(/)

Curriculum

SE Foundations Average: 108.76%

0x05. Processes and signals

DevOps Shell Bash Syscall Scripting

By: Sylvain Kalache

🌣 Weight: 1

Project over - took place from Jun 30, 2023 6:00 AM to Jul 1, 2023 6:00 AM

An auto review will be launched at the deadline

In a nutshell...

Auto QA review: 16.25/26 mandatory & 0.0/17 optional

• Altogether: 62.5%

Mandatory: 62.5%Optional: 0.0%

Calculation: 62.5% + (62.5% * 0.0%) == 62.5%

About Bash projects

Unless stated, all your projects will be auto-corrected with Ubuntu 20.04 LTS.

Resources

Read or watch:

- Linux PID (/rltoken/qVGxUt1QMIV4B4oVrQBlQg)
- Linux process (/rltoken/px2TdWSjVO8i9SB5gHchAw)
- Linux signal (/rltoken/qQSGz9CN52PVF3IPCuaRiw)
- Process management in linux (/rltoken/XIYrlghzNZ6Z1cbl_IPaiA)

man or help:

- ps
- pgrep



Help



- kill
- exit
- trap

Learning Objectives

At the end of this project, you are expected to be able to explain to anyone (/rltoken/_zeQBWHdINNOM-5lqFDhSQ), without the help of Google:

General

- · What is a PID
- What is a process
- How to find a process' PID
- How to kill a process
- · What is a signal
- What are the 2 signals that cannot be ignored

Copyright - Plagiarism

- You are tasked to come up with solutions for the tasks below yourself to meet with the above learning objectives.
- You will not be able to meet the objectives of this or any following project by copying and pasting someone else's work.
- You are not allowed to publish any content of this project.
- Any form of plagiarism is strictly forbidden and will result in removal from the program.

Requirements

General

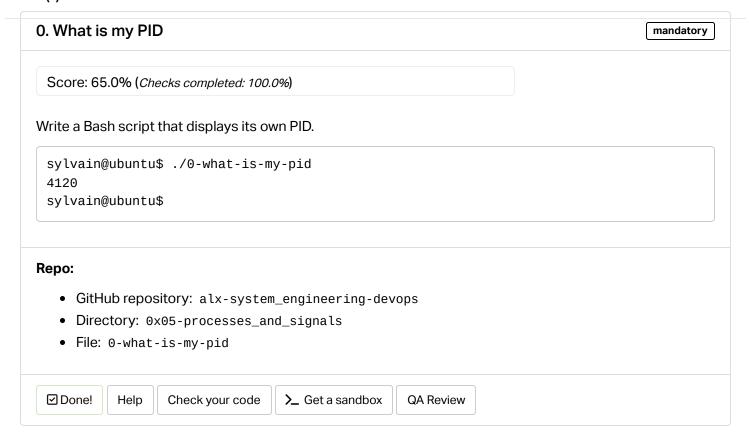
- Allowed editors: vi, vim, emacs
- All your files will be interpreted on Ubuntu 20.04 LTS
- All your files should end with a new line
- A README.md file, at the root of the folder of the project, is mandatory
- All your Bash script files must be executable
- Your Bash script must pass Shellcheck (version 0.7.0 via apt-get) without any error
- The first line of all your Bash scripts should be exactly #!/usr/bin/env bash
- The second line of all your Bash scripts should be a comment explaining what is the script doing

More Info

For those who want to know more and learn about all signals, check out this article (/rltoken/BOU-KVNMqfKEIBo VOI26A).



Taşks



1. List your processes

mandatory

Score: 65.0% (Checks completed: 100.0%)

Write a Bash script that displays a list of currently running processes.

