CSED211 Lab 10 Report

Shell Lab: Writing Your Own Unix Shell

20220041 김유진

I. Method

이번 실습은 간단한 shell을 구현하여 process control 및 signal handling을 익히는 것을 목표로 한다. 주어진 tsh.c 코드에서 채워지지 않은 함수 7가지(eval, builtin_cmd, do_bgfg, waitfg, sigchld_handler, sigint_handler, sigtstp_handler)를 구현하고, 이를 16가지 케이스(trace1~trace16)을 실행시켜보며 첨부된 reference shell(tshref)과 비교하여 알맞게 기능하고 있는지를 확인한다. shell은 사용자가 명령을 입력하면 그에 맞게 프로세스를 실행해주는 장치이다. 이때 프로세스는 background 혹은 foreground에서 실행될 수 있다는 것과, child process인지 parents process인지에 따라서도 다른 양상으로 실행된다는 것을 이용한다.

II. Code Explanation

1) `void eval(char *cmdline)`

이 함수는 사용자의 입력을 알맞게 파싱하고 주어진 command가 builtin command가 아닐 때 이를 foreground 혹은 background에서 실행시킨다.

```
    char *argv[MAXARGS]; /* argument list */
    char buf[MAXLINE]; /* modified command line */
    int bg; /* background flag */
    pid_t pid; /* process id */

    strcpy(buf, cmdline);
    bg = parseline(buf, argv);
    if(argv[0] == NULL) { /* ignore empty lines */
        return;
    }
```

argv는 argument list를 저장하는 변수이고, buf는 command를 받을 변수이며, bg는 background에서 실행되는 경우에 1을, foreground에서 실행될 경우에는 0을 저장하며, pid는 process id를 저장하는 변수이다. 먼저 cmdline을 복사한 buf를 알맞게 파싱하여 bg를 추출한다. 그리고 argv[0]을 살펴보며 빈 줄은 무시하도록 해주었다.

```
if(!builtin_cmd(argv)) {
```

```
        /* make sigset and block SIGCHLD*/
        sigset_t mask;
        if(sigemptyset(&mask) < 0) {
            unix_error("sigemptyset error");
        }
        if(sigaddset(&mask, SIGCHLD) < 0 ) {
            unix_error("sigaddset error");
        }
        if(sigprocmask(SIG_BLOCK, &mask, NULL) < 0) {
            unix_error("sigprocmask error");
        }
```

주어진 argv가 builtin command인지 확인하고, 아닌 경우에 해당 명령을 실행시켜줄 수 있도록 한다. sigemptyset, sigaddset, sigprocmask 함수를 호출하며 선언해준 mask를 세팅해준다. 각 과정에서 실패할 경우 에러메시지를 띄우도록 해주었다.

```
if((pid = fork()) < 0) { /* fork fails */
            unix_error("fork error");
        }
        if(pid == 0) { /* child runs user job */
            sigprocmask(SIG_UNBLOCK, &mask, NULL); /* unblock */
            setpgid(0, 0);
            if(execve(argv[0], argv, environ) < 0) { /* execute target program
*/
                printf("%s: Command not found.\n", argv[0]);
                exit(0);
            }
        }
        else { /* parents */
            addjob(jobs, pid, bg ? BG : FG, cmdline); /* add child to job list
*/
            sigprocmask(SIG_UNBLOCK, &mask, NULL);
            if(!bg) { /* runs in foreground */
                waitfg(pid);
            }
            else { /* runs in background */
                printf("[%d] (%d) %s", pid2jid(pid), pid, cmdline);
            }
        }
    }
    return;
```

본격적으로 명령을 실행하기 전 child process인지 parents process인지 구분하는 작업이 필요하다. fork()를 호출하여 pid의 값을 설정해주고 0보다 작은 경우에는 fork가 성공적으로 이루어지지 않았으므로 에러메시지를 표시한다. pid가 0이라면 child process이므로 앞서 block해두었던 것을 sigprocmask로 unblock해주고, setpgid(0, 0)을 통해 process group을 설정한다. execve를 호출하여 주어진 인자를 실행하는데, 만약 실패하면 command를 찾을 수 없다는 메시지를 띄우고 exit한다.

그 외의 경우는 parents process에 해당하게 된다. addjob 함수를 호출해 jobs 배열에 pid를 추가해주는 작업이 필요한데, background인지 foreground인지의 여부를 함께 저장하고, unblock 작업을 해준다. bg 변수에 저장된 값에 따라 foreground에서 실행하는 경우에는 waitfg를 이용해 job이 종료될 때까지 기다리게 하고, background에서 실행되는 경우에는 실행정보를 출력한다.

