Go Installfest

24 December 2016

Thach Le
Triet Pham

Who are we?

- Programming addict
- Java, NodeJS, Go
- Backend at DF/Backend at Lozi
- 1.5 years with Go

runikitkat.com (http://runikitkat.com)

Goal

Help new gophers know what exactly they need to start using Go.

Agenda

- Install, set up Go
- Go commands + tools
- Glide
- Beego
- Gin
- Gorm
- gRPC

Install, set up Go

Install Go

- Using HomeBrew
- Download from Homepage(https://golang.org/doc/install)
- Install from source.

Set up Go

- GOPATH
- GOBIN
- GOROOT
- PATH

Go commands + tools

Commands

- go build
- go test
- go run
- go get
- go fmt
- go lint
- go vet

Integrate with Editor/IDE

- Vim: vim-go
- Emacs: go-mode
- Sublime: GoSublime
- Atom: go-plus
- VSCode: vscode-go

Hello world

```
package main

import "fmt"

func main() {
    fmt.Println("Hello world")
}
```

Go import

```
package main

import "github.com/k0kubun/pp"

func main() {
    pp.Println("Hello world")
}
```

Glide

Introduction

- Cargo, npm, pip, Maven...
- Packages manager

Install

- curl https://glide.sh/get | sh
- brew install glide

Usage

```
$ glide create                                # Start a new workspace
$ open glide.yaml                             # and edit away!
$ glide get github.com/Masterminds/cookoo # Get a package and add to glide.yaml
$ glide install                               # Install packages and dependencies
# work, work, work
$ go build                                    # Go tools work normally
$ glide up                                    # Update to newest versions of the package
```

Beego

Introduction

- http://beego.me/
- Fullstack web framework
- RESTful, MVC model
- High performance

Installation

- go get github.com/astaxie/beego
- go get github.com/beego/bee

Using

```
$ cd $GOPATH/src
$ bee new hello
$ cd hello
$ bee run
```

Gin

Introduction

- https://gin-gonic.github.io/gin/
- Minimal framework - but use for API much better
- Fast
- Easy to use

Installation

- go get github.com/gin-gonic/gin

Using

- import "github.com/gin-gonic/gin"

```
package main

import "github.com/gin-gonic/gin"

func main() {
    // Creates a gin router with default middleware:
    // logger and recovery (crash-free) middleware
    router := gin.Default()

    router.GET("/someGet", getting)
    router.POST("/somePost", posting)
    router.PUT("/somePut", putting)
    router.DELETE("/someDelete", deleting)
    router.PATCH("/somePatch", patching)
    router.HEAD("/someHead", head)
    router.OPTIONS("/someOptions", options)

    // By default it serves on :8080 unless a
    // PORT environment variable was defined.
    router.Run()
    // router.Run(":3000") for a hard coded port
}
```

Gorm

Introduction

- ORM library
- Developer friendly
- Auto migration
- Logger

Installation

- go get -u github.com/jinzhu/gorm

Using

- import "github.com/jinzhu/gorm"
- import _ "github.com/lib/pq"

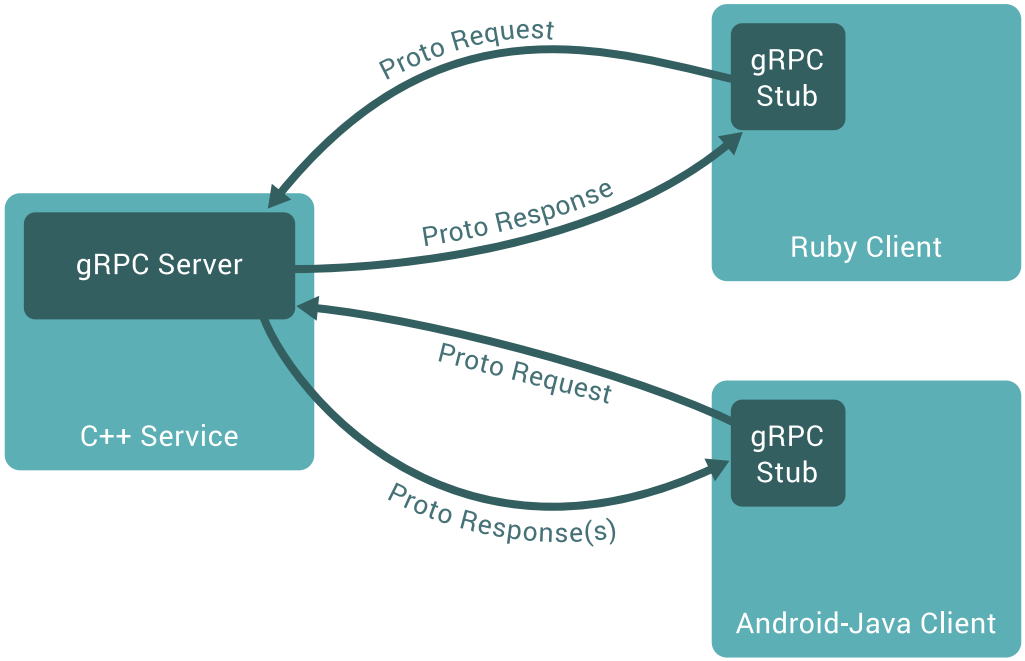
Code

```
type Product struct {
    gorm.Model
    Code  string
    Price uint
}

func main() {
    db, err := gorm.Open("sqlite3", "test.db")
    if err != nil {
        panic("failed to connect database")
    }
    defer db.Close()

    db.AutoMigrate(&Product{})

    db.Create(&Product{Code: "L1212", Price: 1000})

    var product Product
    db.First(&product, "code = ?", "L1212") // find product with code l1212

    db.Model(&product).Update("Price", 2000)

    db.Delete(&product)
}
```

gRPC

Introduction

- Distributed services and clients
- Across languages
- RPC
- Protobuf
- HTTP/2



Installation

- go get google.golang.org/grpc
- go get github.com/grpc/grpc-go

Using

- Define a service in a .proto file.
- Generate server and client code using the protocol buffer compiler.
- Use the Go gRPC API to write a simple client and server for your service.

Thank you

Thach Le
Triet Pham
thach@dwarvesf.com (mailto:thach@dwarvesf.com)
triet.phm@gmail.com (mailto:triet.phm@gmail.com)