Erste Schritte für Softwareentwickler:

Golang für ARM-Linux-Systeme

Diese Anleitung dient als Einstieg in die Entwicklung von individuellen Kommunikations- und Automatisierungslösungen auf der Basis von ARM-Linux-Systemen in Go.

Die von Google entwickelte Sprache ist zwar schnell und leicht zu erlernen, im Gegensatz zu Scriptsprachen aber auch sehr mächtig und performant. Sie bringt von Hause aus einen Cross-Compiler für ARM-Linux-Systeme mit und bietet so einen einfachen Workflow für die Programmierung der pure.box, dem vielseitigen Kommunikations- und Automatisierungsserver von Wiesemann & Theis.

Nach der Installation und Konfiguration einer Arbeitsumgebung wird in diesem Tutorial der grundlegende Umgang mit dem Gotool über die Kommandozeile anhand eines einfachen Hello-World-Programms für die pure.box gezeigt.

Schritt 1: Go installieren

Go ist eine freie Software und kann von der Projektseite heruntergeladen werden. Alternativ können Sie Go aus den Paketquellen installieren. Nach der Installation müssen einige Umgebungsvariablen gesetzt, bzw. angepasst werden:

- **PATH**
  Die Umgebungsvariable PATH muss um das Unterverzeichnis **bin** im GO-Verzeichnis erweitert werden, damit Go systemweit ausgeführt werden kann.
- **GOROOT**
  Die Umgebungsvariable GOROOT enthält den Pfad zum Verzeichnis der Go-Installation. Diese Variable wird nur dann gesetzt, wenn Go in ein anderes Verzeichnis als das Standardverzeichnis /**usr**/**local**/**go** installiert wurde.
- **GOPATH**
  Die Umgebungsvariable GOPATH enthält den Pfad zum Arbeitsverzeichnis des Benutzers. Dieses Verzeichnis enthält später die Unterverzeichnisse **src** für Quellen, **lib** für vorkompilierte Bibliotheken und **bin** für ausführbare Dateien.

Setzen der Umgebungsvariablen
Setzen Sie zunächst die Variable GOPATH auf ihr persönlichen Arbeitsverzeichnis. Dazu bearbeiten Sie die Datei .bashrc in Ihrem Home-Verzeichnis und fügen dort beispielsweise die folgende Zeile ein.

```
export GPOATH=~/go
```

Wenn Sie Go nicht aus den Paketquellen installiert haben, müssen Sie nun die Variable GOROOT setzen und die Variable PATH erweitern. Diese Änderungen werden sinnvollerweise für das gesamte System gesetzt. Bearbeiten Sie daher mit root-Rechten die Datei /etc/profile und fügen Sie hier die beiden folgenden Zeilen ein:

```
#Pfad zum Installationsverzeichnis von Go
export GOROOT=/usr/local/go
export PATH=$PATH:$GOROOT/bin
```

Git installieren
Um ohne großen Aufwand Spracherweiterungen und Pakete von Drittanbietern zu installieren, verwendet Go Git, die von Linus Torvalds entwickelte Software für verteilte Versionsverwaltung. Diese kann von www.git-scm.com herunterladen oder über die Paketverwaltung installieren werden.

Installation überprüfen
Das Go-Tool ist das Schweizer Taschenmesser für die Go-Entwicklung. Um zu überprüfen, ob die Installation erfolgreich war, können Sie damit nun die Go-Umgebung anzeigenlassen

Geben Sie im Terminal das folgende Kommando ein:

```
go env
```

Es wird nun eine Übersicht über alle für Go relevanten Umgebungsvariablen ausgegeben:

```
GOARCH="amd64"
GOBIN=""
GOEXE=""
GOHOSTARCH="amd64"
GOHOSTOS="linux"
GOOS="linux"
GOPATH="/home/martin/go"
GORACE=""
GOROOT="/usr/lib/go"
GOTOOLDIR="/usr/lib/go/pkg/tool/linux_amd64"
CC="gcc"
GOGCCFLAGS="-fPIC -m64 -pthread -fmessage-length=0 -fdebug-prefix-map=/tmp/go-build863343041=/tmp/go-build
CXX="g++"
CGO_ENABLED="1"
```

Bekommen Sie eine Fehlermeldung angezeigt, ist das bin-Verzeichnis im **GOROOT** wahrscheinlich nicht im Suchpfad. Untersuchen Sie die Umgebungsvariable **PATH** mit dem Befehl **echo %PATH%**. Überprüfen Sie dann, ob **GOPATH** ihrem Arbeitsverzeichnis entspricht.

