Introduction to Computer Networks

Socket Tutorial

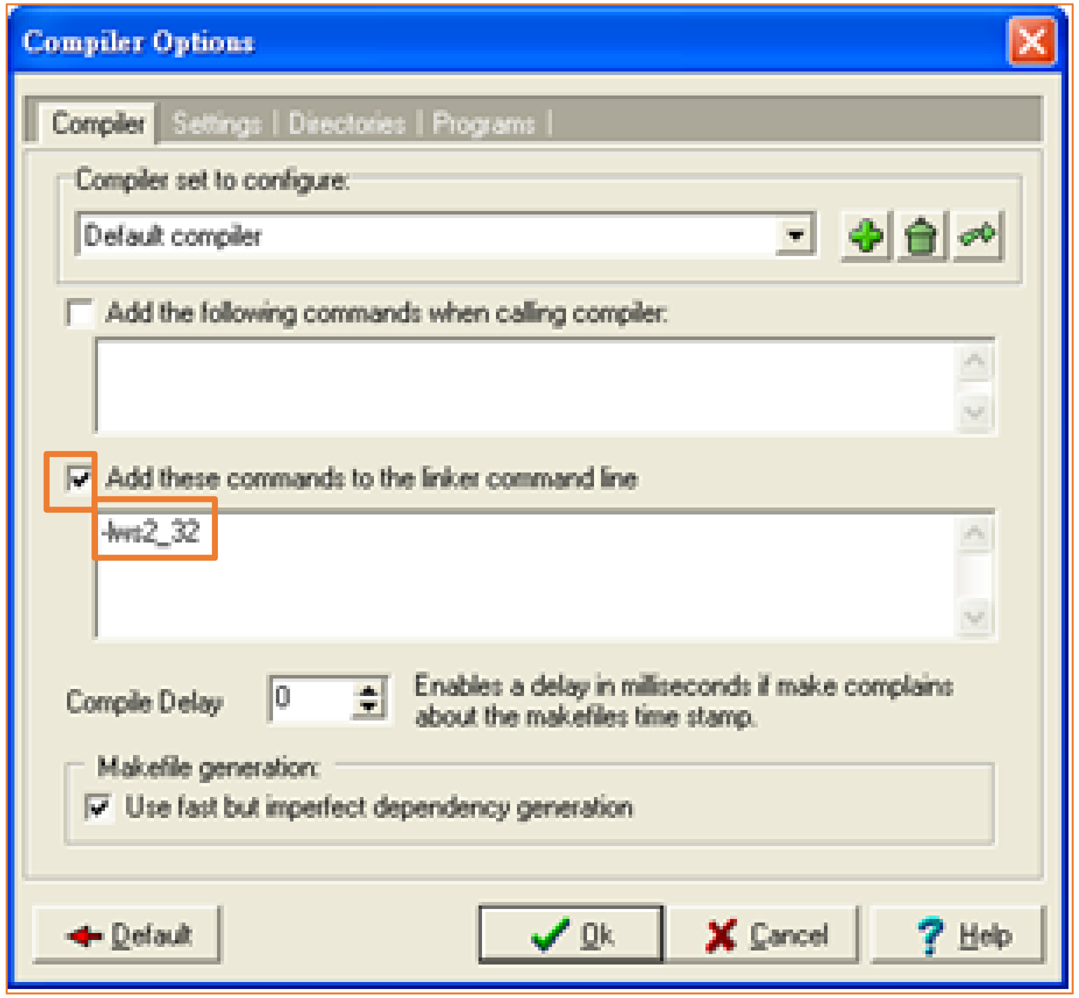
Outline

Dev C++ environment setting

- Download Dev-C++ from http://www.bloodshed.net/devcpp.html
- Orwell Dev-C++ 5.11 (Recommended)
- http://orwelldevcpp.blogspot.tw/2015/04/dev-c-511-released.html
- To compile the source file or project see p4
- Alternative option to compile the project, see p5

Dev C++: To build the source file or project

- Tools → Compiler Options → Add “-lws2_32” to the linker command line
- Remember to include winsock2.h



Compiler Options

Compiler | Settings | Directories | Programs

Compiler set to configure:

Default compiler

[ ] Add the following commands when calling compiler:

[✓] Add these commands to the linker command line

```
-lws2_32
```

Compile Delay 0 Enables a delay in milliseconds if make complains about the makefiles time stamp.