2) `int builtin_cmd(char **argv)`

builtin_cmd 함수는 주어진 명령이 builtin인지 아닌지를 확인하고, 만약 맞다면 1을, 아니라면 0을 리턴하도록 한다.

```
char *cmd = argv[0]; /* command */
    if(strcmp(cmd, "quit") == 0) {
        exit(0);
    }
    else if (strcmp(cmd, "fg") == 0 || strcmp(cmd, "bg") == 0) {
        do_bgfg(argv);
    }
    else if(strcmp(cmd, "jobs") == 0) {
        listjobs(jobs);
    }
    else { /* not builtin cmd */
        return 0;
    }
    return 1; /* builtin cmd */
```

cmd 변수를 선언하여 argv로부터 command를 받아와 저장하고, 이를 quit, fg, bg, jobs와 비교하여 각각 알맞은 함수를 호출하고 1을 반환한다. cmd가 quit인 경우에는 exit하여 쉘을 종료하고, cmd가 fg 혹은 bg인 경우에는 do_bgfg 함수를 호출한다. cmd가 jobs인 경우에는 listjobs를 호출하여 jobs 배열을 출력하도록 한다. 그 외의 경우는 builtin 명령이 아니므로 0을 반환하게 한다.

3) `void do_bgfg(char **argv)`

이 함수는 builtin command인 bg와 fg를 수행하는 함수로 bg인 경우에는 background job을 background에서 실행시키고 fg는 foreground에서 실행한다.

```
char *cmd = argv[0]; /* command */
    char *id = argv[1]; /* pid or jid */
    if(id == NULL) { /* no id */
        printf("%s command requires PID or %%jobid argument\n", cmd);
        return;
    }
```

선언한 변수로는 argv[0]로부터 얻은 command를 담는 cmd와 argv[1]로부터 얻은 process id 혹은 job id를 저장한다. id가 비어있는 경우는 에러 메시지를 출력하고 리턴하도록 하였다.

```
struct job_t* job;
    /* not pid nor jid */
    if(isdigit(id[0]) == 0 && id[0] != '%') {
        printf("%s: argument must be a PID or %%jobid\n", cmd);
        return;
    }
    /* pid */
    else if(isdigit(id[0])) {
        pid_t pid = (pid_t) atoi(id);
        job = getjobpid(jobs, pid);
        if(job == NULL) {
            printf("(%s): No such process\n", id);
            return;
        }
    }
    /* jid */
    else if(id[0] == '%') {
        int jid = atoi(&id[1]);
        job = getjobjid(jobs, jid);
        if(job == NULL) {
            printf("%s: No such job\n", id);
            return;
        }
    }
```

먼저 id가 pid와 jid가 모두 아닌 경우를 구분하여 에러메시지를 띄우고 리턴하도록 해주었다. 만약 pid에 해당하는 경우라면, atoi 함수와 type casting을 이용해 pid를 설정하고 getjobpid 함수를 호출하여 앞서 선언해주었던 job도 세팅한다. 만약 job이 없는 경우는 해당 process가 없음을 에러메시지로 출력하고 리턴하도록 하였다. jid는 '%'로 시작한다는 것을 이용하여 jid라고 판단한 경우, 마찬가지로 atoi 함수를 호출해 jid를 설정하고 이를 getjobjid의 인자로 넘겨주어 job을 찾도록 한다. 만약 job이 없으면 job이 없다는 에러메시지를 출력하고 리턴되게 한다.

```
/* change state and kill */
    if(strcmp(cmd, "bg") == 0) {
        job->state = BG;
        printf("[%d] (%d) %s", pid2jid(job->pid), job->pid, job->cmdline);
        kill(-(job->pid), SIGCONT);
    }
    else if(strcmp(cmd, "fg") == 0) {
        job->state = FG;
        kill(-(job->pid), SIGCONT);
        waitfg(job->pid);
    }
    return;
```

여태까지 리턴되지 않은 경우들은 무사히 해당 job을 찾았으므로 명령에 따라 알맞게 실행시켜주어야 한다. bg인 경우에는 찾은 job의 state를 BG로 설정해주고 상태를 출력한 후, 찾았던 job의

pid로 SIGCONT signal을 보내 kill하도록 요구한다. fg 명령의 경우에는 job의 state를 FG로 설정하고, job의 pid로 SIGCONT signal을 보내 kill하게 요구한 후, 해당 job이 종료될 때까지 기다리도록 waitfg를 호출한다.

4) void waitfg(pid_t pid)

waitfg는 foreground process가 종료될 때까지 기다리도록 하는 함수이다.

```
while((fgpid(jobs) == pid) && (getjobpid(jobs, pid)->state == FG)) {
        sleep(1);
    }
    return;
```

인자로 받은 pid가 foreground group의 id와 같은지를 비교하고, getjobpid 함수를 호출하여 pid로부터 job를 조사해 state가 FG인지 확인하여 만약 이 두가지를 모두 만족하는 동안 sleep을 수행한다.

