Atividade 5 - Implementação de Aplicação Cliente/Servidor com Sockets TCP - Códigos

Gabriel Hidasy Rezende

October 26, 2015

```c
//Cliente.C
#include <sys/socket.h>
#include <sys/types.h>
#include <arpa/inet.h>
#include <netinet/in.h>
#include <stdio.h>
#include <netdb.h>
#include <string.h>
#include <errno.h>
#include <string.h>
#include <stdlib.h>
#include <unistd.h>

#define MAXLINE 4096

int main(int argc, char **argv) {
  int    sockfd, n;
  char   recvline[MAXLINE + 1];
  char   error[MAXLINE + 1];
  struct sockaddr_in servaddr;

  if (argc != 2) {
    strcpy(error,"uso: ");
    strcat(error,argv[0]);
    strcat(error," <IPaddress>");
    perror(error);
    exit(1);
  }

  /* Abre o socket */
```

```
  if ( (sockfd = socket(AF_INET, SOCK_STREAM, 0)) < 0) {
    perror("socket error");
    exit(1);
  }

  /* Prepara as estruturas que serao usadas */
  bzero(&servaddr, sizeof(servaddr));
  servaddr.sin_family = AF_INET;
  servaddr.sin_port = htons(12344);

  /* Aqui eh criada a estrutura que sera usada para a conexao */
  if (inet_pton(AF_INET, argv[1], &servaddr.sin_addr) <= 0) {
    perror("inet_pton error");
    exit(1);
  }

  /* A conexao eh feita */
  if (connect(sockfd, (struct sockaddr *) &servaddr,
      sizeof(servaddr)) < 0) {
    perror("connect error");
    exit(1);
  }
  struct sockaddr_in sockname;
  /* Seus dados sao impressos na tela */
  getsockname(sockfd, (struct sockaddr *) &sockname, (socklen_t
      *) sizeof(sockname));
  printf("local IP is %s\n",inet_ntoa(sockname.sin_addr));

  /* Dados sao recebidos e impressos */
  while ( (n = read(sockfd, recvline, MAXLINE)) > 0) {
    recvline[n] = 0;
    if (fputs(recvline, stdout) == EOF) {
      perror("fputs error");
      exit(1);
    }
  }

  if (n < 0) {
    perror("read error");
    exit(1);
  }

  exit(0);
}
```

```c
//Servidor.c
#include <stdio.h>
#include <stdlib.h>
#include <errno.h>
#include <string.h>
#include <netdb.h>
#include <sys/types.h>
#include <netinet/in.h>
#include <sys/socket.h>
#include <time.h>
#include <unistd.h>
#include <arpa/inet.h>

#define LISTENQ 10
#define MAXDATASIZE 100

int main (int argc, char **argv) {
  int    listenfd, connfd;
  struct sockaddr_in servaddr;
  char   buf[MAXDATASIZE];
  time_t ticks;

  /* No trecho abaixo um socket eh inicializado */
  if ((listenfd = socket(AF_INET, SOCK_STREAM, 0)) == -1) {
    perror("socket");
    exit(1);
  }

  /* Estruturas sao preparadas */
  bzero(&servaddr, sizeof(servaddr));
  servaddr.sin_family    = AF_INET;
  servaddr.sin_addr.s_addr = htonl(INADDR_ANY);
  servaddr.sin_port      = htons(12344);

  /* A porta (12344) e reservada */
  if (bind(listenfd, (struct sockaddr *)&servaddr,
      sizeof(servaddr)) == -1) {
    perror("bind");
    exit(1);
  }

  /* O servidor passa a receber conexoes */
  if (listen(listenfd, LISTENQ) == -1) {
    perror("listen");
```

```
    exit(1);
  }

  /* Loop principal do servidor, aqui ele escuta uma conexao e ao
      receber uma */
  /* retorna a data do dia, note que esse servidor nao eh
      multi-thread mas */
  /* ainda funciona bem com varios clientes pois ele mesmo se
      encarrega de */
  /* fechar a conexao */
  for ( ; ; ) {
    if ((connfd = accept(listenfd, (struct sockaddr *) NULL,
        NULL)) == -1 ) {
      perror("accept");
      exit(1);
    }

    ticks = time(NULL);
    struct sockaddr_in remote;
    int len = sizeof(remote);
    getpeername(connfd, (struct sockaddr *) &remote, &len);
    printf("Remote socket %s\n",inet_ntoa(remote.sin_addr));

    /* Aqui a mensagem de resposta eh montada */
    snprintf(buf, sizeof(buf), "%.24s\r\n", ctime(&ticks));
    /* E enviada */
    write(connfd, buf, strlen(buf));
    /* O socket dessa conexao eh terminado e vamos para o proximo
        cliente */
    close(connfd);
  }
  return(0);
}
```