Go Programming

Audience Course Go Programming

The course Go Programming is intended for developers who want to learn how to program in the Go language and who want to examine its capabilities.

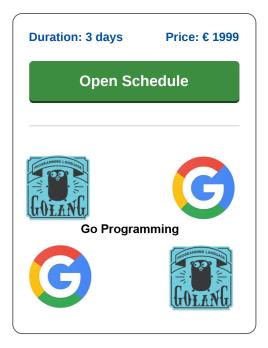### Prerequisites training Go Programming

To participate in this course prior knowledge of and experience with programming in a modern programming language such as Java, C# or Python is necessary.

Realization course Go Programming

The theory is treated on the basis of presentations. Illustrative demos are used to clarify the concepts. There is ample opportunity to practice and theory and exercises are interchanged. The course times are from 9.30 to 16.30.

Certification course Go Programming

Participants receive an official certificate Go Programming after successful completion of the course.

Duration: 3 days   **Price: € 1999**

Open Schedule

Go Programming

Content Course Go Programming

In the course Go Programming participants learn to develop applications using the concise and efficient Go programming language. Go is syntactically similar to C, but with the added benefits of memory safety, garbage collection and structural typing. The concurrency mechanisms in Go are based on Communicating Sequential Processes (CSP) and Go programs can get the best out of multicore machines. Go is a fast statically typed and compiled language but feels like a dynamically typed and interpreted language.

Go Intro

The course starts with a discussion of Go's syntax with data types, type inference, arrays, control flow and operators. The difference between rvalues and lvalues is also covered and attention is paid to immutable data.

Functions and Closures

The functions and parameter passing in Go is treated. The distinction between call by value and call by reference is explained. Variadic functions, recursion and closures are also discussed.

Pointers

Then it's time to pay attention to the use of pointers in Go. The difference with pointers in C is explained as well as pointer arithmetic, nil pointers and pointers to pointers.

Classes

Classes in Go are also part of program. The course explains the two ways in which Go offers an alternative to traditional inheritance. The first is embedding and can be seen as an automated form of composition or delegation. The second is the use of Go interfaces, which provide runtime polymorphism.

Error Handling

Also the specific way of error handling in Go with the error and panic interface is discussed. In this respect attention is also paid to the recover interface and the analysis of stack traces.

Concurrency

Finally concurrency in Go with Go routines, sending and receiving with channels and worker pools and synchronization mechanisms are covered.

SpiralTrain BV
Standerdmolen 10, 2e verdieping
3995 AA Houten

info@spiraltrain.nl
www.spiraltrain.nl
Tel.: +31 (0) 30 – 737 0661

Locations
Houten, Amsterdam, Rotterdam, Eindhoven, Zwolle, Online

Modules Course Go Programming

Module 1 : Intro Go	Module 2 : Language Syntax	Module 3 : Operators and Control Flow
Origins of Go	Identifiers	Operator Types
Features of Go	Data Types	Miscellaneous Operators
Compilation Model	Integers and Floats	Operator Precedence
Type Inference	Strings and Booleans	if and else
Concurrency Support	Derived Types	Nested if
Go Routines	Variable Declaration	switch Statement
Native Binaries	Static Type Declaration	select Statement
Intentionally Exclusions	Type Inference	for Loop
Inheritance	lvalues and rvalues	Nested Loops
Operator Overloading	Constants	Infinite Loops
Installing Go	String Literals	range Keyword
Comments	UTF-8 Sequences	break and continue
Go Programs	Immutability	goto Statement
Module 4 : Functions	**Module 5 : Arrays**	**Module 6 : Pointers**
Function Definition	Array Types	Address Operator
Function Declaration	Declaring Arrays	Pointer Type
Calling Functions	Initializing Arrays	Accessing Pointers
Local and Global Variables	Accessing Array Elements	Pointer Arithmetic
Parameters	Multi Dimensional Arrays	Comparison C Pointers
Return Values	Passing Arrays	Uage of Pointers
Call by Value and by Reference	Variables and Adresses	Dereferencing Pointers
Functions as Values	Slices	Nil Pointers
Function Closures	Nil Slice	Array of Pointers
Methods	Subslicing	Pointer to Pointer
Variadic Functions	len() and cap() Functions	Pointers as Parameters
Recursion	append() and copy() Functions	Type Casting
Module 7 : Data Structures	**Module 8 : Error Handling**	**Module 9 : Concurrency**
User Defined Data Types	Error is Type	Concurrency versus Parallelism
type Statement	Error Interface	Goroutines versus Threads
struct Keyword	Panic Interface	Multiplexing
Structure Definition	Reasons for Panic	Channels
Accessing Structure Members	Comparison to Exceptions	Race Conditions and Deadlock
Passing Structures	Using Multiple Return Values	Multiple Goroutines
Pointers to Structures	Unrecoverable Error	Declaring Channels
Object Oriented Programming	Programmer Error	Sending and Receiving with Channels
Structs versus Classes	Defer Execution	Blocking by Default
Composition over Inheritance	Recover Interface	Unidirectional Channels
Polymorphism	Runtime panics	Buffered Channels
Maps, Keys and Values	Goroutines	Worker Pools
make() and delete() Function	Stack Trace	WaitGroup
Interfaces	Analyzing Stack Traces	Select and Mutex