bobby · 更新于 2022-11-08

上一节 4. token认证 6. grpc中的异常... 下一节

protoc-gen_validate

1. 安装和配置

linux

代码块 预览 复制

```
# fetches this repo into $GOPATH
go get -d github.com/envoyproxy/protoc-gen-validate

# installs PGV into $GOPATH/bin
make build
```

windows

exe 下载地址

exe下载地址

可以点此下载：protoc-gen-validate.zip

将 zip文件中的exe文件拷贝到 go的根补录的bin目录下

生成python源码

代码块

```
protoc -I .  --go_out=plugins=grpc:. --validate_out="lang=go:." helloworld.proto
```

2. proto

1. 新建validate.proto文件内容从 https://github.com/envoyproxy/protoc-gen-validate/blob/master/validate/validate.proto 拷贝
2. 新建helloworl.proto文件

代码块

```
syntax = "proto3";

```

```
5
6   service Greeter {
7       rpc SayHello (Person) returns (Person);
8   }
9
10  message Person {
11      uint64 id    = 1 [(validate.rules).uint64.gt    = 999];
12
13      string email = 2 [(validate.rules).string.email = true];
14      string name  = 3 [(validate.rules).string = {
15                          pattern:   "^[^[0-9]A-Za-z]+( [^[0-9]A-Za-z]+)*$",max_bytes
16
17  }
```

3. 服务端

代码块

```
1   package main
2
3   import (
4       "context"
5       "google.golang.org/grpc/codes"
6       "google.golang.org/grpc/status"
7       "net"
8
9       "google.golang.org/grpc"
10
11      "start/pgv_test/proto"
12  )
13
14
15  type Server struct{}
16
17  func (s *Server) SayHello(ctx context.Context, request *proto.Person) (*proto.Per
18      error){
19      return &proto.Person{
20          Id: 32,
21      }, nil
22  }
23
24  type Validator interface {
25      Validate() error
26  }
27
28  func main(){
29      var interceptor grpc.UnaryServerInterceptor
30      interceptor = func(ctx context.Context, req interface{}, info *grpc.UnaryServ
31          // 继续处理请求
32          if r, ok := req.(Validator); ok {
33              if err := r.Validate(); err != nil {
34                  return nil, status.Error(codes.InvalidArgument, err.Error())
35              }
36          }
37
38          return handler(ctx, req)
39      }
40      var opts []grpc.ServerOption
41      opts = append(opts, grpc.UnaryInterceptor(interceptor))
42
43      g := grpc.NewServer(opts...)
```

意见反馈 收藏教程 标记书签

```
45      lis, err := net.Listen("tcp", "0.0.0.0:50051")
46      if err != nil{
47          panic("failed to listen:"+err.Error())
48      }
49      err = g.Serve(lis)
50      if err != nil{
51          panic("failed to start grpc:"+err.Error())
52      }
53  }
54
```

4. 客户端

代码块

```
1   package  main
2
3   import (
4       "context"
5       "fmt"
6       "google.golang.org/grpc"
7       "start/pgv_test/proto"
8   )
9
10  type customCredential struct{}
11
12
13  func main() {
14      var opts []grpc.DialOption
15
16      //opts = append(opts, grpc.WithUnaryInterceptor(interceptor))
17      opts = append(opts, grpc.WithInsecure())
18
19      conn, err := grpc.Dial("localhost:50051", opts...)
20      if err != nil {
21          panic(err)
22      }
23      defer conn.Close()
24
25      c := proto.NewGreeterClient(conn)
26      //rsp, _ := c.Search(context.Background(), &empty.Empty{})
27      rsp, err := c.SayHello(context.Background(), &proto.Person{
28          Email: "bobby",
29      })
30      if err != nil {
31          panic(err)
32      }
33      fmt.Println(rsp.Id)
34  }
35
```

4. token认证 ‹ 上一节 下一节 › 6. grpc中的异常处理

我要提出意见反馈