEE MORE:

https://threedots.tech/event-driven/

FAQ:

- We issue VAT invoices — please provide company details and Tax ID during checkout.
- It's possible to pay by wire transfer — please contact us with company details.

Any questions?
Let us know:
contact@threedotslabs.com

Miłosz Smółka & Robert Laszczak

We spent the last 15 years working together on many different projects, including those in the financial, healthcare, and security domains. We also led multiple teams in startups using Go as the main backend language.

We are the authors of the threedots.tech blog and the Go With The Domain: Building Modern Business Software in Go ebook, which are some of the most popular resources on advanced programming patterns in Go. Our blog gets 300,000 unique visits annually, while 15,000 developers subscribe to our newsletter.

We are authors of Watermill, the most popular Go library for building event-driven applications.

In 2022, we created a unique training platform that lets developers learn by writing code.

The sale is open only for 2 weeks: 12.04.2024 — 26.04.2024.

(We run the sale only twice a year.)

$350 net/person

You will get **lifetime** access to the training platform.
We offer **Purchasing Power Parity discounts**

Ordering for a team?

10-19 licenses — **10% OFF**
20-29 licenses — **20% OFF**
30+ licenses — **30% OFF**

Looking for a training on Go basics?

We built a traning to get developers productive quickly:

GO IN ONE evening https://threedots.tech/go-in-one-evening/

20+ CODING HOURS*

20 MODULES

94 EXERCISES

BASIC ASYNC

WHAT YOU WILL LEARN	YOU WILL IMPLEMENT
✓ Learn about synchronous and asynchronous communication patterns to build scalable and resilient systems	• Refactor synchronous API calls to asynchronous processing
✓ Understand the benefits and challenges of asynchronous processing to handle high loads and improve responsiveness	• Handle temporary errors with retries in background workers
✓ Implement simple asynchronous processing using goroutines and retries to avoid blocking and handle temporary failures	

MESSAGE BROKER

WHAT YOU WILL LEARN	YOU WILL IMPLEMENT
✓ Understand the role of message brokers in event-driven systems	• Replace in-memory message passing with a production-grade message broker
✓ Learn how to publish and subscribe to messages using a message broker	• Implement message publishing and subscribing using the message broker
✓ Handle errors and retries when processing messages to ensure reliable message delivery	• Refactor message handlers to use the message broker's features
✓ Use consumer groups for scalability and fault tolerance to process messages in parallel and handle failures	

EVENTS

WHAT YOU WILL LEARN	YOU WILL IMPLEMENT
✓ Understand the concept of events and how they differ from messages to build event-driven systems	• Refactor the project to use events instead of plain messages
✓ Learn how to design and structure events for maintainability and evolution	• Design and implement event payloads for different use cases
✓ Implement event marshalling and unmarshalling to serialize and deserialize events	• Add event headers for metadata
✓ Use event headers for metadata to provide context and observability	

ROUTER

WHAT YOU WILL LEARN	YOU WILL IMPLEMENT
✓ Learn how to use Watermill to simplify message routing and handling	• Refactor the project to use Watermill
✓ How to simplify message handler implementation and error handling to focus on business logic	• Implement graceful shutdown for the message router
✓ Implement graceful shutdown and health checks for the message router to ensure service reliability	• Add health check endpoints for monitoring the service

MIDDLEWARES

WHAT YOU WILL LEARN	YOU WILL IMPLEMENT
✓ Learn how to use middleware functions to add cross-cutting concerns and reuse functionality	• Add logging middleware to log incoming messages and errors
✓ Implement logging and correlation ID propagation using middleware to improve observability and traceability	• Implement correlation ID middleware for request tracing
✓ Understand how to handle dependencies and configuration in middleware to keep handlers focused on business logic	• Refactor middleware to handle dependencies and configuration

ERRORS

WHAT YOU WILL LEARN	YOU WILL IMPLEMENT
✓ Learn how to handle different types of errors in event-driven systems to ensure system resilience	• Add error logging middleware to log errors and message details
✓ Implement error logging and monitoring using middleware to detect and diagnose issues	• Implement retry middleware for handling temporary errors
✓ Handle temporary errors, malformed messages, and code bugs to prevent system failures and data loss	• Handle malformed messages and code bugs gracefully

COMPONENT TESTING

WHAT YOU WILL LEARN	YOU WILL IMPLEMENT
✓ Understand the importance of component testing in event-driven systems to ensure system reliability	• Refactor the project to allow running the service in a test environment
✓ Learn how to write mocks for external dependencies to isolate components and improve testability	• Write mocks for external dependencies used in the project
✓ Run the service in a test environment and verify its behavior to catch integration issues early	• Implement component tests for different use cases

EVENTS WITH CQRS PATTERN

WHAT YOU WILL LEARN	YOU WILL IMPLEMENT
✓ Learn about the CQRS pattern and how it relates to event-driven systems to separate read and write concerns	• Refactor the project to use the CQRS event bus from Watermill
✓ Understand how to use a high-level event bus from Watermill to simplify event publishing and handling	• Implement event handlers and processors using the CQRS from Watermill
✓ Implement event handlers and processors using the CQRS component	• Use consumer groups for scaling event processing

AT LEAST ONCE DELIVERY

WHAT YOU WILL LEARN	YOU WILL IMPLEMENT
✓ Understand the concept of at-least-once delivery in event-driven systems to ensure message processing	• Store event data in a database for querying and consistency
✓ Learn how to handle message redelivery and idempotency to prevent duplicate processing	• Implement idempotent event handlers and repositories
✓ Implement idempotent event handlers and repositories to ensure data consistency	• Handle message redelivery and ensure data consistency