5) void sigchld_handler(int sig)

이 함수는 커널이 쉘로 보낸 SIGCHLD signal을 핸들링하도록 하는 함수로 child job이 종료되어 zombie로 남으면 이를 알맞게 reap하고, SIGSTOP 혹은 SIGTSTP signal을 받아 정지한 경우에도 이를 알맞게 처리해주는 역할을 한다.

```
    pid_t pid;
    int status;
    while((pid = waitpid(-1, &status, WNOHANG | WUNTRACED)) > 0) {
        if(WIFEXITED(status)) {
            if(WTERMSIG(status)) {
                unix_error("waitpid error");
            }
            deletejob(jobs, pid);
        }
        else if(WIFSIGNALED(status)) {
            printf("Job [%d] (%d) terminated by signal %d\n", pid2jid(pid),
pid, WTERMSIG(status));
            deletejob(jobs, pid);
        }
        else if(WIFSTOPPED(status)) {
            getjobpid(jobs, pid)->state = ST;
            printf("Job [%d] (%d) stopped by signal %d\n", pid2jid(pid), pid,
WSTOPSIG(status));
        }
    }
    return;
```

변수 pid와 status를 선언해주어 process id와 process의 상태를 각각 저장할 수 있도록 한다. 먼저 모든 child process에 대해 확인해주기 위해 waitpid를 호출해주어 child process의 리턴값과 상태를 조사한다. 첫번째로 WIFEXITED, 즉 child process가 exit된 경우를 확인한다. 이때 child process를 종료시킨 signal이 0이 아니면 waitpid error가 발생했음을 출력할 수 있도록 해주고, jobs list에서 해당 pid의 job이 삭제되도록 한다. 두번째 경우는 WIFSIGNALED, 즉 child process가 어떤 signal을 받아 종료된 케이스로 해당 signal 내용을 WTERMSIG로 출력한 후 마찬가지로 해당 job을 삭제한다. 마지막으로 WIFSTOPPED, 즉 child process가 정지된 상태일 때는 getjobpid 함수를 호출하여 해당 job을 찾고 state를 ST로 바꿔준 후 WSTOPSIG를 통해 정지시킨 signal을 출력한다.

6) `void sigint_handler(int sig)`

사용자가 ctrl-c를 입력함으로써 커널이 보낸 SIGINT signal을 처리하는 함수이다. foreground job에 pid가 있는지를 fgpid로 확인하고 이를 SIGINT로 kill하도록 한다.

```
pid_t pid = fgpid(jobs);
    kill(-pid, SIGINT);
    return;
```

7) `void sigtstp_handler(int sig)`

사용자가 ctrl-z를 입력함으로써 커널이 보낸 SIGTSTP signal을 캐치하여 처리하는 함수이다. 위와 마찬가지로 fgpid로 foreground job을 확인하고 SIGTSTP로 kill할 것을 요구한다.

```
 pid_t pid = fgpid(jobs);
    kill(-pid, SIGTSTP);
    return;
```

III. Result

- trace01

  ./sdriver.pl -t trace01.txt -s ./tshref -a "-p"

  #

  # trace01.txt - Properly terminate on EOF.

  #

./sdriver.pl -t trace01.txt -s ./tsh -a "-p"

#

trace01.txt - Properly terminate on EOF.

#

- trace02

./sdriver.pl -t trace02.txt -s ./tshref -a "-p"

#

trace02.txt - Process builtin quit command.

#

./sdriver.pl -t trace02.txt -s ./tsh -a "-p"

#

trace02.txt - Process builtin quit command.

#

- trace03

./sdriver.pl -t trace03.txt -s ./tshref -a "-p"

#

trace03.txt - Run a foreground job.

#

tsh> quit

./sdriver.pl -t trace03.txt -s ./tsh -a "-p"

#

trace03.txt - Run a foreground job.

#

tsh> quit

- trace04

  ./sdriver.pl -t trace04.txt -s ./tshref -a "-p"

  #

  # trace04.txt - Run a background job.

  #

  tsh> ./myspin 1 &

  [1] (8085) ./myspin 1 &

  ./sdriver.pl -t trace04.txt -s ./tsh -a "-p"

  #

  # trace04.txt - Run a background job.

  #

  tsh> ./myspin 1 &

  [1] (8110) ./myspin 1 &

- trace05

  ./sdriver.pl -t trace05.txt -s ./tshref -a "-p"

  #

  # trace05.txt - Process jobs builtin command.

  #

  tsh> ./myspin 2 &

  [1] (8194) ./myspin 2 &

  tsh> ./myspin 3 &

  [2] (8196) ./myspin 3 &

  tsh> jobs

  [1] (8194) Running ./myspin 2 &

  [2] (8196) Running ./myspin 3 &

./sdriver.pl -t trace05.txt -s ./tsh -a "-p"

#

trace05.txt - Process jobs builtin command.

