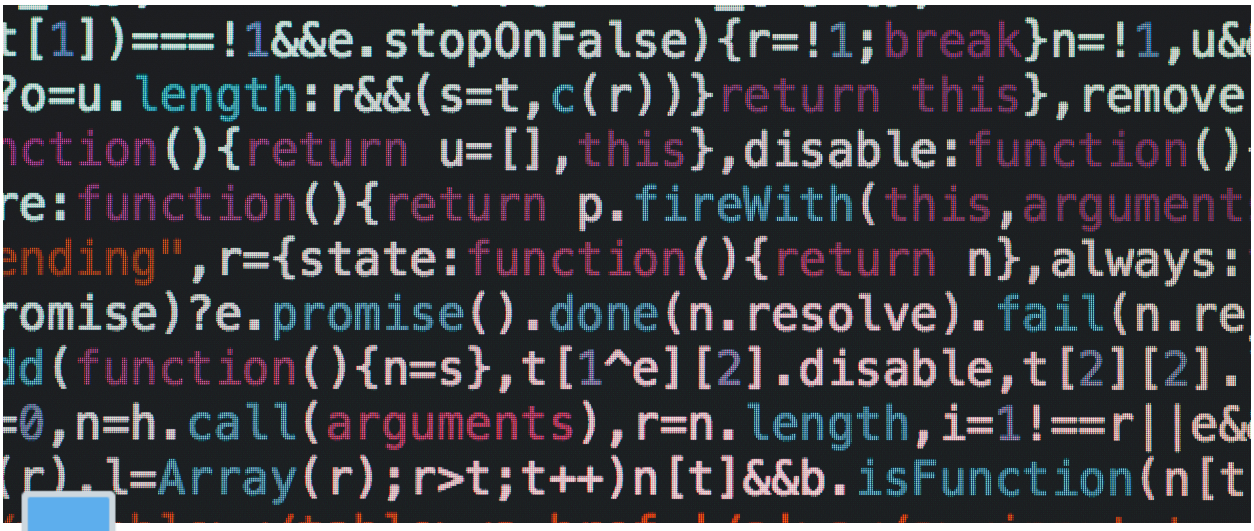

nonohup vs nohup

Functionally there are very different even though they have the same name

nonohup

- Usage:
  - `nonohup command [command-args....]`
- When the user enters the above command, the program executes the command and it's in the background while the next `$S20` prompt is presented
  - **How it works:** When nonohup is indexed behind a command and its args, a child process is created and the command is executed in the child process similar to & (background command). `control + d` or closing the shell using `control a + x` will completely halt the execution of the nonohup and its arguments in the background

nohup

- Usage:
  - `nohup command [command-args ...]`

    - `nonohup - - help | - - version`
- nonohup stands for "no hangup". The nohup command executes the command and its arguments even after the session is disconnected
    - **How it works:** Usually the HUP (hangup signal) is sent to the process to inform it that the user has logged off. nohup however intercept the signal and then redirects the input output from command to nohup.out. instead of standard output.

Differences

- nonohup ends with user logging off while nohup doesn't
- nohup prints information to nohup.out instead of stdout
- nohup can be coupled with (&) while nonohup is a reimplementation of (&)
- Killing nohup requires the user to obtain the pid and use `kill -9 pid`. nonohup can be stopped with `control+d` or `control a + x`
- *Other differences can be seen in the definitions above*