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。

```
// 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 default

#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_CON -3
// 主控机回复从控机是否送风（根据 REPORT_STATE 发过来的从控机状态）
// int ret = REPLY_CON
// bool is_vaild ---是否送风
// double cost --- 实时累计金额
// double power --- 实时累计功率
// int frequence --- 刷新频率 ms 为单位

//////////////////////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 REPORT_STATE 3
// 从控机周期性回复状态，用于主控机监测房间状态
// 只在开机状态下发送，关机需要发送 isOn = false 的消息，告知主控机自己关机
// int op = REPORT_STATE
// bool is_on ---是否开机
// bool is_heat_mode  ---工作模式
// 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