#

tsh> ./myspin 2 &

[1] (8229) ./myspin 2 &

tsh> ./myspin 3 &

[2] (8231) ./myspin 3 &

tsh> jobs

[1] (8229) Running ./myspin 2 &

[2] (8231) Running ./myspin 3 &

- trace06

  ./sdriver.pl -t trace06.txt -s ./tshref -a "-p"

  #

  # trace06.txt - Forward SIGINT to foreground job.

  #

  tsh> ./myspin 4

  Job [1] (8278) terminated by signal 2

  ./sdriver.pl -t trace06.txt -s ./tsh -a "-p"

  #

  # trace06.txt - Forward SIGINT to foreground job.

  #

  tsh> ./myspin 4

  Job [1] (8305) terminated by signal 2

- trace07

  ./sdriver.pl -t trace07.txt -s ./tshref -a "-p"

  #

  # trace07.txt - Forward SIGINT only to foreground job.

  #

  tsh> ./myspin 4 &

  [1] (8333) ./myspin 4 &

  tsh> ./myspin 5

  Job [2] (8335) terminated by signal 2

  tsh> jobs

  [1] (8333) Running ./myspin 4 &

  ./sdriver.pl -t trace07.txt -s ./tsh -a "-p"

  #

  # trace07.txt - Forward SIGINT only to foreground job.

  #

  tsh> ./myspin 4 &

  [1] (8367) ./myspin 4 &

  tsh> ./myspin 5

  Job [2] (8370) terminated by signal 2

  tsh> jobs

  [1] (8367) Running ./myspin 4 &

- trace08

  ./sdriver.pl -t trace08.txt -s ./tshref -a "-p"

  #

  # trace08.txt - Forward SIGTSTP only to foreground job.

#

tsh> ./myspin 4 &

[1] (8396) ./myspin 4 &

tsh> ./myspin 5

Job [2] (8398) stopped by signal 20

tsh> jobs

[1] (8396) Running ./myspin 4 &

[2] (8398) Stopped ./myspin 5

./sdriver.pl -t trace08.txt -s ./tsh -a "-p"

#

trace08.txt - Forward SIGTSTP only to foreground job.

#

tsh> ./myspin 4 &

[1] (8441) ./myspin 4 &

tsh> ./myspin 5

Job [2] (8443) stopped by signal 20

tsh> jobs

[1] (8441) Running ./myspin 4 &

[2] (8443) Stopped ./myspin 5

- trace09

./sdriver.pl -t trace09.txt -s ./tshref -a "-p"

#

trace09.txt - Process bg builtin command

#

tsh> ./myspin 4 &

[1] (8468) ./myspin 4 &

tsh> ./myspin 5

Job [2] (8470) stopped by signal 20

tsh> jobs

[1] (8468) Running ./myspin 4 &

[2] (8470) Stopped ./myspin 5

tsh> bg %2

[2] (8470) ./myspin 5

tsh> jobs

[1] (8468) Running ./myspin 4 &

[2] (8470) Running ./myspin 5

./sdriver.pl -t trace09.txt -s ./tsh -a "-p"

#

trace09.txt - Process bg builtin command

#

tsh> ./myspin 4 &

[1] (8512) ./myspin 4 &

tsh> ./myspin 5

Job [2] (8514) stopped by signal 20

tsh> jobs

[1] (8512) Running ./myspin 4 &

[2] (8514) Stopped ./myspin 5

tsh> bg %2

[2] (8514) ./myspin 5

tsh> jobs

[1] (8512) Running ./myspin 4 &

[2] (8514) Running ./myspin 5

- trace10

./sdriver.pl -t trace10.txt -s ./tshref -a "-p"

#

trace10.txt - Process fg builtin command.

#

tsh> ./myspin 4 &

[1] (8576) ./myspin 4 &

tsh> fg %1

Job [1] (8576) stopped by signal 20

tsh> jobs

[1] (8576) Stopped ./myspin 4 &

tsh> fg %1

tsh> jobs

./sdriver.pl -t trace10.txt -s ./tsh -a "-p"

#

trace10.txt - Process fg builtin command.

