Отчёт по лабораторной работе 14

Именованные каналы

Осокин Иван Юрьевич

Содержание

## 0.1 Цель работы

Цель работы — приобрести простейшие навыки по работе с именованными каналами. Репозиторий автора находится по адресу <https://github.com/horhik/study_2022-2023_os-intro>.

# 1 Выполнение лабораторной работы

## 1.1 Написание программ

В ходе выполнения лабораторной работы необходимо было написать программу, запускающую сервер, и программу, создающую произвольное количество клиентов, отправляющих на сервер локальное время.

Данное задание было реальзовано в следующих программах:

common.h:

/\*  
\* common.h - заголовочный файл со стандартными определениями  
\*/  
  
#ifndef \_\_COMMON\_H\_\_  
#define \_\_COMMON\_H\_\_  
  
#include <stdio.h>  
#include <stdlib.h>  
#include <string.h>  
#include <errno.h>  
#include <sys/types.h>  
#include <sys/stat.h>  
#include <fcntl.h>  
#include <time.h>  
  
#define FIFO\_NAME "/tmp/fifo"  
#define MAX\_BUFF 80  
  
#endif /\* \_\_COMMON\_H\_\_ \*/

server.c:

/\*  
\* server.c - реализация сервера  
\*  
\* чтобы запустить пример, необходимо:  
\* 1. запустить программу server на одной консоли;  
\* 2. запустить программу client на другой консоли.  
\*/  
  
#include "common.h"  
  
int main()  
{  
 clock\_t startstamp, endstamp;  
 double totaltime;  
   
 int readfd; /\* дескриптор для чтения из FIFO \*/  
 int n;  
 char buff[MAX\_BUFF]; /\* буфер для чтения данных из FIFO \*/  
   
 /\* баннер \*/  
 printf("FIFO Server...\n");  
   
 /\* создаем файл FIFO с открытыми для всех  
 \* правами доступа на чтение и запись  
 \*/  
  
 startstamp = clock();  
 if(mknod(FIFO\_NAME, S\_IFIFO | 0666, 0) < 0)  
 {  
 fprintf(stderr, "%s: Невозможно создать FIFO (%s)\n",  
 \_\_FILE\_\_, strerror(errno));  
 exit(-1);  
 }  
   
 /\* откроем FIFO на чтение \*/  
 if((readfd = open(FIFO\_NAME, O\_RDONLY)) < 0)  
 {  
 fprintf(stderr, "%s: Невозможно открыть FIFO (%s)\n",  
 \_\_FILE\_\_, strerror(errno));  
 exit(-2);  
 }  
   
 /\* читаем данные из FIFO и выводим на экран \*/  
 while((n = read(readfd, buff, MAX\_BUFF)) > 0)  
 {  
 if(write(1, buff, n) != n)  
 {  
 fprintf(stderr, "%s: Ошибка вывода (%s)\n",  
 \_\_FILE\_\_, strerror(errno));  
 exit(-3);  
 }  
 }  
   
 close(readfd); /\* закроем FIFO \*/  
   
 /\* удалим FIFO из системы \*/  
 if(unlink(FIFO\_NAME) < 0)  
 {  
 fprintf(stderr, "%s: Невозможно удалить FIFO (%s)\n",  
 \_\_FILE\_\_, strerror(errno));  
 exit(-4);  
 }  
   
 endstamp = clock();  
 totaltime = ((double)(endstamp - startstamp)) / CLOCKS\_PER\_SEC;  
   
 printf("Server had been running for %f seconds\n", totaltime);  
 exit(0);  
}

client.c:

/\*  
\* client.c - реализация клиента  
\*  
\* чтобы запустить пример, необходимо:  
\* 1. запустить программу server на одной консоли;  
\* 2. запустить программу client на другой консоли.  
\*/  
  
#include "common.h"  
  
