Bash Background Process

```
# Starting a process in the background and bringing it back to the foreground
$ sleep 1000 &
[1] 25867
$ fg
sleep 1000


# Disowning a process
$ sleep 1000 &
[1] 26090
$ disown %1
$


$ sleep 1000 &
[1] 26214
$ disown %1
$ ps -ef | grep sleep | grep -v grep
roel    26214 26120  0 13:13 pts/3    00:00:00 sleep 1000
$ exit
# Then, opening a new shell and re-executing
$ ps -ef | grep sleep | grep -v grep
roel    26214    1  0 19:48 ?      00:00:00 sleep 1000


# Placing a command into the background
$ sleep 1000
^Z
[1]+  Stopped            sleep 1000
$ bg %1
[1]+ sleep 1000 &
$


$ sleep 1000
^Z
[1]+  Stopped            sleep 1000
$ bg %1
[1]+ sleep 1000 &
```

```
$ disown %1
$

# Multiple background processes and terminating processes
$ sleep 1000 &
[1] 27158
$ sleep 1000 &
[2] 27159
# We can see here that two background processes ([1] and [2], with PID's 27158 and 27159 respectively) were
started.
# Next, we kill the first process:
$ kill %1
$
[1]-  Terminated          sleep 1000
$

# One done before the other
$ sleep 1000 &
[1] 27406
$ sleep 3 &
[2] 27407
$
# After about 5 seconds, pressing enter, we will see
$
[2]+  Done             sleep 3
# What will happen now if we use fg in this case without the original [1] specifier?
$ fg
sleep 1000
^Z
[1]+  Stopped           sleep 1000
$
# The first process will continue! This is also the case if the reverse procedure were used
$ sleep 10 &
[1] 27346
$ sleep 1000 &
[2] 27347
$
[1]-  Done             sleep 10
$ fg
sleep 1000
^Z
```

```
  [2]+  Stopped                 sleep 1000
```