

The Go Language

Goals, Workspaces, Go Type System, CGo, Packaging, Concurrency, Remote Repositories

The Go Language (GoLang) is a next-generation system programming language that is rapidly growing in popularity. Many wildly successful commercial and open-source projects such as Docker and Kubernetes are written in Go. It is appreciated for a range of innovations, such as built-in concurrency including its channel architecture, simplified tooling, sensible conventions, and lots more. This course is aimed at experienced developers and brings them quickly up to speed programming Go and being able to understand and enhance existing Go source trees and being able to build their own.

We cover creating Go commands and libraries, package management, interaction with repositories, structural typing. We look at Go's rich range of system packages. We see how lack of classes and inheritance is not a problem. We explore the use of Go in a range of popular projects and see what real-world benefits it brings to system-level programming.

Go is the language of choice for modern low-level work and it is increasingly being selected by cutting-edge system developers for their most challenging projects. This course helps each attendee become one.

Contents of One-Day Training Course	
<p>Target Audience Developers wishing to create modern system-level applications using the Go language.</p> <p>Prerequisites Software developers with an object-oriented and system programming (e.g. concurrency) background.</p> <p>Knowledge of C or other system-level language is a plus.</p> <p>Notes All samples and labs in this course use Go v1.11 - the very latest production release.</p> <p>Our instructor will use the GoLand IDE; attendees may use any suitable editor they wish to select</p>	<p style="text-align: center;">Go Introduction</p> <p>What problem set is Go trying to solve? A sensible evolution of C Excellent for system programming Setting up Go on your dev machine</p> <p style="text-align: center;">Tour Of Language Features</p> <p>Goroutines Packaging Building Functions as first class citizens Interfaces and structural typing Lack of classes; no inheritance</p> <p style="text-align: center;">Workspace Management</p> <p>The Go workspace is how code is managed Naming and file placement conventions Handling packages Building & using commands and libraries Role of \$GOPATH <code>go build</code> vs. <code>go install</code> Creating / importing packages / exports</p> <p style="text-align: center;">Go Type System</p> <p>bools, strings, ints, runes (codepoints), floats, complex Zero (default) values Conversions Type inferencing Variables with <code>var</code>, <code>:=</code>, constants</p> <p style="text-align: center;">Go Functions</p> <p>Defining function signatures Returning more than one result Named returns Advanced function usage</p> <p style="text-align: center;">Constructs</p> <p><code>if</code> and <code>if-else</code> statements <code>for</code> loop / <code>defer</code> statement <code>switch</code> (how <code>break</code> is different) ("C's <code>while</code> is spelled <code>for</code> in Go" !!)</p> <p style="text-align: center;">Grouping of Data Items</p> <p>Arrays and maps Slices Structs and pointers</p> <p style="text-align: center;">Types and Their Interfaces</p> <p>Methods – functions with a receiver arg Interfaces define sets of methods</p> <p style="text-align: center;">Concurrency</p> <p>Concurrency primitives built into Go itself goroutines and channels Synchronization</p> <p style="text-align: center;">Testing</p> <p>Go has a built in testing system Test preparation The testing package Executing tests with <code>go test</code></p> <p style="text-align: center;">CGo – Calling C Code</p> <p>Most OS APIs are written in C Need to call them and other C libraries How the pseudo-package C works How Go and C code can interact</p> <p style="text-align: center;">Remote Packages</p> <p>How to access remote repositories Use of <code>go get</code> Incorporating remote repositories into app</p> <p style="text-align: center;">Project</p> <p>Review of Go usage in a larger project.</p>