What are sockets?

OS에 따라 다름

Basic TCP/UDP program flow

TCP, UDP program flow

Server / Client: socket() 전에 getaddrinfo() 해야함

IPv6

- struct sockaddr_in6
- struct in6_addr -> 배열 크기 16

IPv4: AF_INET, 10.12.110.57 IPv6: AF_INET6, 2001:db8:63b3:1::3490

- IP to network inet_aton() / inet_addr() --> inet_pton()
- network to IP inet_ntoa() --> inet_ntop()
- domain name (host) -> IP gethostbyname() --> getaddrinf()

getaddrinfo()

IP 버전 뭘 쓸지 모를 때 사용함

```
getaddrinfo(DNS or IP, http or port, addrinfo *hints, addrinfo **res);
```

struct addrinfo

```
addrinfo{
    int flag; //host 주소 (domain or ip)
    int family; //AF_INET
    int socktype; //SOCK_STREAM
    int protocol; //어떤거든 가능
    size_t addrlen;
    char canonname //별명
    struct addrinfo; //linked list 형태, 하나의 도메인의 여러 IP 존재
}
```

▶ 실습 01: showip.c

```c
/*
** showip.c -- show IP addresses for a host given on the command line
*/

#include <stdio.h>
#include <string.h>
#include <sys/types.h>
```

```c
#include <sys/socket.h>
#include <netdb.h>
#include <arpa/inet.h>
#include <netinet/in.h>

int main(int argc, char *argv[])
{
    struct addrinfo hints, *res, *p;
    int status;
    char ipstr[INET6_ADDRSTRLEN];

    if (argc != 2) {
        fprintf(stderr,"usage: showip hostname\n");
        return 1;
    }

    memset(&hints, 0, sizeof hints);
    hints.ai_family = AF_UNSPEC; // AF_INET or AF_INET6 to force version
    hints.ai_socktype = SOCK_STREAM;

    //hints에 저장된 값으로 server ip를 res에 저장
    if ((status = getaddrinfo(argv[1], NULL, &hints, &res)) != 0) {
        fprintf(stderr, "getaddrinfo: %s\n", gai_strerror(status));
        return 2;
    }

    printf("IP addresses for %s:\n\n", argv[1]);

    for(p = res;p != NULL; p = p->ai_next) {
        void *addr;
        char *ipver;

        // get the pointer to the address itself,
        // different fields in IPv4 and IPv6:
        if (p->ai_family == AF_INET) { // IPv4
            struct sockaddr_in *ipv4 = (struct sockaddr_in *)p->ai_addr;
            addr = &(ipv4->sin_addr);
            ipver = "IPv4";
        } else { // IPv6
            struct sockaddr_in6 *ipv6 = (struct sockaddr_in6 *)p->ai_addr;
            addr = &(ipv6->sin6_addr);
            ipver = "IPv6";
        }

        // convert the IP to a string and print it:
        inet_ntop(p->ai_family, addr, ipstr, sizeof ipstr);
        printf("  %s: %s\n", ipver, ipstr);
    }

    freeaddrinfo(res); // free the linked list

    return 0;
}

```

Colored by Color Scripter

server 입장에서 자신의 IP 알고싶을 때

```
struct addrinfo hints, *servinfo;

memeset(&hints, 0, sizeof hints);
hints.ai_family = AF_UNSPEC; //미정
hints.ai_socktype = SOCK_STREAM; //TCP
hints.ai_flags = AI_PASSIVE; //my IP or Domain?

//3490 - port
if((status = getaddrinfo(NULL, "3490", &hints, &servinfo))!=0){
    error...
}
```

1. hint에 저장된 값으로 해당 서버의 IP를 res에 저장
2. res에서 IP 형식에 맞게 하나씩 출력해나감

client 입장에서 서버와 특정 서버와 connect 하고 싶을 때, IP 찾기

```
struct addrinfo hints, *servinfo;

memeset(&hints, 0, sizeof hints);
hints.ai_family = AF_UNSPEC; //미정
hints.ai_socktype = SOCK_STREAM; //TCP

status = getaddrinfo("www.example.com", "3490", &hints, &servinfo);
```

1. example 서버와 연결하고 싶다면...
2. 하드코딩 하지 않아도 socket(res->ai_family, res->ai_socktype... 으로 바로 할 수 있음) connect도 마찬가지

Simple Time Server

Simple Time Console

```
#include <stdio.h>
#include <time.h>

int main(){
    time_t timer;
    time(&timer);

    printf("Local time is: %s", ctime(&timer));

    return 0;
}
```