Schritt 2: Hello World

Die Entwicklung von Go-Programmen erfolgt innerhalb des GOPATH im Verzeichnis `src`. Legen Sie hier das Verzeichnis `hello` an und darin die Datei `hello.go`.

```
1 package main

3 import "fmt"

5 func main() {
6  fmt.Println("Hello, world!")
7 }
```

Zeile 1:
Wenn Sie in Go eine ausführbare Datei erzeugen wollen, arbeiten Sie im Paket **main**.

Zeile 3:
Das Paket **fmt** stellt Funktionen für die Ein- und Ausgabe bereit.

Zeile 5:
Die Funktion **main()** ist der Einstiegspunkt in das Programm.

Zeile 6:
fmt.Println() gibt einen String, gefolgt von einem Newline-Zeichen aus.

Um das Programm zu kompilieren, geben Sie nun das folgende Kommando ein:

```
go install github.com/wut/tutorial/hello
```

Wenn Sie keine Fehlermeldung angezeigt bekommen, ist der Übersetzungsvorgang erfolgreich gewesen. Die ausführbare Datei hello.exe finden Sie nun im bin-Verzeichnis ihres Go-Path.

```
./hello
Hello World!
```

Schritt 3: Hello pure.box - Cross-Compilation für ARM-Linux

Go macht es besonders einfach, Applikationen für andere Zielplattformen zu kompilieren. Um das Hello-World-Beispiel für die pure.box zu übersetzen, müssen Sie lediglich temporär ein paar Umgebungsvariablen setzen:

- **GOARCH**
  Mit der Umgebungsvariable GOARCH wird die Zielarchitektur gesetzt. Für die pure.box ist das **arm**.
- **GOOS**
  Dies ist das Betriebssystem der Zielplattform. Für die pure.box muss diese Variable auf **linux** gesetzt werden.
- **GOARM**
  Mit GOARM wird dem Compiler mitgeteilt, wie er mit Floatingpoint-Operationen umgehen soll, der Wert für die pure.box ist **5**.

Geben Sie dazu folgendes in der Eingabeaufforderung ein:

```
export GOOS=linux
export GOARCH=arm
export GOARM=5
```

Die plattformübergreifende Übersetzung starten Sie, indem Sie in das Quellcode-Verzeichnis wechseln und dort **go build** ausführen:

```
go build
ls -al
total 1476
drwxr-xr-x 2 gopher users    4096 Dec 25 11:31 .
drwxr-xr-x 3 gopher users    4096 Dec 25 12:08 ..
-rwxr-xr-x 1 gopher users 1496747 Dec 25 11:31 hello
-rw-r--r-- 1 gopher users      88 Dec 25 12:09 hello.go
```

Wie Sie sehen, existiert in ihrem Arbeitsverzeichnis nun die ausführbare Datei **hello**. Cross-Compilation mit Go ist tatsächlich so einfach!

Schritt 4: Ausführen auf der pure.box

Um das Binary auf die Box zu übertragen existieren verschiedene Möglichkeiten: Neben dem Zugriff über FTP kann die Box über SMB auch direkt in ihr Windows-Netzwerk eingebunden werden.

Da der einfachste Weg eine Anwendung auszuprobieren darin besteht, sie in einer Terminalsitzung manuell auszuführen, sollte SSH auf der Box aktiviert sein. Somit ist die Dateiübertragung über SCP naheliegend.

Wenn Sie noch keinen SSH-Client installiert haben, installieren Sie das Paket open-ssh. Für SCP benötigen Sie die folgenden Parameter: Der **Benutzername** ist "admin", das **Passwort** ist das von Ihnen konfigurierte Administrationspasswort. Als **Hostnamen** geben Sie Hostnamen oder die IP-Adresse Ihrer pure.box an.

Datei kopieren:

```
scp hello admin@hostname:/data/userfiles
```

Programm ausführen:
Verbinden Sie sich mit Putty oder über das ssh-Kommando mit der pure.box.

```
ssh admin@hostname
```

Führen Sie anschließend die Datei **hello** im Verzeichnis **/data/userfiles** aus.

```
./hello
Hello pure.box
```

Offene Fragen? Rufen Sie an!

Haben Sie noch offene Fragen zu dieser Anleitung? Unser Support hilft Ihnen gerne weiter!

Wir sind gerne persönlich für Sie da:

Wiesemann & Theis GmbH
Porschestr. 12
42279 Wuppertal
Tel.: 0202/2680-110 (Mo-Fr. 8-17 Uhr)
Fax: 0202/2680-265
info@wut.de