EasyChat项目服务器客户端通信协议

通信数据使用json数据格式

示例：

以账号登录为示例，

账号： 123456 密码：abcdef

对应的json数据： {"purpose":"1","account":"123456","password":"abcdef"}

Json字符串字面量： "{\"purpose\":\"1\",\"account\":\"123456\",\"password\":\"abcdef\"}"

purpose 对应消息的目的，可能是登陆，注册，聊天消息 等等

实际发给服务器的string（对Json字符串字面量每一个单个字符进行位运算加密）：

"\fU\u0007\u0002\u0005\u0007\u0018\u0004\u0012UMU\u001B\u0018\u0010\u001E\u0019U[U\u0016\u0014\u0014\u0018\u0002\u0019\u0003UMUFEDCBAU[U\u0007\u0016\u0004\u0004\u0000\u0018\u0005\u0013UMU\u0016\u0015\u0014\u0013\u0012\u0011U\n"

服务器对收到的string进行字符位运算解加密 得到Json字符串字面量

然后使用json相关的解析函数得到json数据

最后通过判断purpose的值来获取客户端的请求完成不同的操作

注： 所有字符驼峰命名

建议：因为客户端和服务器之间通信操作较多，如果每次都通过比较purpose对应的值来确定操作，性能消耗较大，可以将这些操作的每个字符对应为一个枚举值，通过一个哈希表（hash）来保存，字符作为key 枚举值作为 value 然后用一个switch语句来区别执行不同的操作（感觉这样不用每次都去通过级联式的if语句比较字符串，性能估计会好点，参考一下）

## 登录

C to S : {"purpose":"1","account":"123456","password":"abcdef"}

S to C : {"purpose":"1","result":"ok"}

result

result":"ok" 登录成功

result":"wrong\_password" 密码错误

result":"invalid\_user" 无效用户（账号不存在）

## 注册

C to S :{"purpose":"2","nickName":"nick","password":"wewewe","age":"13","sex":"女"}

S to C : {"purpose":"2"," account ":"服务器分配的账号"}

注：账号最低6位最多11位

## 查找

C to S : {"purpose":"3","account":"123456"}

该账号不存在：

S to C : {"purpose":"3","result":"invalid"}

该账号存在 :

S to C : {"purpose":"3","result":"ok","nickName":"nick","age":"13","sex":"女"}

## 聊天消息的发送

C to S: {"purpose":"4","sender":"123456","receiver":"135556","message":"消息","sendTime":"2017-03-27 14:37:49"}

Server： 服务器通过解析该json字符，获取到receiver的ip地址，将该消息转发给receiver即可。如果离线，就先保存，待其上线再转发

S to C: {"purpose":"4","sender":"123456","receiver":"135556","message":"消息","sendTime":"2017-03-27 14:37:49"}

## 获取好友列表

C to S: {"purpose":"5","sender":"123456"}

S to C: {"purpose":"5","groupName":"我的好友", “account”:”好友的账号”,”nickName”:”好友昵称”,”remarksName”:”好友备注名”}

## 获取群聊列表

C to S: {"purpose":"8","sender":"123456"}

S to C: {"purpose":"8","groupAccount":"群组号", “groupName”:”组名”,”groupOwner”:”群主”,”remarksName”:”群备注名”}

## 添加好友请求

C to S: {"purpose":"6","initiated":"发起添加请求的账号","aims":"目标账号"}

S to C: {"purpose":"6=","initiated":"发起添加请求的账号","aims":"目标账号"}

注意，这个S to C是发送到目标账号的客户端，如果不在线就当作离线消息处理

## 同意添加好友

C to S: {"purpose":"7","initiated":"发起请求的账号","aims":"目标账号"}

S to C: {"purpose":"7=","initiated":"发起添加请求的账号","aims":"目标账号"}

注意，这个S to C是发送到发起请求的账号的客户端，如果不在线就当作离线消息处理

同时刷新两个人的好友列表（或者把这两个账号的信息分别发送给两个账号所在线的客户端） 两个方案二选一 待定

下线

C to S: {"purpose":"9","account":””}