Erste Schritte für Softwareentwickler:

Golang auf Windows 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.

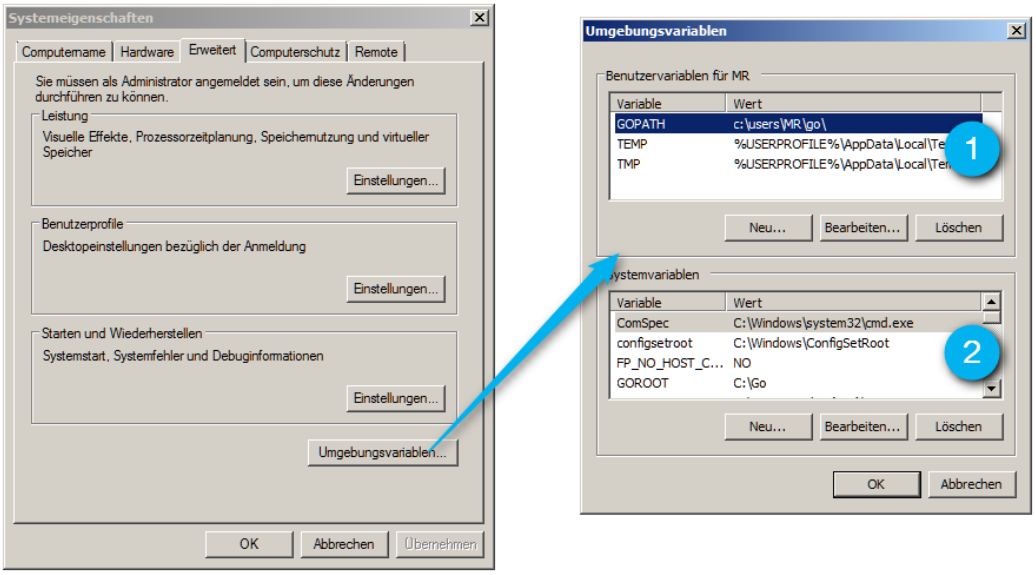
Nach der Installation und Konfiguration einer Arbeitsumgebung wird in diesem Tutorial der grundlegende Umgang mit dem Gotool über die Kommandozeile anhand eines einfachen Hello-World-Programms für die pure.box gezeigt.

Schritt 1: Go installieren

Go ist eine freie Software und kann vonder Projektseite heruntergeladen werden. 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 **C:\Go unter Windows** 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
Klicken Sie in der **Systemsteuerung** auf **System**, dann auf **Erweiterte Systemeinstellungen**. Hier finden Sie die Schaltfläche **Umgebungsvariablen**.



Im oberen Bereich finden Sie die Umgebungsvariablen für den aktuellen Benutzer (1). Im unteren Bereich finden Sie die systemweiten Umgebungsvariablen (2). Legen Sie die Benutzervariable **GOPATH** an und weisen Sie ihr den Pfad zu Ihrem Arbeitsverzeichnis zu. Falls notwendig, legen Sie bei den Systemvariablen die Variable GOROOT an. Erweitern Sie jetzt die Systemvariable **PATH** um das Verzeichnis **bin** in Ihrem GOROOT, also beispielsweise C:\Go\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 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 anzeigen lassen.

Offnen Sie die Eingabeaufforderung und geben Sie das folgende Kommando ein:

```
go env
```

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

```
C:\Users\Gopher\go>go env
set GOARCH=amd64
set GOBIN=
set GOEXE=.exe
set GOHOSTARCH=amd64
set GOHOSTOS=windows
set GOOS=windows
set GOPATH=c:\users\Gopher\go\
set GORACE=
set GOROOT=C:\Go
set GOTOOLDIR=C:\Go\pkg\tool\windows_amd64
set CC=gcc
set GOGCCFLAGS=-m64 -mthreads -fmessage-length=0 -fdebug-prefix-map=C:\Users\Gopher\
AppData\Local\Temp\go-build015282382=/tmp/go-build -gno-record-gcc-switches
set CXX=g++
set 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
2
3  import "fmt"
4
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 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:

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

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

```
go build

dir
Volume in Laufwerk C: hat keine Bezeichnung.
Volumeseriennummer: DADE-DEDA

Verzeichnis von C:\Users\gopher\go\src\hello

05.04.2017  10:51    <DIR>          .
05.04.2017  10:51    <DIR>          ..
09.12.2016  12:54                78 hello.go
05.04.2017  10:51         1.706.496 purebox

               2 Datei(en),      1.706.574 Bytes
               2 Verzeichnis(se), 61.440.692.224 Bytes frei
```

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 auf der Box SSH aktiviert sein. Somit ist die Dateiübertragung über SCP naheliegend.

Wenn Sie noch keinen SSH-Client installiert haben, laden Sie beispielsweise Putty herunter. 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:

```
pscp 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? UnserSupport 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