#

tsh> ./myspin 4 &

[1] (8624) ./myspin 4 &

tsh> fg %1

Job [1] (8624) stopped by signal 20

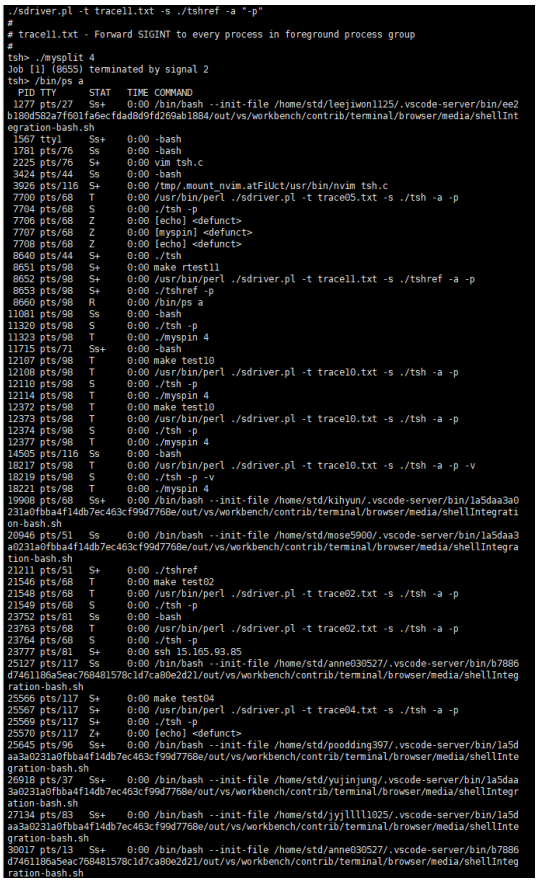
tsh> jobs

[1] (8624) Stopped ./myspin 4 &

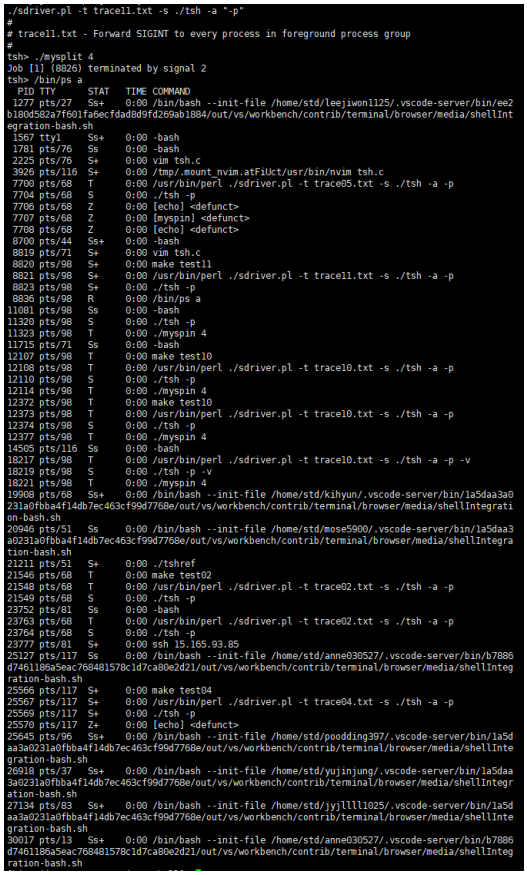
tsh> fg %1

tsh> jobs

- trace11 (다른 수강생들에 대한 정보가 뜨는데 이유가 무엇인지 모르겠습니다.)



```
./sdriver.pl -t trace11.txt -s ./tshref -a "-p"
#
# trace11.txt - Forward SIGINT to every process in foreground process group
#
tsh> ./mysplit 4
Job [1] (8655) terminated by signal 2
tsh> /bin/ps a
  PID TTY      STAT   TIME COMMAND
 1277 pts/27   Ss+    0:00 /bin/bash --init-file /home/std/leejiwon1125/.vscode-server/bin/ee2
b180d582a7f601fa6ecfdad8d9fd269ab1884/out/vs/workbench/contrib/terminal/browser/media/shellInt
egration-bash.sh
 1567 tty1     Ss+    0:00 -bash
 1781 pts/76   Ss     0:00 -bash
 2225 pts/76   S+     0:00 vim tsh.c
 3424 pts/44   Ss     0:00 -bash
 3926 pts/116  S+     0:00 /tmp/.mount_nvim.atFiUct/usr/bin/nvim tsh.c
 7700 pts/68   T      0:00 /usr/bin/perl ./sdriver.pl -t trace05.txt -s ./tsh -a -p
 7704 pts/68   S      0:00 ./tsh -p
 7706 pts/68   Z      0:00 [echo] <defunct>
 7707 pts/68   Z      0:00 [myspin] <defunct>
 7708 pts/68   Z      0:00 [echo] <defunct>
 8640 pts/44   S+     0:00 ./tsh
 8651 pts/98   S+     0:00 make rtest11
 8652 pts/98   S+     0:00 /usr/bin/perl ./sdriver.pl -t trace11.txt -s ./tshref -a -p
 8653 pts/98   S+     0:00 ./tshref -p
 8660 pts/98   R      0:00 /bin/ps a
11081 pts/98   Ss     0:00 -bash
11320 pts/98   S      0:00 ./tsh -p
11323 pts/98   T      0:00 ./myspin 4
11715 pts/71   Ss+    0:00 -bash
12107 pts/98   T      0:00 make test10
12108 pts/98   T      0:00 /usr/bin/perl ./sdriver.pl -t trace10.txt -s ./tsh -a -p
12110 pts/98   S      0:00 ./tsh -p
12114 pts/98   T      0:00 ./myspin 4
12372 pts/98   T      0:00 make test10
12373 pts/98   T      0:00 /usr/bin/perl ./sdriver.pl -t trace10.txt -s ./tsh -a -p
12374 pts/98   S      0:00 ./tsh -p
12377 pts/98   T      0:00 ./myspin 4
14505 pts/116  Ss     0:00 -bash
18217 pts/98   T      0:00 /usr/bin/perl ./sdriver.pl -t trace10.txt -s ./tsh -a -p -v
18219 pts/98   S      0:00 ./tsh -p -v
18221 pts/98   T      0:00 ./myspin 4
19908 pts/68   Ss+    0:00 /bin/bash --init-file /home/std/kihyun/.vscode-server/bin/1a5daa3a0
231a0fbba4f14db7ec463cf99d7768e/out/vs/workbench/contrib/terminal/browser/media/shellIntegrati
on-bash.sh
20946 pts/51   Ss     0:00 /bin/bash --init-file /home/std/mose5900/.vscode-server/bin/1a5daa3
a0231a0fbba4f14db7ec463cf99d7768e/out/vs/workbench/contrib/terminal/browser/media/shellIntegra
tion-bash.sh
21211 pts/51   S+     0:00 ./tshref
21546 pts/68   T      0:00 make test02
21548 pts/68   T      0:00 /usr/bin/perl ./sdriver.pl -t trace02.txt -s ./tsh -a -p
