



МИНИСТЕРСТВО НАУКИ
И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное
образовательное учреждение высшего образования
«НОВОСИБИРСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»



Кафедра теоретической и прикладной информатики
Лабораторная работа № 2
по дисциплине «Операционные системы и компьютерные сети»

ТЕХНОЛОГИЯ КЛИЕНТ-СЕРВЕР: ЭХО-ПОВТОР

Бригада 9

Группа ПМ-24 ГЕРАСИМЕНКО ВАДИМ
ПАРАСКУН ИВАН

Преподаватели СИВАК МАРИЯ АЛЕКСЕЕВНА
КОБЫЛЯНСКИЙ ВАЛЕРИЙ ГРИГОРЬЕВИЧ
Новосибирск, 2024

1. Цель работы

Изучить основные принципы разработки клиент-серверных приложений на примере простейшей однопользовательской программы.

2. Условие

Задача:

Написать простейшее приложение с одним сервером и одним клиентом, используя API-интерфейс низкого уровня.

Вариант задания: (1 вариант)

Клиент пересыпает серверу данные (строки текста). Сервер возвращает клиенту полученные данные, включив в конец каждого предложения количество буквенных символов в нем.

3. Ход работы

Текст программы:

Файл sock.h:

```
#include <stdio.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <netdb.h>
#include <netinet/in.h>
#include <arpa/inet.h>
#include <string.h>
#include <ctype.h>
#include <unistd.h>

#define PORT 2009

int errCatch(char *msg, int sock)
{
    printf("%s", msg);
    close(sock);
    return 0;
}
```

Файл server.c:

```
#include "sock.h"

void req_proc(char *req, char *resp)
{
    char *sntnc = req;
    char *end = req + strlen(req);
    int resplen = 0;

    while (sntnc < end) {
        char *punct = strpbrk(sntnc, ".!?");
        if (punct == NULL) {
            sprintf(resp + resplen, "%s", sntnc);
            resplen += strlen(sntnc);
            break;
        }

        int sntnclen = punct - sntnc + 1;
        int lttrcnt = 0;

        for (char *ch = sntnc; ch <= punct; ch++)
            if (isalpha(*ch))
                lttrcnt++;

        sprintf(resp + resplen, "%.*s (%d)\n", sntnclen, sntnc,
        lttrcnt);
        resplen += sntnclen + 5;
        sntnc = punct + 1;
    }

    resp[resplen - 1] = '\0';
}

int main() {
    int ssock = socket(AF_INET, SOCK_STREAM, 0);

    if (ssock == -1)
        return errCatch("Unable to create socket\n", ssock);
```

```

struct sockaddr_in addr = {
    .sin_family = AF_INET,
    .sin_port = htons(PORT),
    .sin_addr.s_addr = htonl(INADDR_ANY)
};

if (bind(ssock, (const struct sockaddr *)&addr, sizeof(addr)) == -1)
    return errCatch("Unable to bind\n", ssock);

printf("Server started at %s, port %d\n",
    inet_ntoa(addr.sin_addr), htons(addr.sin_port));

if (listen(ssock, 10) == -1)
    return errCatch("Unable to listen\n", ssock);

struct sockaddr_in from;
int csock = accept(ssock, NULL, NULL);

if (csock == -1)
    return errCatch("Unable to accept\n", ssock);

printf("New connection accepted from %s, port %d\n",
    inet_ntoa(from.sin_addr), htons(from.sin_port));

while (1) {
    const int reqlen = 1024;
    char req[reqlen];

    if (recv(csock, req, reqlen, 0) == -1)
        return errCatch("Unable to recv\n", ssock);

    printf("Data received\n");

    if (req[0] == '~') {
        char *resp = "Server closed\n";

        if (send(csock, resp, strlen(resp), 0) == -1)
            return errCatch("Unable to send\n", ssock);

        puts(resp);
        break;
    }
}

```

```
    } else {
        char resp[2048];
        req_proc(req, resp);
        int resplen = strlen(resp);

        printf("Sending response from server\n");

        if (send(csock, resp, resplen, 0) == -1)
            return err_catch("Unable to send\n", ssock);

        printf("Response sent\n\n");
        memset(resp, 0, resplen);
    }
}

close(ssock);
return 0;
}
```

