尝试搭建了一下grpc

首先grpc是什么:

gRPC 是一款高性能、开源的 RPC 框架，产自 Google，基于 ProtoBuf 序列化协议进行开发，支持多种语言（Golang、Python、Java等），本篇只介绍 Python 的 gRPC 使用。因为 gRPC 对 HTTP/2 协议的支持使其在 Android、IOS 等客户端后端服务的开发领域具有良好的前景。gRPC 提供了一种简单的方法来定义服务，同时客户端可以充分利用 HTTP/2 stream 的特性，从而有助于节省带宽、降低 TCP 的连接次数、节省CPU的使用等。

# **安装：**

1. gRPC 的安装：

$ pip install grpcio

1. 安装 ProtoBuf 相关的 python 依赖库：

$ pip install protobuf

1. 安装 python grpc 的 protobuf 编译工具：

$ pip install grpcio-tools



这个文件是proto配置文件，通过这个配置文件，利用编译cmd如下

python -m grpc\_tools.protoc --python\_out=. --grpc\_python\_out=. -I. helloworld.proto

编译完成之后会生成如下的文件：



helloworld\_pb2.py: 用来和 protobuf 数据进行交互

helloworld\_pb2\_grpc.py: 用来和 grpc 进行交互

然后将如下文件放到一起就可以执行了



Server是服务端

Client是客户端

运行 python helloworld\_grpc\_server.py 和 python helloworld\_grpc\_client.py, 就可以看到效果了

# **grpc basic: 4 种通信方式**

helloworld 使用了最简单的 grpc 通信方式: 类似 http 协议的一次 request+response.

根据不同的业务场景, grpc 支持 4 种通信方式:

* 客服端一次请求, 服务器一次应答
* 客服端一次请求, 服务器多次应答(流式)
* 客服端多次请求(流式), 服务器一次应答
* 客服端多次请求(流式), 服务器多次应答(流式)

官方提供了一个 route guide service 的 demo, 应用到了这 4 种通信方式, 具体的业务如下:

* 数据源: json 格式的数据源, 存储了很多地点, 每个地点由经纬度(point)和地名(location)组成
* 通信方式 1: 客户端请求一个地点是否在数据源中
* 通信方式 2: 客户端指定一个矩形范围(矩形的对角点坐标), 服务器返回这个范围内的地点信息
* 通信方式 3: 客户端给服务器发送多个地点信息, 服务器返回汇总信息(summary)
* 通信方式 4: 客户端和服务器使用地点信息 ****聊天(chat)****