Requirements:

- . Must show all processes, for all users, including those which might not have a TTY
- Display in a user-oriented format
- Show process hierarchy

Q

β γlvain@u	ıbuntu\$	6 ./1-	-list_	_yourpr	ocesse	es	head	-50			
USER		%CPU		VSZ	RSS			STAT	START	TIME	COMMAND
root	2	0.0	0.0	0	0	?		S	Feb13	0:00	[kthreadd]
root	3	0.0	0.0	0	0	?		S	Feb13	0:00	<pre>_ [ksoftirqd/0]</pre>
root	4	0.0	0.0	0	Θ	?		S	Feb13	0:00	_ [kworker/0:0]
root	5	0.0	0.0	0	0	?		S<	Feb13	0:00	_ [kworker/0:0H]
root	7	0.0	0.0	0	0	?		S	Feb13	0:02	_ [rcu_sched]
root	8	0.0	0.0	0	Θ	?		S	Feb13	0:03	_ [rcuos/0]
root	9	0.0	0.0	0	Θ	?		S	Feb13	0:00	_ [rcu_bh]
root	10	0.0	0.0	0	Θ	?		S	Feb13	0:00	_ [rcuob/0]
root	11	0.0	0.0	0	Θ	?		S	Feb13	0:00	$\ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ $
root	12	0.0	0.0	0	Θ	?		S	Feb13	0:02	_ [watchdog/0]
root	13	0.0	0.0	0	Θ	?		S<	Feb13	0:00	_ [khelper]
root	14	0.0	0.0	0	Θ	?		S	Feb13	0:00	<pre>_ [kdevtmpfs]</pre>
root	15	0.0	0.0	0	Θ	?		S<	Feb13	0:00	_ [netns]
root	16	0.0	0.0	0	0	?		S<	Feb13	0:00	_ [writeback]
root	17	0.0	0.0	0	Θ	?		S<	Feb13	0:00	_ [kintegrityd]
root	18	0.0	0.0	0	Θ	?		S<	Feb13	0:00	_ [bioset]
root	19	0.0	0.0	0	Θ	?		S<	Feb13	0:00	_ [kworker/u3:0]
root	20	0.0	0.0	0	Θ	?		S<	Feb13	0:00	_ [kblockd]
root	21	0.0	0.0	0	Θ	?		S<	Feb13	0:00	_ [ata_sff]
root	22	0.0	0.0	0	Θ	?		S	Feb13	0:00	
root	23	0.0	0.0	0	Θ	?		S<	Feb13	0:00	_ [md]
root	24	0.0	0.0	0	0	?		S<	Feb13	0:00	_ [devfreq_wq]
root	25	0.0	0.0	0	Θ	?		S	Feb13	0:41	_ [kworker/0:1]
root	27	0.0	0.0	0	0	?		S	Feb13	0:00	_ [khungtaskd]
root	28	0.0	0.0	0	0			S	Feb13	0:00	_ [kswapd0]
root	29	0.0	0.0	0	0	?		S<	Feb13	0:00	_ [vmstat]
root	30	0.0	0.0	0	Θ	?		SN	Feb13	0:00	_ [ksmd]
root	31	0.0	0.0	0	0	?		S	Feb13	0:00	_ [fsnotify_mark]
root	32	0.0	0.0	0	0	?		S	Feb13	0:00	_ [ecryptfs-kthre
a]											_ []
root	33	0.0	0.0	Θ	Θ	?		S<	Feb13	0:00	_ [crypto]
root	45	0.0	0.0	0		?		S<	Feb13	0:00	_ [kthrotld]
root	46	0.0	0.0	0	0			S	Feb13	0:00	_ [kworker/u2:1]
root	65	0.0	0.0	0		?		S<	Feb13	0:00	_ [deferwq]
root	66	0.0	0.0	0		?		S<	Feb13	0:00	_ [charger_manage
r]				-	_						_ [
root	108	0.0	0.0	Θ	Θ	?		S<	Feb13	0:00	_ [kpsmoused]
root	125	0.0	0.0	0		?		S	Feb13	0:00	_ [scsi_eh_0]
root	126	0.0	0.0	0		?		S	Feb13	0:00	_ [kworker/u2:2]
root	172	0.0	0.0	0		?		S	Feb13	0:00	_ [jbd2/sda1-8]
root	173	0.0	0.0	0		?		S<	Feb13	0:00	_ [ext4-rsv-conve
r]	•			•	J			-		2.00	_ [: ::::::::::::::::::::::::::::::::::
root	409	0.0	0.0	0	0	?		S<	Feb13	0:00	_ [iprt]
root	549	0.0	0.0	0		?		S<	Feb13	0:00	_ [kworker/u3:1]
root	808	0.0	0.0	0		?		S	Feb13	0:00	_ [kauditd]
root	834	0.0	0.0	0		?		S<	Feb13	0:00	_ [rpciod]
root	846	0.0	0.0	0		?		S<	Feb13	0:00	_ [nfsiod]
root	1	0.0	0.4	33608	2168			Ss	Feb13		/sbin/init
root	373	0.0	0.0	19472	408			S	Feb13		upstart-udev-bridge
daemon	010	5.0	5.0	10712	700	•		J	, 5010	5.00	apocare duce bi tuge
uaciiion											