Makefile generation:

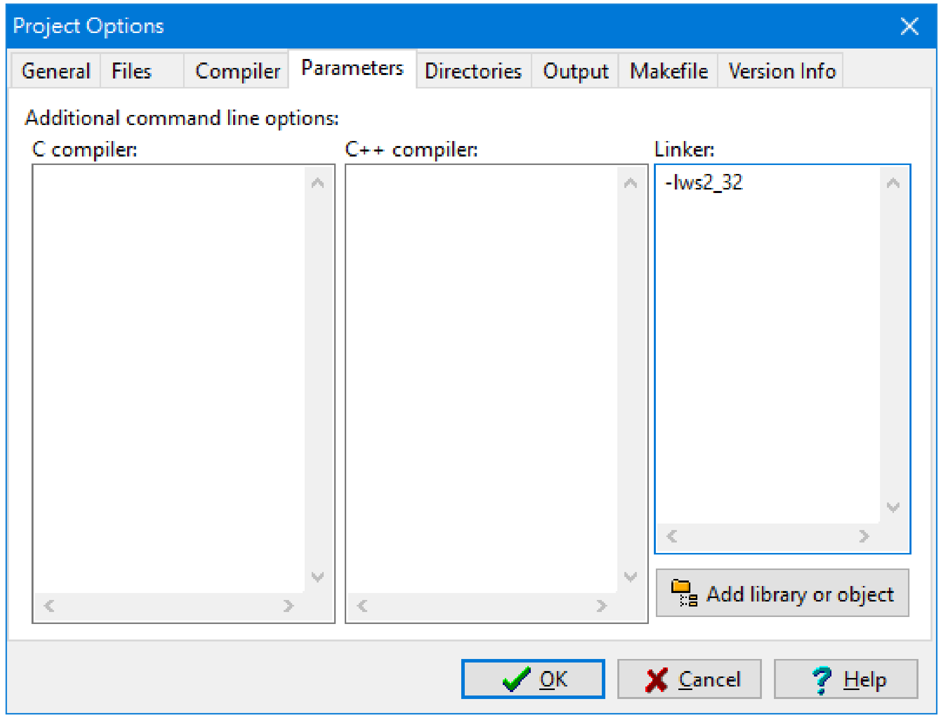
[✓] Use fast but imperfect dependency generation

Dev C++:
Alternative option to compile the project

- File → New → Project
- Choose “Console Application”
- Project → Project Options → Parameters
- Add the command line options

