



work\_thread.c

1. #include "thread.h"
3. #define ARGC 10
5. **void** thread\_start(**int** c)
6. {
7. pthread\_t id;
8. pthread\_create(&id,NULL,work\_thread,(**void** \*)c);
9. }
11. **void** get\_argv(**char** buff[],**char** \* myargv[])
12. {
13. //strtok()
14. **char** \*str = strtok(buff," ");
15. myargv[0] = str;
16. **int** i=1;
17. **while**((str=strtok(NULL," ")) != NULL)
18. {
19. myargv[i++]=str;
20. printf("%s,%s\n",myargv[0],myargv[1]);
21. }
22. }
23. /\*
24. \*gong zuo xian cheng
25. \*
26. \*/
27. **void**\* work\_thread(**void** \* arg)
28. {
29. **int** c = (**int**)arg;
31. **while**(1)
32. {
33. **int** pipefd[4]; // 0 du 1 xie
34. pipe(pipefd); // zi jin cheng he fu jin cheng fen bie guan yi ge
35. **char** buff[128] = {0};
36. **int** n = recv(c,buff,127,0);
37. **if**( n <= 0)
38. {
39. close(c);
40. printf("one client close\n");
41. **break**;
42. }
43. **int** bufflen = strlen(buff);
44. buff[bufflen-1]='\0';
45. **char** \* myargv[ARGC] = {0};
46. get\_argv(buff,myargv);
47. pid\_t pid = fork();
48. // int pipefd[4]; // 0 du 1 xie
49. // pipe(pipefd); // zi jin cheng he fu jin cheng fen bie guan yi ge
50. // pid\_t pid = fork();
51. **if**( pid == 0)
52. {
53. //guan dao he dup2(1,pipfd[1])
54. //guan dao he dup2(2,pipfd[1])
55. close(pipefd[0]);
56. dup2(pipefd[1],2);
57. dup2(pipefd[1],1);
58. execvp(myargv[0],myargv);
59. close(pipefd[1]);
60. exit(0);
62. }
63. **else**
64. {
65. close(pipefd[1]);
66. **char** buff2[128]={0};
67. **if**(read(pipefd[0],buff2,127) == 0)
68. {
69. strncpy(buff2,myargv[0],strlen(myargv[0]));
70. strcat(buff2," success");
71. send(c,buff2,strlen(buff2),0);
72. }
73. **else**
74. {
75. send(c,buff2,strlen(buff2),0);
76. }
77. close(pipefd[0]);
78. }
79. }
81. }