消息服务器**详细文档**

1. 系统设计图

2. 系统详细设计图

1. 系统实现

Message Server 结构和一些文件的说明

1. /message\_server/core/ 文件夾

.

├── config.js

├── logger\_client.js

├── message\_router.js

├── message\_server.js

├── message\_server.log

└── protobuf\_message.desc

- config.js ：里面是message server相关的配置，mysql ,redis, gearman 等

- logger\_client.js ：log4j的一个导出的模块，里面是配置logger的信息

- message\_router.js ：message router对不同消息进行路由处理,把消息的处理过程从message server的代码中分离

- message\_server.js ： message server是message server的核心代码，建立连接，事件监听等

- message\_server.log ： message server相关的log文件

- protobuf\_message.desc ：protobuf\_message是node使用的protobuf描述文件，用来解析protobuf封装的消息

2. /message\_server/service/ 文件夹

.

├── message\_client.py

├── message\_service.py

└── protobuf\_message\_pb2.py

- message\_client.py ：message client提供一个连接message server的功能，和发消息的功能

- message\_service.py ：为django提供一个gearman的接口，调用消息客户端发送消息

- protobuf\_message\_pb2.py ：python能识别的protobuf文件

3. /message\_server/package.json 文件

- package.json ：记录message server开发的版本信息，依赖库，作者等信息

＋如何使用package.json

当需要添加一些node依赖包，需要现在package.json里面配置，然后使用 sudo npm install

4. /message\_server/node\_modules 文件夹

－ node\_modules ：用来存放node一些依赖包

5. /message\_server/protobuf\_message.proto 文件

－ protobuf\_message.proto ：是protobuf的源文件，通过不用的编译命令可以生成不同环境所需要的文件

＋ 如何生成node protobuf 能读取的desc 文件

protoc --descriptor\_set\_out = protobuf\_message.desc --include\_imports protobuf\_message.proto

＋ 如何生成python protobuf pb2文件

protoc protobuf\_message.proto --python\_out=. ("."为当前目录，可以指定其他路径)

1. 系统部署

* 注意事项

1.修改protobuf\_message.proto文件之后需要通知安卓端，IOS端，WEB端，同时需要重新生成java，c++，python，nodejs四个版本的proto文件，所以修改proto文件的时候一定要谨慎。

2.启动messsage\_server的时候需要用supervise来启动，保证message\_server进程挂掉之后能自动重启。

3.日志文件，在message\_server不同逻辑里面记录对应的日志，通常用的日志类型有 error,info.debug等

4. 依赖库的安装，message\_server所有依赖库都写在package.json文件里，在message\_server 目录下，可以通过运行 npm install 来完成依赖的安装。如果以后需要添加其他的依赖库，需要及时更新到package.json文件，格式是“库名”：“版本号”

* 启动服务

1. 先启动message\_server/core里面的message\_server.js，可以用命令 node message\_server.js ，同样也可以尝试用forever等库来启动和管理node进程

2. 启动和消息服务器相关的gearman脚本

├── gearman\_worker\_message\_client.py

├── gearman\_worker\_update\_location.py

└── gearman\_worker\_update\_message.py

gearman\_worker\_message\_client.py 用于django向消息服务之间的消息通信

gearman\_worker\_update\_location.py 用于更新用户地理位置

gearman\_worker\_update\_message.py用来更新用户发送的消息记录

3. 调试和测试

如果想手动给message\_server发送消息，可以开启一个django 的shell 直接调用发消息的gearman接口，这样可以手动的测试消息。也可以通过两个消息客户端来测试消息