21549 pts/68   S      0:00 ./tsh -p
23752 pts/81   Ss     0:00 -bash
23763 pts/68   T      0:00 /usr/bin/perl ./sdriver.pl -t trace02.txt -s ./tsh -a -p
23764 pts/68   S      0:00 ./tsh -p
23777 pts/81   S+     0:00 ssh 15.165.93.85
25127 pts/117  Ss     0:00 /bin/bash --init-file /home/std/anne030527/.vscode-server/bin/b7886
d7461186a5eac768481578c1d7ca80e2d21/out/vs/workbench/contrib/terminal/browser/media/shellInteg
ration-bash.sh
25566 pts/117  S+     0:00 make test04
25567 pts/117  S+     0:00 /usr/bin/perl ./sdriver.pl -t trace04.txt -s ./tsh -a -p
25569 pts/117  S+     0:00 ./tsh -p
25570 pts/117  Z+     0:00 [echo] <defunct>
25645 pts/96   Ss+    0:00 /bin/bash --init-file /home/std/poodding397/.vscode-server/bin/1a5d
aa3a0231a0fbba4f14db7ec463cf99d7768e/out/vs/workbench/contrib/terminal/browser/media/shellInte
gration-bash.sh
26918 pts/37   Ss+    0:00 /bin/bash --init-file /home/std/yujinjung/.vscode-server/bin/1a5daa
3a0231a0fbba4f14db7ec463cf99d7768e/out/vs/workbench/contrib/terminal/browser/media/shellIntegr
ation-bash.sh
27134 pts/83   Ss+    0:00 /bin/bash --init-file /home/std/jyjllll1025/.vscode-server/bin/1a5d
aa3a0231a0fbba4f14db7ec463cf99d7768e/out/vs/workbench/contrib/terminal/browser/media/shellInte
gration-bash.sh
30017 pts/13   Ss+    0:00 /bin/bash --init-file /home/std/anne030527/.vscode-server/bin/b7886
d7461186a5eac768481578c1d7ca80e2d21/out/vs/workbench/contrib/terminal/browser/media/shellInteg
ration-bash.sh
```



```
./sdriver.pl -t trace11.txt -s ./tsh -a "-p"
#
# trace11.txt - Forward SIGINT to every process in foreground process group
#
tsh> ./mysplit 4
Job [1] (8826) terminated by signal 2
tsh> /bin/ps a
  PID TTY      STAT   TIME COMMAND
 1277 pts/27   Ss+    0:00 /bin/bash --init-file /home/std/leejiwon1125/.vscode-server/bin/ee2
b180d582a7f601fa6ecfdad8d9fd269ab1884/out/vs/workbench/contrib/terminal/browser/media/shellInt
egration-bash.sh
 1567 tty1     Ss+    0:00 -bash
 1781 pts/76   Ss     0:00 -bash
 2225 pts/76   S+     0:00 vim tsh.c
 3926 pts/116  S+     0:00 /tmp/.mount_nvim.atFiUct/usr/bin/nvim tsh.c
 7700 pts/68   T      0:00 /usr/bin/perl ./sdriver.pl -t trace05.txt -s ./tsh -a -p
 7704 pts/68   S      0:00 ./tsh -p
 7706 pts/68   Z      0:00 [echo] <defunct>
 7707 pts/68   Z      0:00 [myspin] <defunct>
 7708 pts/68   Z      0:00 [echo] <defunct>
 8700 pts/44   Ss+    0:00 -bash
 8819 pts/71   S+     0:00 vim tsh.c
 8820 pts/98   S+     0:00 make test11
 8821 pts/98   S+     0:00 /usr/bin/perl ./sdriver.pl -t trace11.txt -s ./tsh -a -p
 8823 pts/98   S+     0:00 ./tsh -p
 8836 pts/98   R      0:00 /bin/ps a
11081 pts/98   Ss     0:00 -bash
11320 pts/98   S      0:00 ./tsh -p
