28. Developing a client that contacts a given DNS server to resolve a given hostname in java/C.

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#ifdef \_WIN32

#include <winsock2.h>

#include <ws2tcpip.h>

#pragma comment(lib, "ws2\_32.lib")

#else

#include <sys/types.h>

#include <sys/socket.h>

#include <netdb.h>

#include <unistd.h>

#endif

void resolve\_hostname(const char \*hostname) {

#ifdef \_WIN32

WSADATA wsa;

if (WSAStartup(MAKEWORD(2, 2), &wsa) != 0) {

printf("WSAStartup failed\n");

return;

}

#endif

struct addrinfo hints, \*res, \*p;

char ipstr[INET6\_ADDRSTRLEN];

memset(&hints, 0, sizeof hints);

hints.ai\_family = AF\_UNSPEC; // Support both IPv4 & IPv6

hints.ai\_socktype = SOCK\_STREAM;

if (getaddrinfo(hostname, NULL, &hints, &res) != 0) {

printf("Could not resolve hostname: %s\n", hostname);

#ifdef \_WIN32

WSACleanup();

#endif

return;

}

printf("IP addresses for %s:\n", hostname);

for (p = res; p != NULL; p = p->ai\_next) {

void \*addr;

if (p->ai\_family == AF\_INET) { // IPv4

struct sockaddr\_in \*ipv4 = (struct sockaddr\_in \*)p->ai\_addr;

addr = &(ipv4->sin\_addr);

} else { // IPv6

struct sockaddr\_in6 \*ipv6 = (struct sockaddr\_in6 \*)p->ai\_addr;

addr = &(ipv6->sin6\_addr);

}

inet\_ntop(p->ai\_family, addr, ipstr, sizeof ipstr);

printf(" %s\n", ipstr);

}

freeaddrinfo(res);

#ifdef \_WIN32

WSACleanup();

#endif

}

int main(int argc, char \*argv[]) {

if (argc != 2) {

printf("Usage: %s <hostname>\n", argv[0]);

return 1;

}

resolve\_hostname(argv[1]);

return 0;

}

A screenshot of a computer

AI-generated content may be incorrect.