20220415-socket

1.过程描述

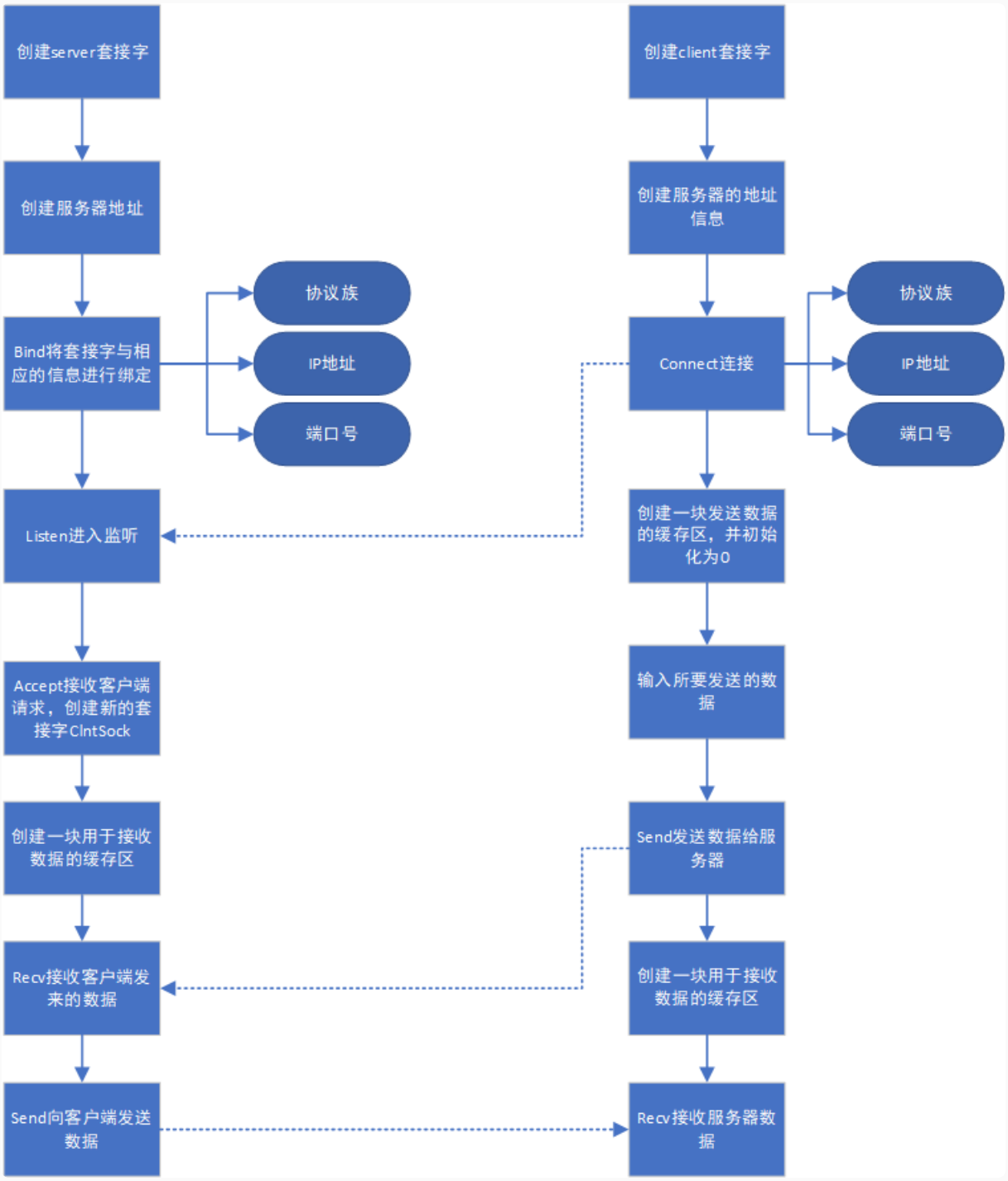
char{0} C++ 复制代码

```
char demo[100]={0}
表示将所有元素初始化为0
```

