GO-PROG: Go Programming

Course Overview

This comprehensive Go Programming (Golang) course outline transitions a developer from foundational syntax to building highly scalable, concurrent backend applications. It aligns with the standard competencies required for modern cloud engineering and microservices development

Course Description

This Go Programming (Golang) course delivers a practical, hands-on deep dive into Google's open-source programming language, engineered for high-performance backend systems. Participants will transition from fundamental syntax to mastering Go’s unique, industry-leading concurrency model. Focus: Building fast, efficient, and highly scalable cloud-native applications.Core Philosophy: Prioritizing simplicity, explicit code structures, and memory safety over complex abstractions.Methodology: Hands-on lab exercises culminating in a production-ready web service or CLI tool.Objectives Write Idiomatic Go: Master standard formatting, explicit error handling, and clean code styling.Manage Memory Safely: Use pointers, slices, and structures efficiently without manual allocation issues.Scale with Concurrency: Design safe, multithreaded systems using goroutines, channels, and mutexes.

Learning Curriculum

Module 1

Introduction & Environment Setup

Go Philosophy: Understanding compilation speed, static typing, and the runtime.Local Setup: Installing the Go toolchain on Windows, macOS, or Linux.IDE Integration: Configuring VS Code or GoLand with official extensions.Go Workspace: Navigating go env, paths, and compiling binaries via go build.First Program: Dissecting package main, import "fmt", and the main() entry function

Module 2

Language Fundamentals

Variables & Constants: Syntax forms, type inference, explicit type declarations, and iota.Basic Types: Working with integers, floats, booleans, strings, and byte/rune types.Zero Values: Explicitly managing default uninitialized states in Go.Control Flow: Writing conditions (if/else, switch) and Go's singular looping construct (for).Functions: Declaring inputs, returning multiple values, and writing variadic functions.Deferred Code: Utilizing defer statement blocks for safe resource cleanup

Module 3

Data Structures & Pointers

Pointers: Memory addresses, dereferencing, and value vs. reference mechanics.Arrays: Fixed-length continuous memory blocks and allocation limitations.Slices: Dynamically resized views into arrays, leveraging append, copy, and slicing.Maps: Unordered key-value hash collections, creation, lookups, and deletion.Structs: Creating user-defined compound types to encapsulate related data fields

Module 4

Object-Oriented Patterns & Interfaces

Methods: Attaching behavior to structs via value or pointer receivers.Composition: Embedding structs inside other structs instead of traditional class inheritance.Interfaces: Implicit implementation, duck typing, and building decoupled systems.Type Assertions: Extracting concrete types from empty interfaces (interface{} or any)

Module 5

Errors & Package Management

Error Handling: Navigating idiomatic explicit checks (if err != nil) instead of try/catch blocks.Panic & Recover: Triggering runtime failures and implementing crash recovery wrappers.Go Modules: Initializing dependencies with go mod init and tracking versions via go.mod.Visibility Rules: Mastering capitalized (exported) vs. lowercase (unexported) identifiers

Module 6

Concurrency (The Core of Go)

Goroutines: Spawning lightweight execution threads asynchronously using the go keyword.Channels: Communicating safely between goroutines using buffered and unbuffered pipelines.Select Block: Multiplexing over multiple channel inputs or timeouts.Sync Package: Thread synchronization using sync.WaitGroup and protecting memory via sync.Mutex.Race Conditions: Detecting multithreading defects using the built-in Go race detector

Module 7

Production Features & Projects

Standard Library: Leveraging the net/http package to launch robust web services.Testing & Benchmarks: Writing unit tests with the native testing framework.Capstone Application: Engineering a functioning RESTful API or high-performance CLI utility

© 2026 Proasia Upskilling Sdn Bhd | www.proasia.my