Introdução às Redes de Comunicação
2014/15

#include <winsock.h> / #include <winsock2.h>

SOCKET **socket**(int af, int type, int protocol); /* PF_INET, SOCK_DGRAM, IPPROTO_UDP, SOCK_STREAM, IPPROTO_TCP*/

struct sockaddr_in a; /* a.**sin_family**, a.sin_addr.**s_addr**, a.**sin_port** */

…**htons**(…);

…**htonl**(…);

…**ntohs**(…);

…**ntohl**(…);

unsigned long **inet_addr**(const char *cp);

INADDR_NONE

struct hostent* **gethostbyname**(char *name);

```
/*
** struct hostent info;
** struct sockaddr_in addr;
** ...
**  memcpy(&(addr.sin_addr.s_addr), info->h_addr, info->h_length);
**
*/
```

char* **inet_ntoa**(struct in_addr in);

int **bind**(SOCKET s, const struct sockaddr *name, int namelen);

int **connect**(SOCKET s, const struct sockaddr *name, int namelen);

SOCKET **accept**(SOCKET s, struct sockaddr *from, int *fromlen);

int **send**(SOCKET s, const char *buf, int len, int flags);

int **recv**(SOCKET s, char *buf, int len, int flags);

int **sendto**(SOCKET s, const char *buf, int len, int flags, struct sockaddr *to, int tolen);

int **recvfrom**(SOCKET s, char *buf, int len, int flags, struct sockaddr *from, int *fromlen);

int **setsockopt**(SOCKET s, int level, int optname, const char *optval, int optlen);

```
/*
**  level = SOL_SOCKET, optname = SO_RCVTIMEO, optval = (char *)&timeoutMsec (DWORD
**  timeoutMsec;)
*/
```

int **getsockname**(SOCKET s, struct sockaddr *name, int *namelen);

int **getpeername**(SOCKET s, struct sockaddr *name, int *namelen);

```
int WSAGetLastError(void); /* WSAETIMEDOUT */

SOCKET_ERROR

INVALID_SOCKET

int select(32, fd_set *readfds, NULL, NULL, struct timeval *timeout);

FD_ZERO(&set);

FD_SET(s, &set);

FD_ISSET(s, &set);

struct timeval {
  long tv_sec;
  long tv_usec;
}

int strcmp(const char *s1, const char *s2);

char * strcpy_s(char * strDestination, int sizeStrDestination, const char * strSource);

int closesocket(SOCKET s);

HANDLE WINAPI CreateThread(
  _In_opt_   LPSECURITY_ATTRIBUTES lpThreadAttributes,
  _In_       SIZE_T dwStackSize,
  _In_       LPTHREAD_START_ROUTINE lpStartAddress,
  _In_opt_   LPVOID lpParameter,
  _In_       DWORD dwCreationFlags,
  _Out_opt_  LPDWORD lpThreadId
);

/*
** void AtendeCliente(LPVOID param)
** ...
** SECURITY_ATTRIBUTES sa;
** DWORD thread_id;
** ...
** sa.nLength=sizeof(sa);
** sa.lpSecurityDescriptor=NULL;
** ...
** CreateThread(&sa, 0, (LPTHREAD_START_ROUTINE)AtendeCliente, (LPVOID)param,
** (DWORD)0, &thread_id);
*/
```