**연결형 서비스 보고서 [1-3]**컴퓨터 네트워크 02분반

2142851 김형준

**달력 정보 받고, 출력하기**

**목차**

1. **코드**
   1. TCPServer.cpp
   2. TCPClient.cpp
   3. 간략한 설명
2. **사진**
   1. 전체화면
   2. TCPClient
   3. TCPServer

* **코드**

**TCPServer.cpp**

*// Common.h 헤더 파일을 사용*

*#include* "../Common.h"

*#include* <cstring>

using namespace std;

*// 서버 포트 번호: 9000*

*// 버퍼 크기: 512*

*#define* SERVERPORT 9000

*#define* BUFSIZE    512

*// Calendar Size*

*#define* CALSIZE 300

*// 윤년인지 판별*

int is\_leap\_year(int *year*)

{

*if* ((year > 0) && !(year % 4) && ((year % 100) || !(year % 400)))

*return* 1;

*return* 0;

}

*// AD 1년 1월 1일 12:00 AM 을 기준으로 year 까지 몇 번의 윤년이 있었는가?*

int count\_leaps(int *year*) {

*return* (year - 1) / 4 - (year - 1) / 100 + (year - 1) / 400;

}

*// AD 1년 1월 1일부터 몇일이 지났는가?*

int count\_days(int *year*) {

*return* (year - 1) \* 365 + count\_leaps(year);

}

*// 주어진 날의 수를 해수로 계산*

int count\_years(int *days*) {

*return* (1 + (days - count\_leaps(days / 365)) / 365);

}

*// 지정한 달은 며칠까지 있는가?*

int get\_days\_of\_the\_month(int *year*, int *month*) {

*// 각달의날짜수*

    static int days\_of\_month[] = {31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31};

*// 해당월의날짜수*

    int days = days\_of\_month[month - 1];

*// 윤년처리*

*if* (is\_leap\_year(year) && month == 2)

        days++;

*return* days;

}

*// AD 1년 1월 1일 부터 며칠이 지났는가?*

int get\_total\_days(int *year*, int *month*, int *day*) {

*// 작년까지의 총 날짜*

    int total\_days = count\_days(year);

*// 올해 지난달 까지의 총 날짜*

    int m;

*for* (m = 1; m < month; ++m) {

        total\_days += get\_days\_of\_the\_month(year, m);

    }

*// 이번 달 오늘까지의 총 날짜*

    total\_days += day;

*return* total\_days;

}

*// 주어진 날짜의 요일 계산*

*// 0 : 일요일, 1 : 월요일, ... 6: 토요일*

int get\_weekday(int *year*, int *month*, int *day*) {

*return* get\_total\_days(year, month, day) % 7;

}

char calendar[CALSIZE];

*// calendar를 그린다*

void draw\_calendar(int *year*, int *month*) {

    int days = get\_days\_of\_the\_month(year, month);

*// 1일의 요일을 구하고 빈칸의 개수를 계산한다*

    int day = 1 - get\_weekday(year, month, 1);

    int i, end;

    static char months[12][10] = {"January", "February", "March", "April", "May", "June", "July", "August", "September", "October", "November", "December"};

    printf("[%4d] Calendar\n", year);

    sprintf(calendar, "[%4d] Calendar\n", year);

    printf("%10s\n", months[month - 1]);

    sprintf(calendar + strlen(calendar), "%10s\n", months[month - 1]);

    printf("%4s%4s%4s%4s%4s%4s%4s\n", "Sun", "Mon", "Tue", "Wed", "Thu", "Fri", "Sat");

    sprintf(calendar + strlen(calendar), "%4s%4s%4s%4s%4s%4s%4s\n", "Sun", "Mon", "Tue", "Wed", "Thu", "Fri", "Sat");

    printf("----------------------------\n");

    sprintf(calendar + strlen(calendar), "----------------------------\n");

*// printf("%s", calendar);*

    end = days - day + 1;

*for* (i = 0; i < end; ++i) {

*// 빈칸 출력*

*if* (day < 1) {

            printf("%4s", " ");

            sprintf(calendar + strlen(calendar), "%4s", " ");

        }

*// 날짜 출력*

*else* {

            printf("%4d", day);

            sprintf(calendar + strlen(calendar), "%4d", day);

        }

        ++day;

*// 일요일이면 줄바꿈*

*if* (i % 7 == 6) {

            printf("\n");

            sprintf(calendar + strlen(calendar), "\n");

        }

    }

    printf("\n");

    sprintf(calendar + strlen(calendar), "\n");

*// printf("%s", calendar);*

}

bool isLeapYear(int *year*) {

*if*((year % 4 == 0 && year % 100 != 0 ) || year % 400 == 0)

*return* true;

*else*

*return* false;

}

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

{

*// 소켓 함수의 반환값을 저장하는 변수 선언*

    int retval;

