**协议**

请大家务必使用宏变量，如果你写的是数字的话，后期一更改你就需要把所有数字重新改一遍。相对于1.0版本协议的区别已标黄

// 文件名为protocol.h

/////////////////////主控机向从控机发送的信息/////////////////////////

#define LOG\_IN\_SUCC -1

// 主控机确定从控机的登陆请求合法

// int ret = LOG\_IN\_SUCC

// bool is\_heat\_mode

// int temp

​

#define LOG\_IN\_FAIL -2

// 主控机确定从控机的登陆请求不合法

// int ret = LOG\_IN\_FAIL

​

#define REPLY\_MONEY -3

// 主控机收到从控机发送的状态后，返回房间计费情况

// int ret = REPLY\_MONEY

// double cost --- 实时累计金额

// double power --- 实时累计功率

​

#define REPLY\_CON -4;

// 主控机回复从控机是否送风

// int ret = REPLY\_CON

// bool is\_vaild

​

///////////////////////从控机向主控机发送的信息///////////////////////

#define LOG\_IN\_USER 1

// 从控机输入房间号和身份证号，让主控机验证是否合法

// int op = LOG\_IN\_USER

// int room\_id

// string user\_id

#define REQ\_STOP 2

// 从控机请求停止送风

// int op = REQ\_STOP

​

#define REQ\_RESUME 3

// 从控机请求继续送风

// int op = REQ\_RESUME

​

#define REQ\_UPDATE 4

// 从控机更改设置：工作模式、设定温度、风速大小

// int op = REQ\_UPDATE

// bool is\_heat\_mode

// int temp

// int speed = 1..3

#define REPORT\_STATE 5

// 从控机周期性回复状态，用于主控机监测房间状态

// int op = REPORT\_STATE

// int set\_temp --- the temperature user wants to get

// int real\_temp --- the actual temperature in user's room

// int speed = 1..3

​

**JSON**

半结构化自解释类型，可以看作成一个struct。如果我们现在想通过socket传递一个自定义的message给其他主机，而socket只能接受字符串，这时候就需要json的帮忙

// 随便举个例子，就比如准备登陆，发送帐号密码过去

// 就按照协议里的LOG\_IN\_USER格式

​

#include <protocol.h>

#include <json.hpp>

​

typedef nlohomann::json json // 为了方便

using json = nlohomann::json // C++11还可以这么写

​

string username = "zhangbokang";

​

// 先再本地构建一个json对象

json message = {

{"op", LOG\_IN\_USER},

{"username", username},

{"pwd", "xxxxxx"}

}

​

// 我们这有个假socket对象

SOCKET socket;

// 发送信息给远程主机，send函数接受一个字符串类型

// json.dump()的返回值是string类型，若send接受char\*，请使用string类型自带的c\_str()函数

// socket.send(json.dump().c\_str())

socket.send(json.dump());

​

-------

-------

-------

-------

-------

-------

// 这是远程主机端

// 又有个假socket对象，假设已经都连接好了

SOCKET socket；

// recv返回一个字符串类型，若是char\*，需要你转换成string类型

char\* rec = socket.recv();

string recv = rec; // 这样就可以，string重载了运算符，会帮你自动转的

// 生成json对象

json message = json::parse(recv);

​

// 现在就可以用这个json对象来获取各字段的信息了

int op = message["op"].get<int>();

if (op == LOG\_IN\_USER) {

string username = message["username"].get<string>();

string password = message["password"].get<string>();

.....

}

​

当然这只是最基本用法，还支持嵌套以及类STL容器行为

详见<https://github.com/nlohmann/json>