链滴

一、go 语言编写 grpc 微服务实例

作者：450370050

原文链接：https://ld246.com/article/1524816248447

来源网站：链滴

grpc

grpc跨平台微服务框架，但是缺少服务治理的功能，服务发现给出了架子需要自己实现。

go包下载

go get -u google.golang.org/grpc

proto工具下载

1、下载idl代码生成工具

https://repo1.maven.org/maven2/com/google/protobuf/protoc/
protoc放在环境变量的path中

2、下载生成go代码插件

go get -u github.com/golang/protobuf/protoc-gen-go

从GoBin目录中拷贝出来 protoc-gen-go，放在环境变量path中

IDL

IDL语法，详细的请查看Protobuf3语言指南

例子

代码脚手架

1、编写book.proto文件

```
syntax = "proto3";

// 请求书详情的参数结构  book_id 32位整形
message BookInfoParams {
    int32 book_id = 1;
}

// 书详情信息的结构    book_name字符串类型
message BookInfo {
    int32 book_id = 1;
    string  book_name = 2;
}

// 请求书列表的参数结构  page、limit   32位整形
message BookListParams {
    int32 page = 1;
    int32 limit = 2;
```

```
}

// 书列表的结构    BookInfo结构数组
message BookList {
    repeated BookInfo book_list = 1;
}
// 定义 获取书详情  和 书列表服务    入参出参分别为上面所定义的结构
service BookService {
    rpc GetBookInfo (BookInfoParams) returns (BookInfo) {}
    rpc GetBookList (BookListParams) returns (BookList) {}
}
```

2、生成对应语言proto代码

protoc --go_out=plugins=grpc:. book.proto

当前文件夹下会生成book.pb.go文件，也可以生成其它语言的代码

protoc --php_out=. book.proto

3、解决小冲突

gol的grpc插件指定context路径contextPkgPath = "golang.org/x/net/context"可能会与我们代码的context冲突 把book.proto中import"golang.org/x/net/context" 更改为 "context"

服务端代码

```
package main

import (
   "grpc-test/pb"
   "net" "context" "google.golang.org/grpc"
)

/**
创建BookServer结构 实现 BookServiceServer接口
type BookServiceServer interface {
   GetBookInfo(context.Context, *BookInfoParams) (*BookInfo, error)
   GetBookList(context.Context, *BookListParams) (*BookList, error)
}
*/
type BookServer struct {}
func (s *BookServer) GetBookInfo(ctx context.Context, in *book.BookInfoParams) (*book.Book
nfo, error) {
   //请求详情时返回 书籍信息
   b := new(book.BookInfo)
   b.BookId = in.BookId
   b.BookName = "21天精通php"
   return b,nil
}

func (s *BookServer) GetBookList(ctx context.Context, in *book.BookListParams) (*book.BookL
st, error) {
   //请求列表时返回 书籍列表
```

```
    bl := new(book.BookList)
    bl.BookList = append(bl.BookList, &book.BookInfo{BookId:1,BookName:"21天精通php"})
    bl.BookList = append(bl.BookList, &book.BookInfo{BookId:2,BookName:"21天精通java"})
    return bl,nil
}

func main() {
    serviceAddress := ":50052"
    bookServer := new(BookServer)
    //创建tcp监听
    ls, _ := net.Listen("tcp", serviceAddress)
    //创建grpc服务
    gs := grpc.NewServer()
    //注册bookServer
    book.RegisterBookServiceServer(gs, bookServer)
    //启动服务
    gs.Serve(ls)
}
```

客户端代码

```
package main

import (
    "fmt"
    "grpc-test/pb" "google.golang.org/grpc"
    "context"
)

func main() {
    serviceAddress := "127.0.0.1:50052"
    conn, err := grpc.Dial(serviceAddress, grpc.WithInsecure())
    if err != nil {
        panic("connect error")
    }
    defer conn.Close()
    bookClient := book.NewBookServiceClient(conn)
    bi,_:=bookClient.GetBookInfo(context.Background(),&book.BookInfoParams{BookId:1})
    fmt.Println("获取书籍详情")
    fmt.Println("bookId: 1", " => ", "bookName:", bi.BookName)

    bl,_ := bookClient.GetBookList(context.Background(), &book.BookListParams{Page:1, Limit:1
})
    fmt.Println("获取书籍列表")
    for _,b := range bl.BookList {
        fmt.Println("bookId:", b.BookId, " => ", "bookName:", b.BookName)
    }
}
```

启动测试

启动server,client

```
GOROOT=D:\Go #gosetup
GOPATH=D:\go\gopath #gosetup
D:\Go\bin\go.exe build -i -o C:\Users\Administrator\AppData\Local\Temp\___17110go_build
main_go.exe D:/go/gopath/src/grpc-test/client/main.go #gosetup
D:\软件\GoLand\bin\runnerw.exe C:\Users\Administrator\AppData\Local\Temp\___17110go_b
ild_main_go.exe #gosetup
获取书籍详情
bookId: 1  =>  bookName: 21天精通php
获取书籍列表
bookId: 1  =>  bookName: 21天精通php
bookId: 2  =>  bookName: 21天精通java

Process finished with exit code 0
```