*// 소켓 생성*

    SOCKET listen\_sock = socket(AF\_INET, SOCK\_STREAM, 0);

*// 유효한 소켓이 아닌 경우 err\_quit() 함수 호출*

*// 오류 출력 후 프로그램 종료*

*if* (listen\_sock == INVALID\_SOCKET) err\_quit("socket()");

*// bind()*

*// 구조체(struct) serveraddr 선언*

*// IPv4 + 포트*

    struct sockaddr\_in serveraddr;

*// serveraddr의 값을 0으로 초기화*

    memset(&serveraddr, 0, sizeof(serveraddr));

*// IPv4를 나타내는 AF\_INET 값 설정*

    serveraddr.sin\_family = AF\_INET;

*// 수신할 IP 주소 설정*

*// INADDR\_ANY: 서버가 시스템에 있는 모든 네트워크 인터페이스를 통해 들어오는 연결을 수신한다.*

    serveraddr.sin\_addr.s\_addr = htonl(INADDR\_ANY);

*// 수신할 포트 설정*

*// 포트: SERVERPORT 값 (#define SERVERPORT 9000)*

    serveraddr.sin\_port = htons(SERVERPORT);

*// bind(): 서버 소켓에 주소 할당*

*// listen\_sock: 서버 소켓의 파일 디스크립터*

    retval = bind(listen\_sock, (struct sockaddr \*)&serveraddr, sizeof(serveraddr));

*if* (retval == SOCKET\_ERROR) err\_quit("bind()");

*// listen()*

*// 클라이언트의 접속 대기*

    retval = listen(listen\_sock, SOMAXCONN);

*if* (retval == SOCKET\_ERROR) err\_quit("listen()");

*// 데이터 통신에 사용할 변수*

*// 통신할 소켓, 클라이언트 주소, 버퍼 선언*

    SOCKET client\_sock;

    struct sockaddr\_in clientaddr;

    socklen\_t addrlen;

    char buf[BUFSIZE + 1];

*while* (1) {

*// accept()*

*// 클라이언트 연결 요청이 들어올 시 소켓 생성*

*// (그 전까지 대기)*

        addrlen = sizeof(clientaddr);

        client\_sock = accept(listen\_sock, (struct sockaddr \*)&clientaddr, &addrlen);

*if* (client\_sock == INVALID\_SOCKET) {

            err\_display("accept()");

*break*;

        }

*// 접속한 클라이언트 정보 출력*

*// 클라이언트 접속 시 IP 주소와 포트 출력*

        char addr[INET\_ADDRSTRLEN];

        inet\_ntop(AF\_INET, &clientaddr.sin\_addr, addr, sizeof(addr));

        printf("\n[TCP 서버] 클라이언트 접속: IP 주소=%s, 포트 번호=%d\n",

            addr, ntohs(clientaddr.sin\_port));

*// 클라이언트와 데이터 통신*

*while* (1) {

*// 데이터 받기*

            retval = recv(client\_sock, buf, BUFSIZE, 0);

*// 오류 발생 시 break*

*if* (retval == SOCKET\_ERROR) {

                err\_display("recv()");

*break*;

            }

*// 수신 데이터 크기가 0일 시 break*

*else* *if* (retval == 0)

*break*;

*// 받은 데이터 출력*

            buf[retval] = '\0';

            printf("[TCP/%s:%d] %s\n", addr, ntohs(clientaddr.sin\_port), buf);

*// char charYear[10], charMonth[10];*

*// for(int i=0; i<4; i++) {*

*//     charYear[i] = buf[i];*

*// }*

*// for(int i=0; i<2; i++) {*

*//     charMonth[i] = buf[i+6];*

*// }*

*// printf("[charYear] %s\n", charYear);*

*// printf("[charMonth] %s\n", charMonth);*

*// int year = atoi(charYear);*

*// int month = atoi(charMonth);*

*// printf("[year] %d\n", year);*

*// printf("[month] %d\n", month);*

            int year, month;

            char dot;

            sscanf(buf, "%d.%c%d", &year, &dot, &month);

            printf("[strlen(calendar)] %ld\n", strlen(calendar));

            draw\_calendar(year, month);

*// 데이터 보내기*

*// retval = send(client\_sock, buf, retval, 0);*

            retval = send(client\_sock, calendar, 500, 0);

*if* (retval == SOCKET\_ERROR) {

                err\_display("send()");

*break*;

            }

        }

*// 소켓 닫기*

*// 연결 종료*

        close(client\_sock);

*// 종료 시 메시지 출력*

        printf("[TCP 서버] 클라이언트 종료: IP 주소=%s, 포트 번호=%d\n",

            addr, ntohs(clientaddr.sin\_port));

    }

*// 소켓 닫기*

    close(listen\_sock);

*return* 0;