--> 현재 시각 출력됨

Simple Time Console + Networking

127.0.0.1:port 에 접근하면 HTTP를 통해 현재 시간 출력해주는 코드

▶ 실습 02: time_server.c

```c
/*
 * MIT License
 *
 * Copyright (c) 2018 Lewis Van Winkle
 *
 * Permission is hereby granted, free of charge, to any person obtaining a copy
 * of this software and associated documentation files (the "Software"), to deal
 * in the Software without restriction, including without limitation the rights
 * to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
 * copies of the Software, and to permit persons to whom the Software is
 * furnished to do so, subject to the following conditions:
 *
 * The above copyright notice and this permission notice shall be included in all
 * copies or substantial portions of the Software.
 *
 * THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
 * IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
 * FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
 * AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
 * LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,
 * OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE
 * SOFTWARE.
 */


#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <arpa/inet.h>
#include <netdb.h>
#include <unistd.h>
#include <errno.h>

#define ISVALIDSOCKET(s) ((s) >= 0)
#define CLOSESOCKET(s) close(s)
#define SOCKET int
#define GETSOCKETERRNO() (errno)

#include <stdio.h>
#include <string.h>
#include <time.h>

int main() {
    printf("Configuring local address...\n");

    struct addrinfo hints;
    memset(&hints, 0, sizeof(hints));
    hints.ai_family = AF_INET;
    hints.ai_socktype = SOCK_STREAM;
    hints.ai_flags = AI_PASSIVE;

```

```
    struct addrinfo *bind_address;
    getaddrinfo(0, "9001", &hints, &bind_address);


    printf("Creating socket...\n");
    SOCKET socket_listen;
    socket_listen = socket(bind_address->ai_family,
            bind_address->ai_socktype, bind_address->ai_protocol);
    if (!ISVALIDSOCKET(socket_listen)) {
        fprintf(stderr, "socket() failed. (%d)\n", GETSOCKETERRNO());
        return 1;
    }


    printf("Binding socket to local address...\n");
    if (bind(socket_listen,
                bind_address->ai_addr, bind_address->ai_addrlen)) {
        fprintf(stderr, "bind() failed. (%d)\n", GETSOCKETERRNO());
        return 1;
    }
    freeaddrinfo(bind_address);


    printf("Listening...\n");
    if (listen(socket_listen, 10) < 0) {
        fprintf(stderr, "listen() failed. (%d)\n", GETSOCKETERRNO());
        return 1;
    }


    printf("Waiting for connection...\n");
    struct sockaddr_storage client_address;
    socklen_t client_len = sizeof(client_address);
    SOCKET socket_client = accept(socket_listen,
            (struct sockaddr*) &client_address, &client_len);
    if (!ISVALIDSOCKET(socket_client)) {
        fprintf(stderr, "accept() failed. (%d)\n", GETSOCKETERRNO());
        return 1;
    }


    printf("Client is connected... ");
    char address_buffer[100];
    getnameinfo((struct sockaddr*)&client_address,
            client_len, address_buffer, sizeof(address_buffer), 0, 0,
            NI_NUMERICHOST);
    printf("%s\n", address_buffer);


    printf("Reading request...\n");
    char request[1024];
    //HTTP Request 전송
    int bytes_received = recv(socket_client, request, 1024, 0);
    printf("Received %d bytes.\n", bytes_received);
    //printf("%.*s", bytes_received, request); 사용자 브라우저나 사용자 환경같은거 보낸
```

```
109     printf("Sending response...\n");
110     //HTTP Response
111     const char *response =
112         "HTTP/1.1 200 OK\r\n"
113         "Connection: close\r\n"
114         "Content-Type: text/plain\r\n\r\n" //text-plain -> 암호화 되지 않음
115         "Local time is: ";
116     int bytes_sent = send(socket_client, response, strlen(response), 0);
117     printf("Sent %d of %d bytes.\n", bytes_sent, (int)strlen(response));
118 
119     time_t timer;
120     time(&timer);
121     char *time_msg = ctime(&timer);
122     //HTTP 보내는 데 시간 걸리니까 time은 따로 보내기
123     bytes_sent = send(socket_client, time_msg, strlen(time_msg), 0);
124     printf("Sent %d of %d bytes.\n", bytes_sent, (int)strlen(time_msg));
125 
126 
127     //시간 보낸 후 종료
128     printf("Closing connection...\n");
129     CLOSESOCKET(socket_client);
130 
131     printf("Closing listening socket...\n");
132     CLOSESOCKET(socket_listen);
133 
134 
135     printf("Finished.\n");
136 
137     return 0;
138 }
139 
```

Colored by Color Script