root 378 0.0 0.2 49904 1088 ? Ss Feb13 0:00 /lib/systemd/system (d) udevd --daemon 518 0.0 644 ? Ss Feb13 0:00 rpcbind root 0.1 23416 statd 547 0.0 0.1 21536 852 ? Ss Feb13 0:00 rpc.statd -L sylvain@ubuntu\$

Repo:

- GitHub repository: alx-system_engineering-devops
- Directory: 0x05-processes_and_signals
- File: 1-list_your_processes



2. Show your Bash PID

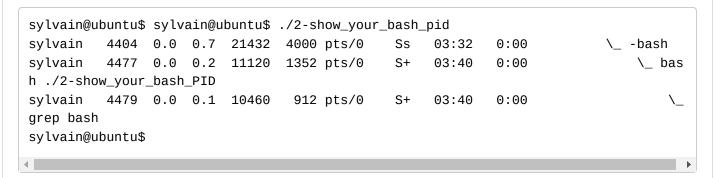
mandatory

Score: 65.0% (Checks completed: 100.0%)

Using your previous exercise command, write a Bash script that displays lines containing the bash word, thus allowing you to easily get the PID of your Bash process.

Requirements:

- You cannot use pgrep
- The third line of your script must be # shellcheck disable=SC2009 (for more info about ignoring shellcheck error here (/rltoken/vErRT8QGU2bwJ6FLvPLzxw))



Here we can see that my Bash PID is 4404.

Repo:

- GitHub repository: alx-system_engineering-devops
- Directory: 0x05-processes_and_signals
- File: 2-show_your_bash_pid



3(Show your Bash PID made easy

mandatory

Score: 65.0% (Checks completed: 100.0%)

Write a Bash script that displays the PID, along with the process name, of processes whose name contain the word bash.

Requirements:

• You cannot use ps

```
sylvain@ubuntu$ ./3-show_your_bash_pid_made_easy
4404 bash
4555 bash
sylvain@ubuntu$ ./3-show_your_bash_pid_made_easy
4404 bash
4557 bash
sylvain@ubuntu$
```

Here we can see that:

- For the first iteration: bash PID is 4404 and that the 3-show_your_bash_pid_made_easy script PID is 4555
- For the second iteration: bash PID is 4404 and that the 3-show_your_bash_pid_made_easy script PID is 4557

Repo:

- GitHub repository: alx-system_engineering-devops
- Directory: 0x05-processes_and_signals
- File: 3-show_your_bash_pid_made_easy

4. To infinity and beyond

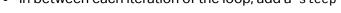
mandatory

Score: 65.0% (Checks completed: 100.0%)

Write a Bash script that displays To infinity and beyond indefinitely.

Requirements:

• In between each iteration of the loop, add a sleep 2

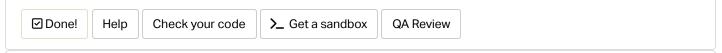


Sylvain@ubuntu\$./4-to_infinity_and_beyond
To infinity and beyond
Sylvain@ubuntu\$

Note that I ctrl+c (killed) the Bash script in the example.

Repo:

- GitHub repository: alx-system_engineering-devops
- Directory: 0x05-processes_and_signals
- File: 4-to_infinity_and_beyond



5. Don't stop me now!

mandatory

Score: 65.0% (Checks completed: 100.0%)

We stopped our 4-to_infinity_and_beyond process using ctrl+c in the previous task, there is actually another way to do this.