 “-lws2_32” at Linker area

- Remember to include winsock2.h



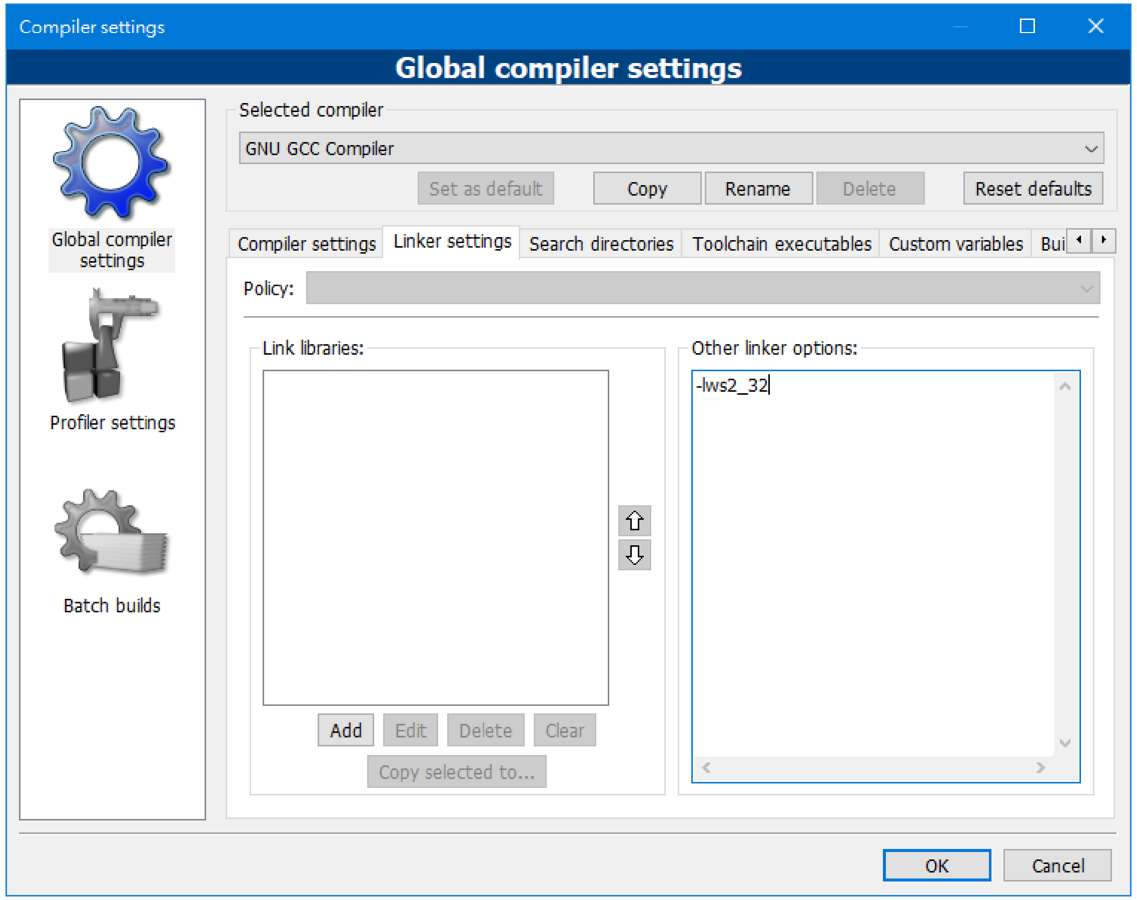
-lws2_32

CodeBlocks environment setting

- Download CodeBlocks from http://www.codeblocks.org/downloads
- To compile the source file or project, see p7
- Alternative option to compile the project, see p8

CodeBlocks : To compile the source file or project

- Settings → Compiler and debugger → Linker Settings
- Add "-lws2_32" to Other linker options
- Remember to include winsock2.h

