

Kommunikationsnetze

Sommersemester 2003 - Übung 9

Thema: Datei auf Anforderung senden

In dieser Übung sollen Sie das Server-Programm aus Übung 8 weiterentwickeln. Das Server Programm soll nun die angeforderte Datei öffnen und über das Netz an den Browser senden.

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

int main()
{
    int server_socket, client_socket, anzahl, laenge;
    struct sockaddr_in serverinfo, clientinfo;
    char dateiname[100], empfangen[1000];
    char *position1, *position2;
    char text_http_ok[] = "HTTP/1.0 200 OK\r\n\r\n";
    char text_html_anfang[] = "<HTML><BODY>";
    char text_html_ende[] = "</BODY></HTML>";

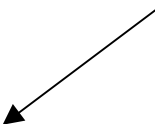
    server_socket = socket(AF_INET, SOCK_STREAM, 0);

    serverinfo.sin_family = AF_INET;
    serverinfo.sin_addr.s_addr = inet_addr("192.168.103.xxx");
    serverinfo.sin_port = htons(5000);
    laenge = sizeof(serverinfo);

    bind(server_socket, &serverinfo, laenge);
    listen(server_socket, 3);

    while (1)
    {
        client_socket = accept(server_socket, &clientinfo, &laenge);
        printf("\nVerbindung mit %s", inet_ntoa(clientinfo.sin_addr));
        anzahl = read(client_socket, empfangen, sizeof(empfangen));
        empfangen[anzahl]=0;
        write(client_socket, text_http_ok, strlen(text_http_ok));
        write(client_socket, text_html_anfang, strlen(text_html_anfang));
        if (position1 = strstr(empfangen, "GET"))
        {
            if (position2 = strstr(empfangen, "HTTP"))
            {
                laenge = position2 - position1 - 6;
                strncpy(dateiname, position1+5, laenge);
                dateiname[laenge]=0;
                printf("\nGET Befehl fuer Datei %s gefunden", dateiname);
                write(client_socket, "Dateiname = ", 12);
                write(client_socket, dateiname, laenge);
            }
            else
            {
                printf("\nDer HTTP-Parser hat keinen HTTP-Befehl gefunden");
                write(client_socket, "kein HTTP-Befehl gefunden", 25);
            }
        }
        else
        {
            printf("\nDer HTTP-Parser hat keinen GET-Befehl gefunden");
            write(client_socket, "kein GET-Befehl gefunden", 24);
        }
        printf("\n");
        write(client_socket, text_html_ende, strlen(text_html_ende));
        close(client_socket);
    }
    close(server_socket);
}
```

hier bitte die
IP-Adresse
Ihres PC
eintragen !



Aufgabe 1: Übersetzen Sie das Beispielprogramm und erzeugen Sie eine ausführbare Datei. Starten Sie die ausführbare Datei und prüfen Sie, ob Ihr selbst geschriebener Server läuft (Linux-Befehl `ps`). Starten Sie dann einen Browser und fordern Sie damit eine Datei von Ihrem Server an. Geben Sie dazu eine passende URL im Adressfeld Ihres Browsers ein.

Beispiel: Falls Ihr Server die IP-Adresse `192.168.103.24` hat, dann lautet die vollständige URL der Datei `test.html`:

```
http:// 192.168.103.24:5000/test.html
```

Hinweis: Da der selbst geschriebene Server nicht den für `http` reservierten Port `80` verwendet (sondern Port `5000`, siehe Quellcode), muss die Portnummer beim Aufruf einer Web-Seite mit angegeben werden.

Aufgabe 2: Erweitern Sie das vorliegende Programm so, dass es versucht, die vom Browser angeforderten Datei zu öffnen. Falls die Datei geöffnet werden kann, soll der Text "Datei existiert" zum Browser gesendet werden. Falls die Datei nicht geöffnet werden kann, soll der Text "Datei existiert nicht" an den Browser gesendet werden.

Hinweis: Sie benötigen hierzu einen Dateizeiger (Variablentyp `FILE *`). Öffnen Sie die Datei als Textdatei zum Lesen (Funktion `fopen()` mit dem Modus `rt = read text`). Prüfen Sie, ob der Dateizeiger danach den Inhalt `NULL` hat.

Aufgabe 3: Erweitern Sie das vorliegende Programm so, dass es den Inhalt der geöffneten Datei an den Browser sendet.

Hinweis: Sie können die geöffnete Datei zeichenweise auslesen (Funktion `fgetc()`) und jedes Zeichen in den Socket schreiben (Funktion `write()`). Prüfen Sie bei jedem Zeichen, ob das Ende der Datei erreicht worden ist (`EOF = End of File`).