Bash

2021年10月29日 15:54

Bash Cheatsheet
https://devhints.io/bash

Special Characters in Bash
https://www.howtogeek.com/439199/15-special-characters-you-need-to-know-for-bash/

`set -eou pipefail`	-e: causes the shell to exit if any subcommand or pipeline returns a non-zero status. -o: <option-name>. Sets the <option-name> flag. In this case, it sets the pipefail option. -u: errors if an variable is referenced before being set Pipefail: If set, causes the pipeline to return the exit status of the last (rightmost) command to exit with a non-zero status. It returns zero if all commands in the pipeline exit successfully. This option is disabled by default. Generally make the script more robust
`set +e`	Use +e to disable automatic script failure if the command fails, as it is possible the simulation might go wrong.
`stderr`	Error output stream. Basically show the output on the screen. https://www.cnblogs.com/badboy200800/p/11121880.html
`mkdir -p bin`	Make a bin directory if it doesn't exist **mkdir** [*OPTION*]... *DIRECTORY*... **Description** Create the DIRECTORY(ies), if they do not already exist. Mandatory arguments to long options are mandatory for short options too. -**m**, --**mode**=*MODE* set file mode (as in chmod), not a=rwx - umask -**p**, --**parents** no error if existing, make parent directories as needed -**v**, --**verbose** print a message for each created directory -**Z**, --**context**=*CTX* set the SELinux security context of each created directory to CTX Ads by Google Send feedback Why this ad? https://linux.die.net/man/1/mkdir https://www.runoob.com/linux/linux-comm-mkdir.html **/bin** The **/bin** directory contains **binaries** for use by all users. The '/bin' directory also contains executable files, Linux commands that are used in single user mode, and common commands that are used by all the users, like cat, cp, cd, ls, etc. According to the FHS the **/bin** directory should contain **/bin/cat** and **/bin/date** (among others). The '/bin' directory doesn't contain directories. https://linux.die.net/man/1/mkdir https://www.runoob.com/linux/linux-comm-mkdir.html
VARIANR='$1'	First parameter to script determines the variant

$()
${}
	1. `$()` means: "first evaluate this, and then evaluate the rest of the line".
	Ex :
	`echo $(pwd)/myFile.txt` pwd: print working directory
	will be interpreted as
	`echo /my/path/myFile.txt`
	On the other hand `${}` expands a variable.
	Ex:
	`MY_VAR=toto`
	`echo ${MY_VAR}/myFile.txt`
	will be interpreted as
	`echo toto/myFile.txt`

chmod +x ... chmod u+x ...	The man page of `chmod` covers that. • *u* stands for user. • *g* stands for group. • *o* stands for others. • *a* stands for all. That means that `chmod u+x somefile` will grant only the owner of that file execution permissions whereas `chmod +x somefile` is the same as `chmod a+x somefile`. The chmod man page says: > The format of a symbolic mode is `[ugoa...][[+-=][rwxXstugo...]...][,...]`. Multiple symbolic operations can be given, separated by commas. > A combination of the letters 'ugoa' controls which users' access to the file will be changed: the user who owns it (u), other users in the file's group (g), other users not in the file's group (o), or all users (a). If none of these are given, the effect is as if 'a' were given, but bits that are set in the umask are not affected. Just doing `+x` will apply it to all flags: [u]ser, [g]roup, [o]thers.
cat	1. Create file cat > example.txt 2. Read file cat example.txt

$	Hold variable
	`ThisDistro=Ubuntu`
	`MyNumber=2001`
	`echo $ThisDistro`
	`echo $MyNumber`

basename	1.
	The `basename` command removes any trailing `/` characters:
	`$ basename /usr/local/`
	`$ basename /usr/local`
	Both commands will produce the same output:
	Output
	local
	local
	2.
	Removing a Trailing Suffix
	To remove any trailing suffix from the file name, pass the suffix as a second argument:
	`$ basename -s .conf /etc/sysctl.conf`

	`$ basename -s .conf /etc/sysctl.conf` Output sysctl
echo <&2	0 - stdin 1 - stdout 2 - stderr `echo "hey" >&2` `>` redirect standard output (implicit `1>`) `&` what comes next is a file descriptor, not a file (only for right hand side of `>`) `2` stderr file descriptor number https://stackoverflow.com/questions/23489934/echo-2-some-text-what-does-it-mean-in-shell-scripting https://unix.stackexchange.com/questions/223086/must-i-add-2-in-the-end-of-echo-command
-ne	Not eqaul to
sed	SED command in UNIX is stands for stream editor and it can perform lot's of function on file like, searching, find and replace, insertion or deletion. Though most common use of SED command in UNIX is for substitution or for find and replace. By using SED you can edit files even without opening it, which is much quicker way to find and replace something in file, than first opening that file in VI Editor and then changing it. https://www.geeksforgeeks.org/sed-command-in-linux-unix-with-examples/
diff	diff is **a command-line utility that allows you to compare two files line by line**. It can also compare the contents of directories. 2019年11月25日