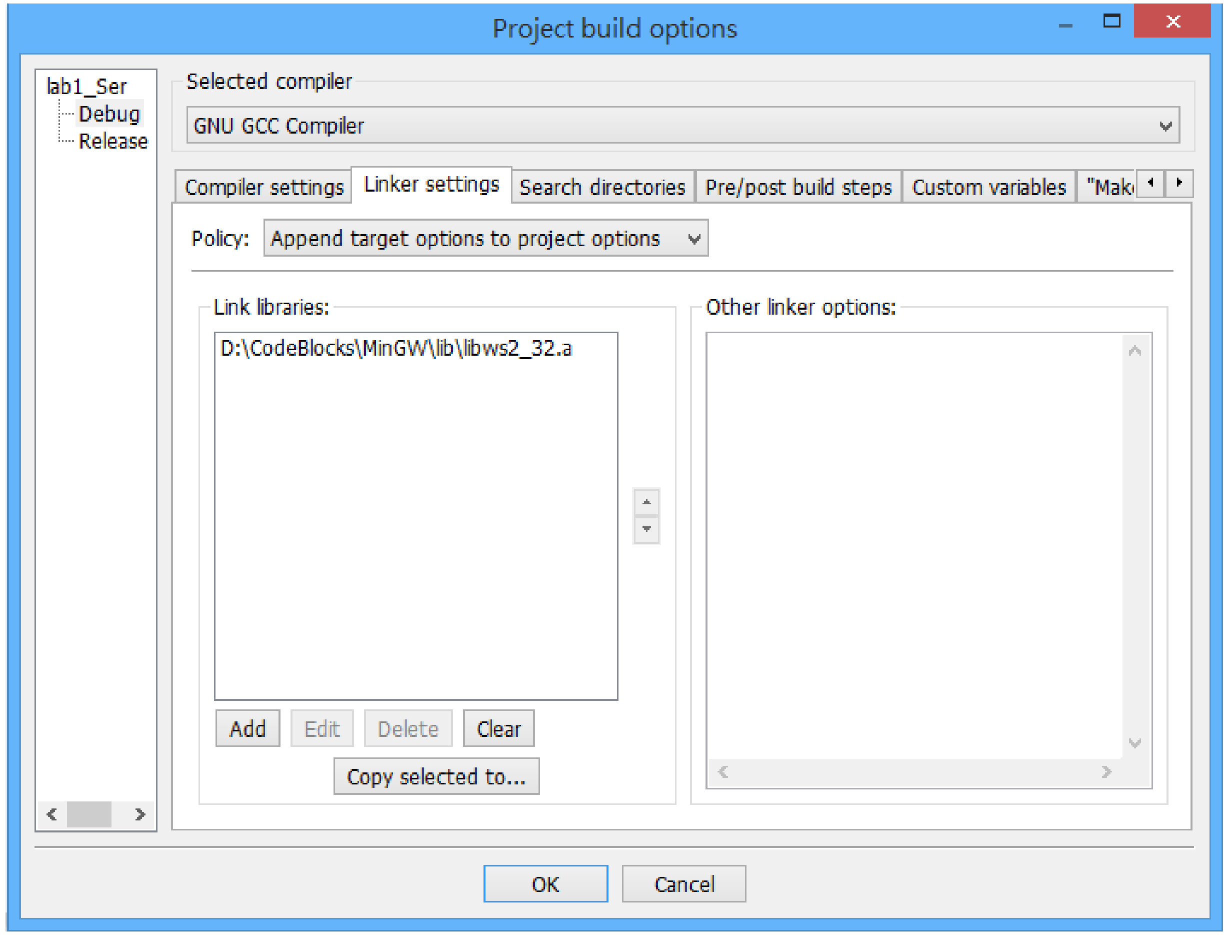


Other linker options:

```
-lws2_32
```

CodeBlocks : Alternative option to compile the project

- File → New → Project
- Right click project → Build Option→ Linker Settings
- Choose C:\Program Files (x86)\CodeBlocks\MinGW\lib ws2_32.a
- Remember to include winsock2.h



Project build options

lab1_Ser
Debug
Release

Selected compiler: GNU GCC Compiler

Compiler settings | Linker settings | Search directories | Pre/post build steps | Custom variables | "Mak

Policy: Append target options to project options

Link libraries:
D:\CodeBlocks\MinGW\lib\libws2_32.a

Other linker options:

Use command line to compile

- gcc -o cli TCP_echo_client_win.c -lws2_32
- gcc -o ser TCP_echo_server_win.c -lws2_32

