- [ ] grpc-web 用于浏览器客户端
- [ ] grpc-js node版本12https://www.npmjs.com/package/@grpc/grpc-js
- [ ] egg-grpc https://github.com/eggjs/egg-grpc
- [ ] typescript

一、grpc相关文档

1. https://docs.microsoft.com/zh-cn/aspnet/core/grpc/?view=aspnetcore-3.1

grpc-load https://github.com/grpc/grpc-node/tree/master/packages/proto-loader

加载proto文件，可以同时加载一个或多个；
入参：proto文件的绝对或相对路径+其他参数

二、论坛参考

探讨gRPC的Node技术生态及实现工具 https://xenojoshua.com/2018/02/grpc-node-ecosystem/
gRPC学习笔记 https://skyao.io/learning-grpc/
服务注册和发现https://juejin.im/post/6844903811237019661#heading-1
https://www.jianshu.com/p/c957a9100562
https://segmentfault.com/a/1190000021194493

三、注意事项

1. 启动client端时 server端要先ready 否则会报错

四、疑问

1、为什么

```
// 回调函数的第一个入参为什么必须是错误对象?
// 如果不按这个规则来 就会报未知错误
greeter.Add({ num: 1, num2: 2 }, (e, res) => {
    data = `hi, egg, Add,${res.num}`
    debug(`debug test:${data}`)
    resolve(data)
})
```

五、笔记

1、grpc-js vs grpc

node版本：12【但是本地使用v10.15.1 也正常启动了且可以运行】
两者不同：https://www.npmjs.com/package/@grpc/grpc-js#migrating-from-grpc
大意如下
1、对于grpc客户端：用 @grpc/proto-loader 代替grpc.load加载proto文件；使用@grpc/grpc-js 的 grpc.loadPackageDefinition加载生成的文件包
2、对于服务端：Server#bindAsync 代替`Server#bind`
3、如果当前正在加载由grpc-tools生成的软件包，则应该使用grpc-tools中的generate_package_definition选项生成文件，然后使用grpc.loadPackageDefinition将生成的文件导出的对象加载到@ grpc / grpc-js中

六、TODO

- [ ] HTTP2 及vs HTTP1
- [ ] 调用流程各个节点的作用及参数含义
- [ ] 各个节点内部源码逻辑，文件作用

- [ ] 项目实施