系统组件使用说明

# Communicator通讯模块使用

模块使用VS2017 MSVC 64位编译器编译,对外接口及功能使用纯C++编写,内部功能和调试使用Qt5.12.1库

## 文件拷贝与集成

拷贝路径:

Fantomas\bin\

1,Fantomas\bin\third\下为头文件,放入待集成程序的头文件引用目录

2,Fantomas\bin中的lib\*\*\*.dll lib\*\*\*.lib放入待集成的程序同级路径下

客户端程序集成:

1,代码中引用模块头文件,举例:

#include "third/communication/communicator/communicator.h"

2,派生Communicator类

使用方法参考Fantomas\testcommdll项目

## 配置

统一使用config\netserver\_config.xml中文件内容.参数固定,使用者只需修改ipAddr,port为目标服务器地址,端口,心跳参数不修改.

### 启动

int Start();

### 停止

int Stop();

### 解析数据

重载解析接口,模块收到数据帧后会自动调用重载函数

virtual int DealFrame(int type, int length, unsigned char \*value) = 0;

### 发送数据

直接调用

virtual int SendFrame(int type, int length, unsigned char \*value);

### 通讯模块消息

重载接口,模块连接成功或失败后会调用该接口

virtual int MessageFrame(int type, int length, unsigned char \*value) = 0;

用到的结构体为

struct MessageCode

{

int code; //通讯模块信息代码

};

code类型目前为:

#define COMM\_Code\_Connect\_Success 100 //连接成功

#define COMM\_Code\_Connect\_Break 102 //连接断开

## 使用要求

DealFrame接口耗时应小于通讯读通道线程轮询间隔,一般应小于5ms

## 测试

目前模块内部托管心跳机制,调用方不处理断线流程

配置文件中IP地址改为192.169.100.8,端口改为测试端口60001

# commsocket组件

## 独立使用方法

预编译声明中添加

INDEPENDENT\_MODULE\_COMM\_COMMSOCKET

将commsocket文件夹拷贝至工程第三方库的路径下,将libcommsocket.dll/lib放在可执行文件路径,代码中引用commsocket/commsocket.h文件,使用下方说明的接口即可.

## 客户端使用流程

接口具体参数参考“接口说明”一节

windows下准备启用通讯业务时,执行:

CommBase::SocketStartup

tcp方式连接服务器时调用:

CommTcpSocket::ConnectChannel

tcp方式向socket写入时调用:

CommTcpSocket::Write

tcp方式从socket读取时调用:

CommTcpSocket::Read

关闭socket时调用:

CommBase::CloseSocket

获取socket错误码时调用:

CommBase::GetLastError

停止socket业务时调用:

CommBase::SocketCleanup

## 接口说明

具体说明在头文件中可以看到.

### 基础接口

//初始化socket,程序启动时用(win)

static void SocketStartup();

//清理socket,程序退出时用(win)

static void SocketCleanup();

//关闭socket

//sock:socket id retry:重复关闭次数

static int CloseSocket(int sock, int retry = 5);

//获取上一个socket错误码

//sockFlag:0-调用全局GetLastError 非0-调用WSAGetLastError

static int GetLastError(int sockFlag = 0);

### tcp接口

#### tcp客户端

//建立tcp连接(参数: hostname:目标IP地址 port:目标端口 noblock\_flag:阻塞模式 返回值:socket id)

//noblock\_flag: 1:noblock 0:block

static int ConnectChannel(char \*hostname, int port, unsigned long noblock\_flag = TRUE);

//读取socket(返回值:读取字节数)

//retry:单次读取产生允许异常后的重复次数

static int Read(int sock,char \*buf,int size,int retry = 0);

//写入socket(返回值:写入字节数)

//retry:单次写入产生允许异常后的重复次数

static int Write(int sock,char \*buf,int size,int retry = 2);

#### tcp服务器

暂无

### udp接口

暂无说明

# 调试

未完成