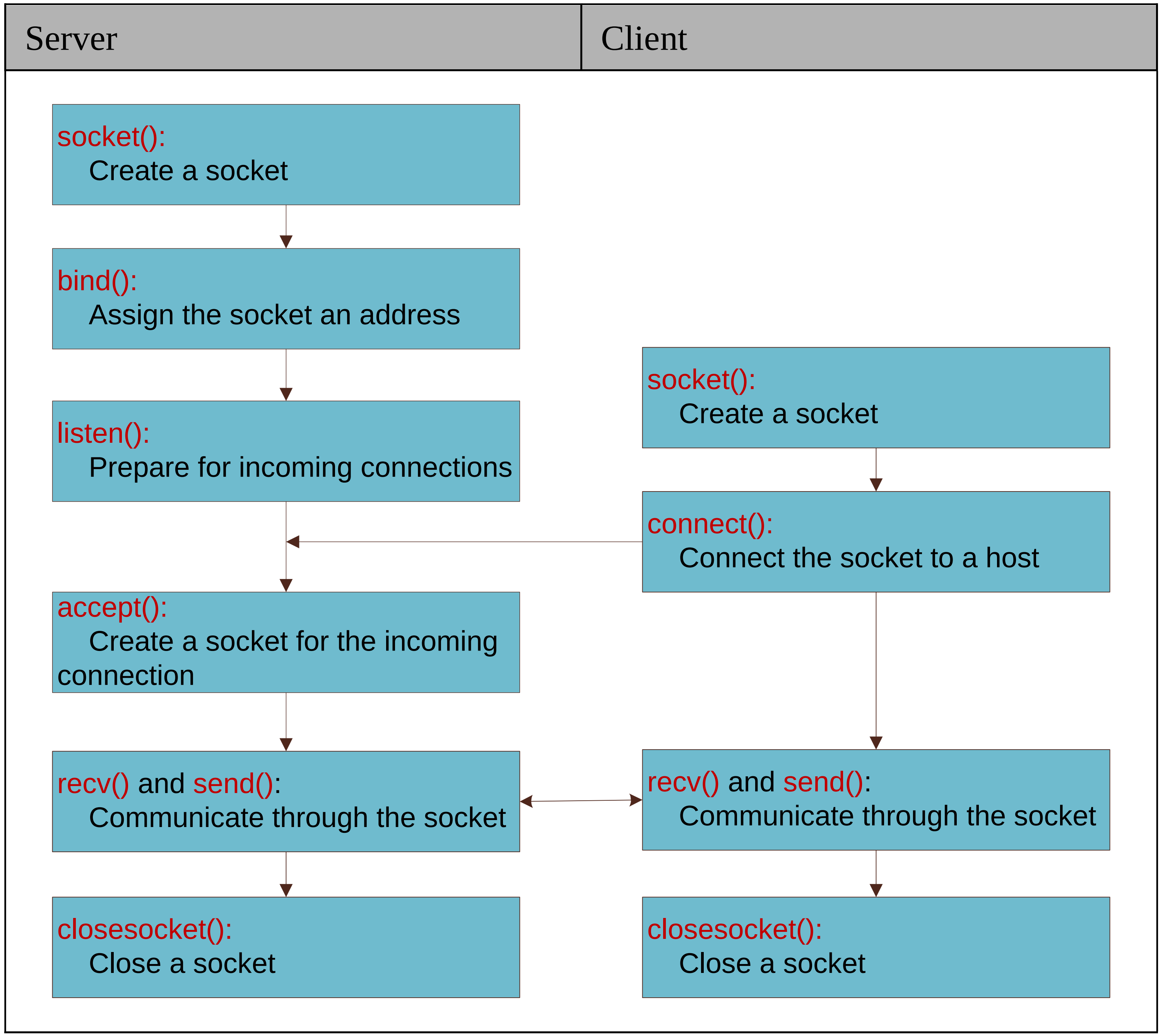
Linux/Unix system

- Include <sys/socket.h>, <sys/types.h>, <netinet/in.h> and <arpa/inet.h> in your program
- Compile with:
  - gcc -o cli TCP_echo_client_linux.c
  - gcc -o ser TCP_echo_server_linux.c