INADDR_ANY C++ 复制代码

```
//way one
sockAddr.sin_addr.s_addr = inet_addr("127.0.0.1");
//way two
servAddr.sin_addr.s_addr = htonl(INADDR_ANY);
//将sin_addr设置为INADDR_ANY",转换过来就是0.0.0.0，泛指本机的意思，也就是表示本机的所有IP，因为有些机子不止一块网卡，多网卡的情况下，这个就表示所有网卡ip地址的意思。
```

服务器与客户端交互逻辑


创建server套接字
创建服务器地址
Bind将套接字与相应的信息进行绑定
协议族
IP地址
端口号
Listen进入监听
Accept接收客户端请求，创建新的套接字ClntSock
创建一块用于接收数据的缓存区
Recv接收客户端发来的数据
Send向客户端发送数据
创建client套接字
创建服务器的地址信息
Connect连接
协议族
IP地址
端口号
创建一块发送数据的缓存区，并初始化为0
输入所要发送的数据
Send发送数据给服务器
创建一块用于接收数据的缓存区
Recv接收服务器数据

回声客户端Server.cpp C++ 复制代码

```cpp
#include <stdio.h>
#include <WinSock2.h>
#pragma comment (lib,"ws2_32.lib")

#define BUF_SIZE 100

int main()
{
    WSADATA wsaData;
    WSAStartup(MAKEWORD(2, 2), &wsaData);

    SOCKET servSock = socket(AF_INET, SOCK_STREAM, 0);

    SOCKADDR_IN sockAddr;
    memset(&sockAddr, 0, sizeof(sockAddr));
    sockAddr.sin_family = PF_INET;
    sockAddr.sin_addr.s_addr = inet_addr("127.0.0.1");
    sockAddr.sin_port = htons(1234);
    bind(servSock, (SOCKADDR*)&sockAddr, sizeof(SOCKADDR));

    listen(servSock, 20);

    SOCKADDR clntAddr;
    int nSize = sizeof(SOCKADDR);
    SOCKET clntSock = accept(servSock, (SOCKADDR*)&clntAddr, &nSize);
    char buffer[BUF_SIZE];//缓冲区
    int strLen = recv(clntSock, buffer,BUF_SIZE, 0);//接收客户端发来的数据
    send(clntSock, buffer, strLen, 0);//将数据原样返回

    closesocket(clntSock);
    closesocket(servSock);

    WSACleanup();
    return 0;
}

```

回声客户端Client.cpp C++ 复制代码

```cpp
#include <stdio.h>
#include <stdlib.h>
#include <WinSock2.h>
#pragma comment (lib,"ws2_32.lib")

#define BUF_SIZE 100

int main()
{
    WSADATA wsaData;
    WSAStartup(MAKEWORD(2, 2), &wsaData);

    SOCKET sock = socket(AF_INET, SOCK_STREAM, 0);

    SOCKADDR_IN sockAddr;
    memset(&sockAddr, 0, sizeof(sockAddr));
    sockAddr.sin_family = PF_INET;
    sockAddr.sin_addr.s_addr = inet_addr("127.0.0.1");
    sockAddr.sin_port = htons(1234);
    connect(sock, (SOCKADDR*)&sockAddr, sizeof(SOCKADDR));

    char bufSend[BUF_SIZE] = { 0 };
    printf("Input a string: ");
    gets_s(bufSend);//用scanf("%s",bufsend)的话一旦遇到空格就会自动忽略之后的内容
    send(sock, bufSend, strlen(bufSend), 0);

    char bufRecv[BUF_SIZE] = { 0 };
    recv(sock, bufRecv, BUF_SIZE, 0);

    printf("Message from server:%s\n", bufRecv);

    closesocket(sock);

    WSACleanup();

    system("pause");
    return 0;
}
```

