UNIX Lab：MiniShell With C (1/2)

Objective:

In the Linux platform, using C/C++ language to write a Mini Shell command interpreter environment (i.e. similar to Bash Shell). The environment can be recycled to accept user input commands and some parameters from standard input, and it can interpret and execute the above command, the final results of user input commands shown up on the standard output.

```
bash-2.03$ ps_03
**************welcome to mini shell**************
MINI SHELL#pwd
/home/unixmng/oscourses/ps_prog
MINI SHELL#exit
****************** mini shell exit******************
bash-2.03$
```

Requirements:

- Support the user to enter a command line and a plurality of parameters, and the analytical execution, and output results;
- Support the cd command, if no parameters are returned to the current user login directory (see notes below);
- Support to the "current path" and "username" to prompt string;
- Support for automatic ignores the processing space in the command line;
- Support automatically ignore treatment on the command line tab bond;
- Support for input and output, error output redirection;
- Support in a row with ";" (marked) to perform multiple commands separated and a plurality of the order of the parameters, such as:
  MINI SHELL#pwd**;** ls –l**;**date

Commited Files:

- Source Code
- Running Case

related head files：

```
#include <stdio.h>
#include <string.h>
#include <unistd.h>
#include <pwd.h>
```

related functions：

- fork 、execvp、wait
- strcmp、strcpy、strncpy
- fopen、fclose、fscanf、fprintf、fgets、fputs
- sprintf ： translate to string
- atoi ： translate to int
- getlogin ：
- getcwd ：
- chdir ： change current directory
- getenv ：
- signal ：

```
[root@solaris10 leesh]#cat waitchildwithsignal_v1.c
/**
*       waitchildwithsignal_v1.c
*       func: parent process wait for child process exit asynchronously.
*/
#include <stdio.h>
#include <signal.h>
#define MASKVALH  0x0000FF00
#define MASKVALL  0x000000FF
#define HBYTE(x)  ((x & MASKVALH) >>8)
#define LBYTE(x)  (x & MASKVALL)
```

```
#define LINEBUFSIZE 256
static void sig_child(int);
/**
* main func
*/
main()
{
        int p1, p2,cnt;
        char buf[LINEBUFSIZE];
        setbuf(stdout,NULL);
        if (signal(SIGCHLD,sig_child)==SIG_ERR){
                fprintf(stderr,"signal for SIGCHILD error\n");
        }

        while ((p1 = fork()) == -1);
        if (p1 == 0) {
                printf("child(1) pid=%d\n", getpid());
                sleep(1);
                exit(10);
        }
        while ((p2 = fork()) == -1);
        if (p2 == 0) {
                        printf("child(2) pid=%d\n", getpid());
                        sleep(2);
                        execlp("date","date",(char*)0);
                        exit(30);
        }
        printf("parent pid=%d\n", getpid());
        printf("welcome to leesh\n");
        while(1){
                printf("%s","leesh>");
                memset(buf,'\0',(LINEBUFSIZE-1));
                cnt=read(0,buf,254);
                if (cnt>0){
                  buf[cnt-1]='\0';
                }
                if (strcmp(buf,"exit")==0){
                        break;
                }
                printf("%s\n",buf);
        }
        printf("bye leesh\n");
        exit(0);
}
```

```
/**
* signal for child exit
*/
static void sig_child(int signo){
        int status, childpid;
        if ( signo == SIGCHLD ){
                if((childpid = wait(&status)) != -1){
                        if (LBYTE(status) == 0)
                                                    printf("wait child(pid=%d) exitcode=%d\n", childpid,
HBYTE(status));
                }
        }
}
[root@solaris10 leesh]#
```

- get user login directory

```
/////////////////////get user default dir
int
getuserdir (char *aoUserDir)
{
  char *LoginId;
  struct passwd *pwdinfo;
  if (aoUserDir == NULL)
    return -9;
  if ((LoginId = getlogin ()) == NULL) {
    perror ("getlogin");
    aoUserDir[0] = '\0';
    return -8;
  }
  if ((pwdinfo = getpwnam (LoginId)) == NULL) {
    perror ("getpwnam");
    return -7;
  }
  strcpy (aoUserDir, pwdinfo->pw_dir);
}
```