Go - Advanced

Duration: 3 Days

Description

Go is the most *powerful, performant, and scalable* programming language today for creating web applications, microservices, and other distributed services. Go is the language of the Internet age, and the latest version of Go comes with major architectural changes. It is an open source programming language that makes it easy to build simple, reliable, and efficient software.

In this advanced golang course, you'll learn:

- Scalability in Go
  - Performance with Concurrency, profiling
  - Codebase - in terms of architecture and refactoring and keeping your code testable
- Writing a gRPC based application
- Understand when and where to use concurrency to keep data consistent and applications non-blocking, responsive, and reliable
- Working with databases
- Writing and running unit tests

Pre-requisites

- Must have coded sufficiently in Go before.
- Understands structs, pointers, arrays, function calls, language structure, importing & packaging libraries.

Pre-Read Recommendations

For them to read up prior to the event, they can go through Tour of Go until the `"Methods & Interfaces"` topic. Additionally, the participants are encouraged to refer the notes in each topic in the Go Training Examples directory until `08-02-errors`.

Course Outline

Day 1: Concurrency

- Concurrency: Goroutines, Parallelism
  - Concurrency with goroutines
  - Concurrency and Parallelism
- Concurrency: Sync, WaitGroup, Mutexes
  - Sync, WaitGroup
  - Mutexes
  - Deadlocks
  - RW Mutexes
- Concurrency: Handling Race Conditions
  - Example of Race Condition

- Concurrency: Channels
    - Channels
    - Channel Direction
    - Closing Channels
    - Range Over Channels
    - Channels - Select
    - Timeouts
- Concurrency: `context` package
- Concurrency: Patterns

Day 2: Testing and Benchmarking, ReSTFul & gRPC applications

- Unit Testing & Dependency Management
    - Writing and Running Unit Tests
    - Working with `go mod`
    - Writing assertions using `stretchr/testify`
- Benchmarking
    - What are benchmarks?
    - Writing and Running Benchmarks
    - Additional libraries for testing
- Writing a ReSTful API
    - Introducing the `gorilla` toolkit
    - Project Layout
    - Refactoring to a Clean-code architecture (DDD)
    - Writing Testable code
- Advanced Testing
    - Mocking
    - HTTP Testing
    - Concurrency Testing
- Intro to gRPC
    - Defining a Protobuf file
    - `unary` vs `streaming` API

Day 3: Databases, Profiling, Debugging, Docker & Kubernetes

- Databases
    - `init` and then main
    - `database/sql` package
    - Working with sqlite
    - Working with mongodb
- Profiling
    - Profiling - with an example

        - CPU
        - Memory
    - Tracing and comparing traces
- Debugging an application
    - With `VSCode` & `delve`
    - Logging
        - Multi-level logging
- Configuring a Go application
    - Environment Variables
    - CLI `flags`
- Docker and Containerization
    - Docker Introduction
    - Writing Docker files
    - Building Docker files
    - Using registries
    - Including go http apps into Container
- Kubernetes and Microservices
    - Kubernetes Model
    - Writing kubernetes config files
    - Deploying apps with docker and kubernetes