Write a Bash script that stops 4-to_infinity_and_beyond process.

Requirements:

• You must use kill

Terminal #0

Q

```
Sylvain@ubuntu$ ./4-to_infinity_and_beyond to infinity and beyond
Terminated
sylvain@ubuntu$
```

Terminal #1

```
sylvain@ubuntu$ ./5-dont_stop_me_now
sylvain@ubuntu$
```

I opened 2 terminals in this example, started by running my 4-to_infinity_and_beyond Bash script in terminal #0 and then moved on terminal #1 to run 5-dont_stop_me_now . We can then see in terminal #0 that my process has been terminated.

Repo:

- GitHub repository: alx-system_engineering-devops
- Directory: 0x05-processes_and_signals
- File: 5-dont_stop_me_now

6. Stop me if you can

mandatory

Score: 65.0% (Checks completed: 100.0%)

Write a Bash script that stops 4-to_infinity_and_beyond process.

Requirements:

• You cannot use kill or killall

Q

Terminal #0

```
Aylvain@ubuntu$ ./4-to_infinity_and_beyond
To infinity and beyond
Terminated
sylvain@ubuntu$
```

Terminal #1

```
sylvain@ubuntu$ ./6-stop_me_if_you_can
sylvain@ubuntu$
```

I opened 2 terminals in this example, started by running my 4-to_infinity_and_beyond Bash script in terminal #0 and then moved on terminal #1 to run 6-stop_me_if_you_can . We can then see in terminal #0 that my process has been terminated.

Repo:

- GitHub repository: alx-system_engineering-devops
- Directory: 0x05-processes_and_signals
- File: 6-stop_me_if_you_can

7. Highlander

mandatory

Score: 32.5% (Checks completed: 50.0%)

Write a Bash script that displays:

- To infinity and beyond indefinitely
- With a sleep 2 in between each iteration
- I am invincible!!! when receiving a SIGTERM signal

Make a copy of your 6-stop_me_if_you_can script, name it 67-stop_me_if_you_can, that kills the 7-highlander process instead of the 4-to_infinity_and_beyond one.

Terminal #0

```
Sylvain@ubuntu$ ./7-highlander to infinity and beyond
 To infinity and beyond
 I am invincible!!!
 To infinity and beyond
 I am invincible!!!
 To infinity and beyond
 To infinity and beyond
 To infinity and beyond
 I am invincible!!!
 To infinity and beyond
 ^C
 sylvain@ubuntu$
Terminal #1
 sylvain@ubuntu$ ./67-stop_me_if_you_can
 sylvain@ubuntu$ ./67-stop_me_if_you_can
 sylvain@ubuntu$ ./67-stop_me_if_you_can
 sylvain@ubuntu$
Istarted 7-highlander in Terminal #0 and then run 67-stop_me_if_you_can in terminal #1, for every
iteration we can see I am invincible!!! appearing in terminal #0.
Repo:
   • GitHub repository: alx-system_engineering-devops
   • Directory: 0x05-processes_and_signals
   • File: 7-highlander
                   Check your code
 ☐ Done?
            Help
                                    Ask for a new correction
                                                          >_ Get a sandbox
                                                                            QA Review
8. Beheaded process
                                                                                        mandatory
 Score: 65.0% (Checks completed: 100.0%)
Write a Bash script that kills the process 7-highlander.
Terminal #0
 sylvain@ubuntu$ ./7-highlander
 To infinity and beyond
 To infinity and beyond
 To infinity and beyond
 To infinity and beyond
 Killed
 sylvain@ubuntu$
```

Terminal #1											
sylvain@ubuntu\$./8-beheaded_process sylvain@ubuntu\$											
I started 7-highlander in Terminal #0 and then run 8-beheaded_process in terminal #1 and we can see that the 7-highlander has been killed.											
Repo:											
 GitHub repository: alx-system_engineering-devops 											
Directory: 0x05-processes_and_signals											
File: 8-beheaded_process											
☑ Done! Help Check your code											

Done with the mandatory tasks? Unlock 3 advanced tasks now!

Copyright © 2024 ALX, All rights reserved.

Q