11323 pts/98   T      0:00 ./myspin 4
11715 pts/71   Ss     0:00 -bash
12107 pts/98   T      0:00 make test10
12108 pts/98   T      0:00 /usr/bin/perl ./sdriver.pl -t trace10.txt -s ./tsh -a -p
12110 pts/98   S      0:00 ./tsh -p
12114 pts/98   T      0:00 ./myspin 4
12372 pts/98   T      0:00 make test10
12373 pts/98   T      0:00 /usr/bin/perl ./sdriver.pl -t trace10.txt -s ./tsh -a -p
12374 pts/98   S      0:00 ./tsh -p
12377 pts/98   T      0:00 ./myspin 4
14505 pts/116  Ss     0:00 -bash
18217 pts/98   T      0:00 /usr/bin/perl ./sdriver.pl -t trace10.txt -s ./tsh -a -p -v
18219 pts/98   S      0:00 ./tsh -p -v
18221 pts/98   T      0:00 ./myspin 4
19908 pts/68   Ss+    0:00 /bin/bash --init-file /home/std/kihyun/.vscode-server/bin/1a5daa3a0
231a0fbba4f14db7ec463cf99d7768e/out/vs/workbench/contrib/terminal/browser/media/shellIntegrati
on-bash.sh
20946 pts/51   Ss     0:00 /bin/bash --init-file /home/std/mose5900/.vscode-server/bin/1a5daa3
a0231a0fbba4f14db7ec463cf99d7768e/out/vs/workbench/contrib/terminal/browser/media/shellIntegra
tion-bash.sh
21211 pts/51   S+     0:00 ./tshref
21546 pts/68   T      0:00 make test02
21548 pts/68   T      0:00 /usr/bin/perl ./sdriver.pl -t trace02.txt -s ./tsh -a -p
21549 pts/68   S      0:00 ./tsh -p
23752 pts/81   Ss     0:00 -bash
23763 pts/68   T      0:00 /usr/bin/perl ./sdriver.pl -t trace02.txt -s ./tsh -a -p
23764 pts/68   S      0:00 ./tsh -p
23777 pts/81   S+     0:00 ssh 15.165.93.85
25127 pts/117  Ss     0:00 /bin/bash --init-file /home/std/anne030527/.vscode-server/bin/b7886
d7461186a5eac768481578c1d7ca80e2d21/out/vs/workbench/contrib/terminal/browser/media/shellInteg
ration-bash.sh
25566 pts/117  S+     0:00 make test04
25567 pts/117  S+     0:00 /usr/bin/perl ./sdriver.pl -t trace04.txt -s ./tsh -a -p
25569 pts/117  S+     0:00 ./tsh -p
25570 pts/117  Z+     0:00 [echo] <defunct>
25645 pts/96   Ss+    0:00 /bin/bash --init-file /home/std/poodding397/.vscode-server/bin/1a5d
aa3a0231a0fbba4f14db7ec463cf99d7768e/out/vs/workbench/contrib/terminal/browser/media/shellInte
gration-bash.sh
26918 pts/37   Ss+    0:00 /bin/bash --init-file /home/std/yujinjung/.vscode-server/bin/1a5daa
3a0231a0fbba4f14db7ec463cf99d7768e/out/vs/workbench/contrib/terminal/browser/media/shellIntegr
ation-bash.sh
27134 pts/83   Ss+    0:00 /bin/bash --init-file /home/std/jyjllll1025/.vscode-server/bin/1a5d
aa3a0231a0fbba4f14db7ec463cf99d7768e/out/vs/workbench/contrib/terminal/browser/media/shellInte
gration-bash.sh
30017 pts/13   Ss+    0:00 /bin/bash --init-file /home/std/anne030527/.vscode-server/bin/b7886
d7461186a5eac768481578c1d7ca80e2d21/out/vs/workbench/contrib/terminal/browser/media/shellInteg
ration-bash.sh
```

