祉来好！

目前实现的代码，象你所言，已经有相当的稳定性的提升，可以先保留一个版本吧。

目前，基于Duplicate server服务端只能打开一个Firefox 上的web 网页的实际情况，为了实现多用户的对话聊天，我们需要改变方案，做以下调整。

1. server 服务端，默认以一个固定的帐号/密码, 登录 my.replika.ai .
2. server 服务端，接受 duplicate app 端的网络链接。 duplicate app 登录的帐号/密码， 为 duplicate 系统的帐号/密码， 而非 replika 系统的帐号密码。（目前实现，也可以先不验证 app 端的帐号/密码，允许起登录。）
3. app 发送到server的信息， 在server 端缓存在一个消息队列里。
4. server 从消息队列里取信息，（翻译后）通过Web网页发送给 replika 服务端，得到Web网页返回的聊天信息后，（翻译后）发送给相应的app 。
5. 重复 步骤4 .
6. 如server 长时间没有信息发送给 replika , 可能和 replika 的Web 网页session连接会超时中断. 此后，当 server 有消息发送给 replika 的时候，可能会返回失去会话此类的错误，此时，需要重新 执行 步骤1， 以固定账号/密码，登录my.replika.ai , 登录成功后， 再重发该消息。

1. 为什么需要消息队列缓存？

因为有多个app 向 server 发送消息。 对于每个消息，要等待到 replika 返回应答的消息后，返回给app 后， 这轮对话事务才算完结的。

所以，需要消息队列来缓存在这个一来一回整个事务执行过程中，其他app发来的消息。

消息队列里缓存的 消息的数据结构， 可以是 ( C 伪代码描述)

class TMSGObj {

int ConnectID ; //或者 int Socket

Sting TextMsg;

}

也就是, 缓存的消息里应该包括 哪个连接 (发送过来的)， 是什么文本信息。 通过该连接，可以返回app 所获得的replika 的返回的对话信息。

1. server 端，需要维护一个 网络连接列表 ， 可以定义 为 ConnectList[] , 这个列表，最大为100的长度吧，当然，目前也可以实现为10个的长度。

接入的连接，存放在该 网络裂解列表中， 关闭或中断了的连接，需要从该表中清除。