多人聊天室Server.cpp C++ 复制代码

```cpp
#include <WinSock2.h>
#include <iostream>
#pragma comment(lib,"ws2_32.lib")
#define MAX_BUF 100
using namespace std;

DWORD WINAPI ThreadFun(LPVOID lpThreadParameter);

int main()
{
    WSADATA wsaData;
    if (WSAStartup(MAKEWORD(2, 2), &wsaData) != 0)
    {
        cout << "WSAStartup Error: " << WSAGetLastError() << endl;
    }

    //创建流式套接字
    SOCKET s = socket(AF_INET, SOCK_STREAM, IPPROTO_TCP);
    if (s == INVALID_SOCKET)
    {
        cout << "Socket error: " << WSAGetLastError() << endl;
    }

    //绑定端口和ip
    SOCKADDR_IN addr;
    memset(&addr, 0, sizeof(SOCKADDR_IN));
    addr.sin_family = PF_INET;
    addr.sin_port = htons(8000);
    addr.sin_addr.s_addr = inet_addr("127.0.0.1");

    if (bind(s, (SOCKADDR*)&addr, sizeof(SOCKADDR)) == SOCKET_ERROR)
    {
        cout << "Bind Error: " << WSAGetLastError() << endl;
    }

    //监听
    listen(s, 20);

    //主线程循环接受客户端的连接
    while (1)
    {
        SOCKADDR_IN caddr;
        int len = sizeof(SOCKADDR);
        SOCKET c = accept(s, (SOCKADDR*)&caddr,&len);
        if (c != INVALID_SOCKET)
```

```
        {
            //创建线程，并且传入与客户端通讯的套接字
            HANDLE hThread = CreateThread(NULL, 0, ThreadFun,
(LPVOID)c,0,NULL);
            CloseHandle(hThread);//关闭对线程的引用
        }
    }
    //关闭套接字
    closesocket(s);
    WSACleanup();
    return 0;
}

DWORD WINAPI ThreadFun(LPVOID lpThreadParameter)
{
    //与客户端通讯，发送或者接收数据
    SOCKET c = (SOCKET)lpThreadParameter;
    cout << "欢迎 " << c << " 进入聊天室!" << endl;

    //发送数据
    char buf[MAX_BUF] = { 0 };
    sprintf(buf, "欢迎%d进入聊天室!",c);
    send(c, buf, MAX_BUF, 0);

    //循环接收客户端数据
    int ret = 0;
    do
    {
        char buf2[MAX_BUF] = { 0 };
        ret = recv(c, buf2, MAX_BUF, 0);
        cout << c << " 说: " << buf2 << endl;
    } while (ret != SOCKET_ERROR && ret != 0);

    cout << c << "离开了聊天室";
    return 0;
}
```

多人聊天室Client.cpp C++ 复制代码

```cpp
#include <WinSock2.h>
#include <iostream>
#pragma comment(lib,"ws2_32.lib")
#define BUF_SIZE 100
using namespace std;

int main()
{
    WSADATA wsaData;
    if (WSAStartup(MAKEWORD(2, 2), &wsaData) != 0)
    {
        cout << "WSAStartup error: "<<GetLastError() << endl;
        return 0;
    }

    //创建流式套接字
    SOCKET s = socket(AF_INET, SOCK_STREAM, IPPROTO_TCP);
    if (s == INVALID_SOCKET)
    {
        cout << "socket error: " << GetLastError() << endl;
        return 0;
    }

    //链接服务器
    SOCKADDR_IN addr;
    addr.sin_family = PF_INET;
    addr.sin_addr.s_addr = inet_addr("127.0.0.1");
    addr.sin_port = htons(8000);
    if (connect(s, (SOCKADDR*)&addr, sizeof(SOCKADDR)) == SOCKET_ERROR)
    {
        cout << "connect error: " << GetLastError() << endl;
        return 0;
    }
    //接收服务器的消息
    char buf[BUF_SIZE] = { 0 };
    recv(s, buf, BUF_SIZE, 0);
    cout << buf << endl;

    //随时给服务端发消息
    int ret = 0;
    do
    {
        char buf[BUF_SIZE] = { 0 };
        cout << "请输入聊天内容：";
        gets_s(buf);
```

```
46              ret = send(s, buf, 100, 0);
47          } while (ret != SOCKET_ERROR && ret != 0);
48
49          closesocket(s);
50          WSACleanup();
51          return 0;
52      }
```

2.结果输出

今天的活动跟昨天差不多，主要实现了一个回声客户端跟一个多人聊天室，其中聊天室里面由于涉及到线程没有完全搞明白，此外服务器跟客户端之间具体参数的传递机制也没太搞懂。后面两天继续socket编程，之后将开始一个C++项目。