Файл **client.c**:

```
#include "sock.h"

int main() {
    int csock = socket(AF_INET, SOCK_STREAM, 0);

    if (csock == -1)
        return errCatch("Unable to create socket\n", csock);

    char ip[16];
    printf("Enter IP: ");
    fgets(ip, 16, stdin);

    struct sockaddr_in cin = {
        .sin_family = AF_INET,
        .sin_port = htons(PORT),
        .sin_addr.s_addr = inet_addr(ip)
    };

    if (connect(csock, (struct sockaddr *)&cin, sizeof(cin)) == -1)
        return errCatch("Unable to connect\n", csock);

    printf("Connection made successfully\n");

    while (1) {
        printf("Enter text to send request or '~' to shutdown
server\n");

        int reqlen = 1024;
        char req[reqlen];
        fgets(req, reqlen, stdin);

        printf("Sending request from client\n");

        if (send(csock, req, reqlen, 0) == -1)
            errCatch("Unable to send\n", csock);

        int resplen = 2048;
        char resp[resplen];
        int s;
```

```

if ((s = recv(csock, resp, resplen, 0)) == -1)
    return errCatch("Unable to recv\n", csock);

if (req[0] == '~') {
    puts(resp);
    break;
}

printf("\nResponse from server: %s\n", resp);

memset(req, 0, strlen(req));
memset(resp, 0, strlen(resp));
}

close(csock);
return 0;
}

```

Проверка работоспособности:

```

Last login: Sun Oct  6 19:01:02 on ttys004
[vadimgerasimenko@MacBook-Air-Vadim-2 ~ % cd Documents/nstu/KC/lab2
[vadimgerasimenko@MacBook-Air-Vadim-2 lab2 % cd output
[vadimgerasimenko@MacBook-Air-Vadim-2 output % ./server
Server started at 0.0.0.0, port 2009
New connection accepted from 0.0.0.0, port 0
Data received
Sending response from server
Response sent
-
Data received
Server closed
vadimgerasimenko@MacBook-Air-Vadim-2 output %

Last login: Sun Oct  6 19:44:29 on ttys003
[vadimgerasimenko@MacBook-Air-Vadim-2 ~ % cd Documents/nstu/KC/lab2/output
[vadimgerasimenko@MacBook-Air-Vadim-2 output % ./client
Enter IP: 0.0.0.0
Connection made successfully
Enter text to send request or '~' to shutdown server
Hello. My name is Vadim. Okey, how are you?
Sending request from client
-
Response from server: Hello. (5)
My name is Vadim. (13) Okey, how are you? (13)
Enter text to send request or '~' to shutdown server
-
Sending request from client
Server closed
vadimgerasimenko@MacBook-Air-Vadim-2 output %

```

Клиент:

```

Last login: Sun Oct  6 19:44:29 on ttys003
[vadimgerasimenko@MacBook-Air-Vadim-2 ~ % cd Documents/nstu/KC/lab2/output
[vadimgerasimenko@MacBook-Air-Vadim-2 output % ./client
Enter IP: 0.0.0.0
Connection made successfully
Enter text to send request or '~' to shutdown server
Hello. My name is Vadim. Okey, how are you?
Sending request from client

Response from server: Hello. (5)
My name is Vadim. (13) Okey, how are you? (13)
Enter text to send request or '~' to shutdown server
-
Sending request from client
Server closed

vadimgerasimenko@MacBook-Air-Vadim-2 output %

```

Сервер:

```
Last login: Sun Oct  6 19:01:02 on ttys064
[vadimgerasimenko@MacBook-Air-Vadim-2 ~ % cd Documents/nstu/KC/lab2
[vadimgerasimenko@MacBook-Air-Vadim-2 lab2 % cd output
[vadimgerasimenko@MacBook-Air-Vadim-2 output % ./server
Server started at 0.0.0.0, port 2009
New connection accepted from 0.0.0.0, port 0
Data received
Sending response from server
Response sent

Data received
Server closed

vadimgerasimenko@MacBook-Air-Vadim-2 output %
```

4. Вывод

Вывод: в ходе работы были изучены основные принципы разработки клиент-серверных приложений на примере простейшей однопользовательской программы.