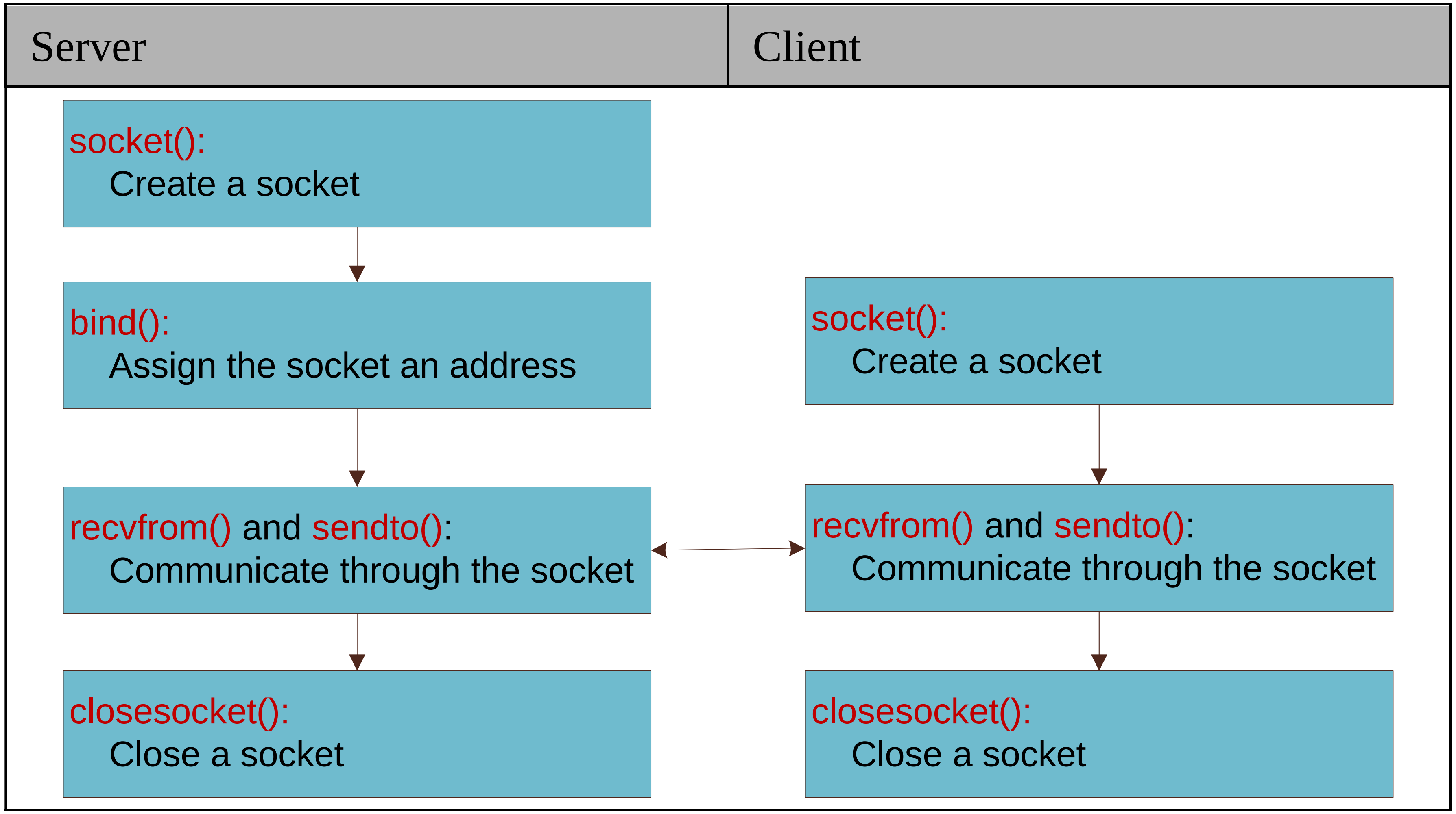
Socket programming

- TCP flow chart
- UDP flow chart
- Data structure of address
- Functions

TCP flow chart


Server
Client
socket(): Create a socket
bind(): Assign the socket an address
listen(): Prepare for incoming connections
accept(): Create a socket for the incoming connection
recv() and send(): Communicate through the socket
closesocket(): Close a socket
socket(): Create a socket
connect(): Connect the socket to a host
recv() and send(): Communicate through the socket
closesocket(): Close a socket

UDP flow chart



Data structure of address

Structure	Usage

struct sockaddr_in {
    short sin_family;
    unsigned short sin_port;
    struct in_addr sin_addr;
    char sin_zero[8];
};
	sin_family = AF_INET; (Address Family)
	sin_port = htons(80);
	sin_addr=inet_addr(“127.0.0.1”); (for client)
	sin_addr.s_addr = INADDR_ANY; (for server)

The **htons** function converts a **u_short** from host to TCP/IP network byte order (which is big-endian).