#define MESSAGE "Hello Server!!!\n"  
  
char\* LocalTime()  
{  
 /\* Get local time \*/  
 time\_t rawtime;  
 struct tm \* timeinfo;  
  
 time ( &rawtime );  
 timeinfo = localtime ( &rawtime );  
 return asctime (timeinfo);  
}  
  
int CreateClient(int i)  
{  
 /\* Get access to FIFO \*/  
  
 printf("FIFO Client %i...\n", i);  
  
 int n = open(FIFO\_NAME, O\_WRONLY);  
 if(n < 0)  
 {  
 fprintf(stderr, "[FIFO Client %d] %s: Невозможно открыть FIFO (%s)\n",  
 i, \_\_FILE\_\_, strerror(errno));  
 exit(-1);  
 }  
 return n;  
}  
  
void PrintMessage(int client, int i, char\* gist)  
{  
 /\* Dispatch message to server \*/  
 char\* msg[MAX\_BUFF];  
 sprintf(msg, "[FIFO Client %d] %s", i, gist);  
  
 int msglen = strlen(msg);  
 if(write(client, msg, msglen) != msglen)  
 {  
 fprintf(stderr, "[FIFO Client %d] %s: Ошибка записи в FIFO (%s)\n",  
 i, \_\_FILE\_\_, strerror(errno));  
 exit(-2);  
 }  
}  
  
int CloseClient(int client, int i)  
{  
 /\* Closes specified client \*/  
 printf("[FIFO Client %d]: Закрыт клиент %d\n", i, i);  
 return close(client);  
}  
  
  
int main(int argc, char\* argv[])  
{  
 int N; // amount of servers to be created  
 if (argc == 1) // N not specified  
 {  
 N = 1;  
 printf("%d server is to be created\n", N);  
 }  
 else  
 {  
 N = atoi(argv[1]);  
 printf("%d servers are to be created\n", N);  
 }  
  
 /\* Initialize N clients \*/  
 int jopakonya[N];  
 for (int i = 1; i <= N; i++)  
 {  
 jopakonya[i] = CreateClient(i);  
 }  
  
 /\* Clients' job described here \*/  
 char\* message;  
 for (int i = 1; i <= N; i++)  
 {  
 message = LocalTime();  
 PrintMessage(jopakonya[i], i, message);  
 }  
  
 /\* Close all clients \*/  
 for (int i = 1; i <= N; i++)  
 {  
 CloseClient(jopakonya[i], i);  
 }  
  
 exit(0);  
}

Makefile:

all: server client  
  
server: server.c common.h  
 gcc server.c -o server  
  
client: client.c common.h  
 gcc client.c -o client  
  
clean:  
 -rm server client \*.o

## 1.2 Ответы на контрольные вопросы

1. В чем ключевое отличие именованных каналов от неименованных?

*Ответ*: именованные каналы имеют идентификатор.

1. Возможно ли создание неименованного канала из командной строки?

*Ответ*: sí.

1. Возможно ли создание именованного канала из командной строки?

*Ответ*: sí.

1. Опишите функцию языка С, создающую неименованный канал.

*Ответ*: int pipe (int filedes[2]).

1. Опишите функцию языка С, создающую именованный канал.

*Ответ*: int mkfifo(const char \*pathname, mode\_t mode).

1. Что будет в случае прочтения из fifo меньшего числа байтов, чем находится в канале? Большего числа байтов?

*Ответ*: в первом случае возвращается требуемое число байтов, остаток сохраняется для последующих чтений. При чтении числа байт, большего чем находится в канале, возвращается доступное число байт.

1. Что будет в случае записи в fifo меньшего числа байтов, чем позволяет буфер? Большего числа байтов?

*Ответ*: свойство - анализ кода; для анализа необходимо скомпилировать программу.

1. Могут ли два и более процессов читать или записывать в канал?

*Ответ*: ну у меня получилось. Значит, вроде да.

1. Опишите функцию write (тип возвращаемого значения, аргументы и логику работы). Что означает 1 (единица) в вызове этой функции в программе server.c (строка 42)?

*Ответ*: write(int fildes, const void \*buf, size\_t nbyte, off\_t offset); 1 (единица) - значит 1 (единица).

1. Опишите функцию strerror.

*Ответ*: char\* strerror( int errnum ).

## 1.3 Заключение

В ходе выполнения лабораторной работы были изучены простейшие навыки по работе с именованными каналами. Цель работы была достигнута.