}

**TCPClient.cpp**

*#include* "../Common.h"

*// 서버의 IP 주소: 127.0.0.1*

char \*SERVERIP = (char \*)"127.0.0.1";

*#define* SERVERPORT 9000

*#define* BUFSIZE    512

*#define* CALSIZE 300

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

{

*// 소켓 함수 반환값을 저장할 변수 선언*

    int retval;

*// 명령행 인수가 있으면 IP 주소로 사용*

*if* (argc > 1) SERVERIP = argv[1];

*// 소켓 생성*

*// socket() 함수 호출 -> TCP 소켓 생성*

    SOCKET sock = socket(AF\_INET, SOCK\_STREAM, 0);

*// 오류 발생 시 메시지 출력 후 종료*

*if* (sock == INVALID\_SOCKET) err\_quit("socket()");

*// connect()*

*// serveraddr: 주소 정보 설정*

    struct sockaddr\_in serveraddr;

*// 구조체 serveraddr 0으로 초기화*

    memset(&serveraddr, 0, sizeof(serveraddr));

*// AF\_INET: IPv4 주소 체계 사용함*

    serveraddr.sin\_family = AF\_INET;

*// inet\_pton(): 문자열 형태의 IP 주소를 이진 형태로 변환 후 sin\_addr에 저장*

    inet\_pton(AF\_INET, SERVERIP, &serveraddr.sin\_addr);

*// server의 포트 번호 설정*

    serveraddr.sin\_port = htons(SERVERPORT);

*// connect(): 서버 연결*

    retval = connect(sock, (struct sockaddr \*)&serveraddr, sizeof(serveraddr));

*// 연결 실패 시 오류 출력 후 종료*

*if* (retval == SOCKET\_ERROR) err\_quit("connect()");

*// 데이터 통신에 사용할 변수*

    char buf[BUFSIZE + 1];

*// 변수 len: 입력 데이터 길이 저장 변수*

    int len;

*// 서버와 데이터 통신*

*while* (1) {

*// 데이터 입력*

        printf("\n[보낼 데이터 (YYYY. MM)] ");

*// fgets를 통해 데이터 입력 받고, buf에 저장*

*// 입력 실패 시 break*

*if* (fgets(buf, BUFSIZE + 1, stdin) == NULL)

*break*;

*// '\n' 문자 제거*

*// 입력 데이터 길이 계산 후 len에 저장*

        len = (int)strlen(buf);

*// \n 문자가 있는 경우 제거*

*if* (buf[len - 1] == '\n')

            buf[len - 1] = '\0';

*// 입력이 없다면 break*

*if* (strlen(buf) == 0)

*break*;

*// send(): 데이터를 소켓에 작성*

        retval = send(sock, buf, (int)strlen(buf), 0);

*if* (retval == SOCKET\_ERROR) {

            err\_display("send()");

*break*;

        }

*// 데이터 전송한 경우: 전송 바이트 출력*

        printf("[TCP 클라이언트] %d바이트를 보냈습니다.\n", retval);

*// 데이터 받기*

*// recv(): 데이터를 소켓에서 읽어옴*

        retval = recv(sock, buf, 500, MSG\_WAITALL);

*// 수신 실패 or 연결 종료: err\_display() 출력 후 break*

*if* (retval == SOCKET\_ERROR) {

            err\_display("recv()");

*break*;

        }

*else* *if* (retval == 0)

*break*;

*// 받은 데이터 출력*

        buf[retval] = '\0';

        printf("[TCP 클라이언트] %d바이트를 받았습니다.\n", 500);

        printf("[받은 데이터]\n%s\n", buf);

    }

*// 소켓 닫기*

    close(sock);

*return* 0;

}

**간략한 설명**

* Year, month를 dot를 기준으로 구분합니다.
* int year, month;
* char dot;
* sscanf(buf, "%d.%c%d", &year, &dot, &month);
* sscanf를 이용하여 year, month 변수에 값을 할당합니다.
* draw\_calendar 함수를 이용하여 calendar를 그립니다.
* void draw\_calendar(int *year*, int *month*)
* printf와 동시에 sprintf를 이용하여 calendar 변수에 즉시 저장합니다.
* printf("----------------------------\n");
* sprintf(calendar + strlen(calendar), "----------------------------\n");
* 이후 최적화를 위해 calendar 변수의 length를 서버 콘솔창에 출력합니다.
* **사진**

**전체화면**

텍스트, 스크린샷, 소프트웨어, 멀티미디어 소프트웨어이(가) 표시된 사진

자동 생성된 설명

**TCPServer**

텍스트, 스크린샷, 디스플레이, 소프트웨어이(가) 표시된 사진

자동 생성된 설명

**TCPClient**

텍스트, 스크린샷, 디스플레이, 소프트웨어이(가) 표시된 사진

자동 생성된 설명