Bash Cheatsheet

This document contains bash specific commands / syntax which may not be completely POSIX complaint.

POSIX complaint shell scripting doc will be ready after this one ;)

Variables

- **Define**

```
name='PJ'
age=10
os=$(uname)
```

- **Use**

```
echo $name
printf '- %s\n' \
        $age \
        $os
```

- **Execute**

```
$ bash tmp.sh
```

- or if it has shebang and execute permission

```
$ ./tmp.sh
PJ
- 10
- Linux
$_
```

- **Length**

```
a='Hello World'
b=1917
echo ${#a} # 11
echo ${#b} # 4
```

Special Variables

Variable	Description
`$0`	Name of script
`$1, $2, $3, ...`	First, Second, Third, ... argument of script
`$#`	Number of arguments were passed to the script
`$@`	All arguments of the script (can be iterated)
`$*`	All arguments of the script (cannot be iterated)
`$?`	Return value of the last execution in script
`$$`	The PID of the script
`$USER`	The user which is running the script (username)
`$HOSTNAME`	The host name of the machine
`$LINENO`	Current line number inside script
`$RANDOM`	Random number

Input

- **Basic**

```
read input_variable
echo ${input_variable}
```

- **with prompt message**

```
read -p 'are you sure? ' input_variable
```

- **silent input**

```bash
read -sp 'Input is silent: ' input_variable
```

Arrays

- **Define**

```bash
files=('f1.txt' 'f2.txt' 'f3.txt')
echo ${files[0]} # the first element
echo ${files[*]} # all elements
echo ${files[@]} # same
echo ${#files} # size/length of array
```

- **Add elements**

```bash
files+=('f4.txt')
```

- **Remove elements**

```bash
unset files[0]
```

Arithmetic

Basic Expressions

Operator	Operation
`+`	Addition
`-`	Subtraction
`*`	Multiply
`/`	Deviation
`%`	Remainder
`var++`	Increase by 1
`var--`	Decrease by 1

- **let**

```
let a=4+5 # 9
let 'A = 4 + 6' # 10
let a++ # 10
let A-- # 9
```

- **expr**

```
var_two=$( expr 4 \* 5 ) # 20
```

- **Double Parentheses**

```
a=$((3 + 5)) # 8
b=$(( a + 3 )) # 11
(( b++ )) # 12
```

Conditions

- **Test Operations**

Operator	Operation
`!`	Not

- **String Operations**

Operator	Operation
`-z`	Is null
`-n`	Is not null
`==`	Is equal
`!=`	Is not equal

- **Numerical Operations**

Operator	Operation
`-eq`	equal
`-lt`	less than
`-gt`	greater than
`-le`	less-equal to

Operator	Operation
`-ge`	greater-equal to

- **File Operations**

Operator	Operation
`-e`	Exists
`-d`	Exists and it’s a directory
`-f`	Exists and it’s a file
`-r`	Exists and has *read* permission
`-w`	Exists and has *write* permission
`-x`	Exists and has *execute* permission
`-s`	Exists and it’s not empty

- **if**

```
if [[ `echo $(date +%s) % 5 | bc` -eq 0 ]]; then
    echo "It can be devided by 5 without any reminder"
elif [[ ${second_condition} ]]; then
    echo "The second is true"
else
    printf '%s\n' \
        "Nothing is true" \
        "Everything is permitted"
fi
```

- **Nested**

```
if [ 10 -gt 5 ]; then
    echo True
    if (( 10 % 2 == 0 )); then
        echo and Even
    fi
fi
```

- **inline**

```bash
[[ ${some_condition} ]] && echo "it's true" || echo 'false'
```

Loops

- for

```bash
for i in {1..10}
do
    echo ${i}
done
```

- while

```bash
counter=1
while [[ ${counter} -le 9 ]]; do
    echo "${counter}"
    ((counter++))
done
```

- until

```bash
counter=1
until [[ ${counter} -gt 9 ]]
do
    echo "${counter}"
    ((counter++))
done
```

- select

```bash
names='Kyle Cartman Stan Quit'
PS3='Select character: '
select name in ${names}; do
    [[ $name == 'Quit' ]] && break
    echo Hello ${name}
done
echo Bye
```