



CS 361 – Lab 6

Foreground, Background

MONDAY, MARCH 4TH, 2019.

Foreground

- ▶ by default, processes are started in the foreground
- ▶ CTRL-Z sends the SIGTSTP signal to the foreground process – tells it to pause the execution of the command and return control to the terminal
- ▶ can resume execution of the command in the foreground again by typing fg

```
$ fg
```

Background

- ▶ background processes return control to the shell immediately without waiting for the process to complete
- ▶ User able able to start processes directly in the background using "&"

```
$ pwd &
```

- ▶ to see all stopped or backgrounded processes, type jobs command

```
$ jobs
```

setpgid(), getpgid()

- ▶ `int setpgid(pid_t pid, pid_t pgid)`
- ▶ sets the process group ID of the process *pid* to *pgid*
- ▶ `pid_t getpgid(pid_t pid);`
- ▶ returns the process group ID of the process specified by *pid*

tcsetpgrp(), tcgetpgrp()

- ▶ to get and set terminal foreground process group
- ▶ `int tcsetpgrp(int fildes, pid_t newid);`
- ▶ `tcsetpgrp` sets the foreground process group ID to `newid` based on the file descriptor `fildes` which is connected to terminal
- ▶ `pid_t tcgetpgrp(int fd);`

```
pid = fork();
if(pid == 0)
{
    pid = getpid() // pid is initialized to zero in the Fork
    setpgid(pid, pid);
    tcsetpgrp(STDIN_FILENO, pid); // And now you're under control
    // And then you do what they told you.
    exit(0);
}
if(pid < 0)
{
    perror("fork");
    exit(1);
}
setpgid(pid, pid);
```

Task

- ▶ 1. Finish the code(lab6_samp.c on github) so that control is returned to the main process before exiting.
- ▶ 3. Why must we block SIGTTIN and SIGTTOU to be able to transfer control?
 - Try removing the signal handler registers and see what it does.
- ▶ 2. Modify the code to take an input parameter to send a job to the background. Send a job to the background.