// router\_s.cpp: 定义控制台应用程序的入口点。

//

#include "stdafx.h"

#include "zmq.h"

#include "zmq\_utils.h"

#include "zhelper.h"

#include <fstream>

#include "json\json.h"

#include "json\config.h"

void \*frontend = NULL;

int main()

{

void \*ctx = zmq\_ctx\_new();

frontend = zmq\_socket(ctx, ZMQ\_ROUTER);

zmq\_bind(frontend, "tcp://192.168.1.102:5556");

int count = 0;

Json::Value value;

Json::Reader reader;

std::ifstream in;

in.open("face\_detect\_send\_data.json", std::ios::binary);

if (!in.is\_open())

return -1;

if (reader.parse(in, value))

{

printf("serip ==== %s \n", value["msgHead"]["serverIP"].toStyledString().data());

}

while (count++<50) {

char identity[100] = { 0 };

char id[100] = { 0 };

int ret = zmq\_recv(frontend, identity, 100, 0);

if (ret != -1) {

printf("identity ==== %s\n", identity);

strcpy(id, identity);

}

memset(identity, 0, 100);

ret = zmq\_recv(frontend, identity, 100, 0);

if (ret != -1) {

printf("empty ==== %s\n", identity);

}

zmq\_msg\_t msg\_recv;

zmq\_msg\_init(&msg\_recv);

ret = zmq\_msg\_recv(&msg\_recv, frontend, 0);

if (ret != -1) {

printf("content ==== %s\n", (char \*)zmq\_msg\_data(&msg\_recv));

}

zmq\_send(frontend, id, strlen(id), ZMQ\_SNDMORE);

zmq\_send(frontend, "", 0, ZMQ\_SNDMORE);

zmq\_send(frontend, "main hello", 10, 0);

while (1) {

zmq\_send(frontend, id, strlen(id), ZMQ\_SNDMORE);

zmq\_send(frontend, "", 0, ZMQ\_SNDMORE);

zmq\_send(frontend, (char \*)value.toStyledString().data(), strlen((char \*)value.toStyledString().data()), 0);

Sleep(1000);

////

int ret = zmq\_recv(frontend, identity, 100, 0);

if (ret != -1) {

printf("identity ==== %s\n", identity);

strcpy(id, identity);

}

memset(identity, 0, 100);

ret = zmq\_recv(frontend, identity, 100, 0);

if (ret != -1) {

printf("empty ==== %s\n", identity);

}

zmq\_msg\_t msg\_recv;

zmq\_msg\_init(&msg\_recv);

ret = zmq\_msg\_recv(&msg\_recv, frontend, 0);

if (ret != -1) {

printf("content ==== %s\n", (char \*)zmq\_msg\_data(&msg\_recv));

}

///

}

}

zmq\_close(frontend);

return 0;

}