Functions

- **int WSAStartup(WORD wVersionRequested, LPWSADATA lpWSAData)**

  在使用winsock前需先呼叫本函式

  參數:

    wVersionRequested : DLL版本

      MAKEWORD(1, 2):版本2.1

      MAKEWORD(2, 2):版本2.2

    lpWSAData: WSADATA結構

  回傳值:成功會回傳0, 否則回傳錯誤代碼

- **int WSACleanup();**

  結束使用winsock時呼叫

  回傳值:成功會回傳0, 否則回傳錯誤代碼

- **int socket(int af, int type, int protocol)**

  參數:
  af ：位址資料族系(family)，用不同方式表示網路位址。
  type：通訊方式
    SOCKET_STREAM：代表TCP
    SOCKET_DGRAM：代表UDP
  Protocal：傳輸協定編號 選擇IPPROTO_TCP (TCP通訊協定) 或寫入0，交由系統設定
  回傳值：-1表示建立socket發生錯誤 若成功則回傳非負整數，稱為socket descriptor(sockdes)

- **int bind(int sockdes, const struct sockaddr *addr, socklen_t addrlen)**

  bind()是把設定的address綁在Socket身上
  參數：
  sockdes : 指定好通訊協定的socket
  addr : 指定本地端位址，資料格式為sockaddr
  addrlen : addr之資料長度(單位byte)
  回傳值：-1表錯誤，否則為0

- **int listen(int sockdes, int backlog)**

  等待請求
  參數：
  sockdes：設定好bind(),並且尚未連線的socket
  backlog：等待Server接受連線前，同時最大連線數
  回傳值：-1表錯誤，否則為0

- **int accept(int sockdes, struct sockaddr *addr, socklen_t *addrlen)**

  接受請求
  參數：
  sockdes：一個設定為listen狀態的socket
  addr：Client端位址資訊
  addrlen：addr長度
  回傳值：-1表示錯誤，否則傳回另一個包含Client端資訊的新socket descriptor，作為傳送資料用

- **int connect(int sockdes, const struct sockaddr *addr, socklen_t addrlen)**
  建立連線
  設定方式請參照bind()函式
  回傳值：-1表錯誤，否則回傳0

- **ssize_t recv(int sockdes, void *buf, size_t len, int flags)**
  參數：
  sockdes：一個建立連線成功的socket
  buf：呼叫recv，用來儲存收到資料的暫存器
  len：buf的長度(byte)
  flags：選擇工作模式，一般填入0
  回傳值：-1表錯誤，否則傳回接受到資料的長度(byte)

- **ssize_t send(int sockdes, const void *buf, size_t len, int flags)**
  參數：
  sockdes：一個建立連線成功的socket
  buf：用來儲存將送出資料的暫存器
  len：buf的長度(byte)
  flags：選擇工作模式，一般填入0
  回傳值：-1表錯誤，否則傳回送出資料的長度(byte)

- **int closesocket(int sockdes)**
  關閉socket

- **int recvfrom(int sockdes, char *buf, int len, int flags, sockaddr *from, int *fromlen)**

  參數：
  sockdes：一個建立連線成功的socket
  buf：呼叫recv，用來儲存收到資料的暫存器
  len：buf的長度(byte)
  flags：選擇工作模式，一般填入0
  from：連接對象的位址資訊
  fromlen：from長度
  回傳值：-1表示錯誤，否則傳回接收到的字節數

- **int sendto(int sockdes, const char *buf, int len, int flags, const sockaddr *to, int tolen)**

  參數：
  sockdes：一個建立連線成功的socket
  buf：呼叫recv，用來儲存收到資料的暫存器
  len：buf的長度(byte)
  flags：選擇工作模式，一般填入0
  to：連接對象的位址資訊
  tolen：to長度
  回傳值：-1表示錯誤，否則傳回發送的字節數

Examples

- TCP echo server/client