- trace12, trace13

  (trace11와 마찬가지로 불필요한 정보를 포함한 채로 너무 길어져 별도로 첨부하지는 않겠습니다. 그러나 reference shell과 유사하게 작동하고 있음을 확인할 수 있었습니다.)

- trace14

  ./sdriver.pl -t trace14.txt -s ./tshref -a "-p"

  #

  # trace14.txt - Simple error handling

  #

  tsh> ./bogus

  ./bogus: Command not found

```
tsh> ./myspin 4 &
[1] (9344) ./myspin 4 &
tsh> fg
fg command requires PID or %jobid argument
tsh> bg
bg command requires PID or %jobid argument
tsh> fg a
fg: argument must be a PID or %jobid
tsh> bg a
bg: argument must be a PID or %jobid
tsh> fg 9999999
(9999999): No such process
tsh> bg 9999999
(9999999): No such process
tsh> fg %2
%2: No such job
tsh> fg %1
Job [1] (9344) stopped by signal 20
tsh> bg %2
%2: No such job
tsh> bg %1
[1] (9344) ./myspin 4 &
tsh> jobs
[1] (9344) Running ./myspin 4 &
```

```
./sdriver.pl -t trace14.txt -s ./tsh -a "-p"
```

```
#
# trace14.txt - Simple error handling
#
tsh> ./bogus
./bogus: Command not found.
tsh> ./myspin 4 &
[1] (9394) ./myspin 4 &
tsh> fg
fg command requires PID or %jobid argument
tsh> bg
bg command requires PID or %jobid argument
tsh> fg a
fg: argument must be a PID or %jobid
tsh> bg a
bg: argument must be a PID or %jobid
tsh> fg 9999999
(9999999): No such process
tsh> bg 9999999
(9999999): No such process
tsh> fg %2
%2: No such job
tsh> fg %1
Job [1] (9394) stopped by signal 20
tsh> bg %2
%2: No such job
tsh> bg %1
```

[1] (9394) ./myspin 4 &

tsh> jobs

[1] (9394) Running ./myspin 4 &

- trace15

  ./sdriver.pl -t trace15.txt -s ./tshref -a "-p"

  #

  # trace15.txt - Putting it all together

  #

  tsh> ./bogus

  ./bogus: Command not found

  tsh> ./myspin 10

  Job [1] (9492) terminated by signal 2

  tsh> ./myspin 3 &

  [1] (9495) ./myspin 3 &

  tsh> ./myspin 4 &

  [2] (9497) ./myspin 4 &

  tsh> jobs

  [1] (9495) Running ./myspin 3 &

  [2] (9497) Running ./myspin 4 &

  tsh> fg %1

  Job [1] (9495) stopped by signal 20

  tsh> jobs

  [1] (9495) Stopped ./myspin 3 &

  [2] (9497) Running ./myspin 4 &

  tsh> bg %3

  %3: No such job

```
tsh> bg %1
[1] (9495) ./myspin 3 &
tsh> jobs
[1] (9495) Running ./myspin 3 &
[2] (9497) Running ./myspin 4 &
tsh> fg %1
tsh> quit
```

```
./sdriver.pl -t trace15.txt -s ./tsh -a "-p"
#
# trace15.txt - Putting it all together
#
tsh> ./bogus
./bogus: Command not found.
tsh> ./myspin 10
Job [1] (9534) terminated by signal 2
tsh> ./myspin 3 &
[1] (9537) ./myspin 3 &
tsh> ./myspin 4 &
[2] (9540) ./myspin 4 &
tsh> jobs
[1] (9537) Running ./myspin 3 &
[2] (9540) Running ./myspin 4 &
tsh> fg %1
Job [1] (9537) stopped by signal 20
tsh> jobs
```

[1] (9537) Stopped ./myspin 3 &

[2] (9540) Running ./myspin 4 &

tsh> bg %3

%3: No such job

tsh> bg %1

[1] (9537) ./myspin 3 &

tsh> jobs

[1] (9537) Running ./myspin 3 &

[2] (9540) Running ./myspin 4 &

tsh> fg %1

tsh> quit

- trace16

./sdriver.pl -t trace16.txt -s ./tshref -a "-p"

#

trace16.txt - Tests whether the shell can handle SIGTSTP and SIGINT

signals that come from other processes instead of the terminal.

#

tsh> ./mystop 2

Job [1] (9585) stopped by signal 20

tsh> jobs

[1] (9585) Stopped ./mystop 2

tsh> ./myint 2

Job [2] (9594) terminated by signal 2

./sdriver.pl -t trace16.txt -s ./tsh -a "-p"

#

trace16.txt - Tests whether the shell can handle SIGTSTP and SIGINT

signals that come from other processes instead of the terminal.

#

tsh> ./mystop 2

Job [1] (9620) stopped by signal 20

tsh> jobs

[1] (9620) Stopped ./mystop 2

tsh> ./myint 2

Job [2] (9626) terminated by signal 2