OUTBOX

WHAT YOU WILL LEARN	YOU WILL IMPLEMENT
✓ Learn about the outbox pattern for ensuring data consistency in distributed systems	• Implement the outbox pattern for publishing events within transactions
✓ Understand how to publish events within database transactions to maintain data integrity	• Forward events from the outbox to the message broker
✓ Implement event forwarding from the outbox to the message broker to decouple services	• Refactor the project to use the outbox pattern for consistency

* Based on average time spent by our trainees

20+ CODING HOURS*

20 MODULES

94 EXERCISES

COMMANDS IN CQRS PATTERN

WHAT YOU WILL LEARN

- ✓ Understand the difference between commands and events to model user intent and system state
- ✓ Learn how to use the CQRS command bus for asynchronous processing to improve system responsiveness
- ✓ Implement command handlers and processors

YOU WILL IMPLEMENT

- Refactor the project to use commands for asynchronous processing
- Implement command handlers and processors
- Use separate topics for commands and events

READ MODELS

WHAT YOU WILL LEARN

- ✓ Learn about read models and their role in event-driven systems to optimize data for querying
- ✓ Understand how to build and update read models from events to maintain data consistency
- ✓ Implement read models for different querying scenarios to improve system performance

YOU WILL IMPLEMENT

- Design and implement read models for various querying needs
- Update read models based on incoming events
- Expose read models through API endpoints

MESSAGE ORDERING

WHAT YOU WILL LEARN

- ✓ Understand the challenges of message ordering in event-driven systems to ensure data consistency
- ✓ Learn different strategies for handling message ordering to prevent race conditions
- ✓ Implement message ordering using partitioning and entity versioning to maintain data integrity

YOU WILL IMPLEMENT

- Refactor the project to handle message ordering using different strategies
- Implement message ordering using partitioning and entity versioning
- Ensure data consistency and correctness in the presence of out-of-order messages

DATA LAKE

WHAT YOU WILL LEARN

- ✓ Learn about data lakes and their role in event-driven systems to store and analyze events
- ✓ Understand how to store events in a data lake for future processing and insights
- ✓ Implement event storage and retrieval from a data lake to enable data-driven decision making

YOU WILL IMPLEMENT

- Store all events in a central data lake for future processing
- Implement event forwarding from the message broker to the data lake
- Retrieve events from the data lake for building read models or analytics

INTERNAL AND VERSIONED

WHAT YOU WILL LEARN

- ✓ Understand the concept of internal and versioned events to manage event schema evolution
- ✓ Learn how to evolve events over time without breaking compatibility to support system evolution
- ✓ Implement event versioning and internal event handling to decouple services

YOU WILL IMPLEMENT

- Add versioning to events to allow for schema evolution
- Implement internal events for use within a single service or team
- Handle event versioning and compatibility in the project

MIGRATING READ MODELS

WHAT YOU WILL LEARN

- ✓ Learn how to migrate read models when event schemas change to maintain data consistency
- ✓ Understand the process of rebuilding read models from a data lake to recover from data loss or corruption
- ✓ Implement read model migration and rebuilding to ensure system resilience

YOU WILL IMPLEMENT

- Migrate read models when event schemas change
- Rebuild read models from events stored in the data lake
- Ensure data consistency and correctness during read model migration

METRICS AND ALERTING

WHAT YOU WILL LEARN

- ✓ Understand the importance of observability in event-driven systems to ensure system health and performance
- ✓ Learn how to instrument the system with metrics and alerts to detect and diagnose issues
- ✓ Implement metrics and alerts for monitoring and troubleshooting to improve system reliability

YOU WILL IMPLEMENT

- Instrument the project with metrics for monitoring and troubleshooting
- Set up alerts for detecting and responding to issues
- Use metrics and alerts to ensure the system's health and performance

TRACING

WHAT YOU WILL LEARN

- ✓ Learn about distributed tracing and its role in event-driven systems to understand system behavior
- ✓ Understand how to instrument the system with tracing to identify performance bottlenecks and errors
- ✓ Implement tracing for end-to-end visibility and troubleshooting to improve system observability

YOU WILL IMPLEMENT

- Instrument the project with distributed tracing
- Propagate trace context across service boundaries
- Use tracing for end-to-end visibility and troubleshooting

FAULT TOLERANCE

WHAT YOU WILL LEARN

- ✓ Understand the importance of fault tolerance in event-driven systems to ensure system availability
- ✓ Learn different strategies for handling failures and ensuring system resilience to prevent downtime
- ✓ Implement fault tolerance using retries, circuit breakers, and poison queues to handle errors gracefully

YOU WILL IMPLEMENT

- Implement retry mechanisms for handling transient failures
- Use circuit breakers to prevent cascading failures
- Set up a poison queue for handling and monitoring failed messages

SAGAS AND PROCESS MANAGERS

WHAT YOU WILL LEARN

- ✓ Learn about sagas and process managers for coordinating long-running processes to ensure data consistency
- ✓ Understand the differences between orchestration and choreography to choose the right approach
- ✓ Implement sagas and process managers for complex business workflows to handle distributed transactions

YOU WILL IMPLEMENT

- Design and implement sagas and process managers for complex workflows
- Handle compensating actions and rollbacks in case of failures
- Ensure data consistency and correctness across multiple services

* Based on average time spent by our trainees