NOTES on:
Automate Daily Development Tasks with Bash

Change current working directory

```
# to go into a directory from your current directory
cd <file path here>

# to go up a directory from your current directory
cd ..
```

List a directory's contents

```
# list information
ls

# for more details, add -l (long)
ls -l

# to see hidden files/folders (like .git or .npmignore)
ls -a

# Note, flags can be combined like so
ls -la
```

Output a file to the screen (stdout)

```
cat <file name>

# shows it with line numbers
cat -n <file name>
```

View a file in bash

```
# view the file without dumping it all onto your screen
less <file name>

#  Useful shortcuts in less
#   Shift+g   (jump to end)
#   g         (go back to top)
#   /         (search)
#   q         (quit/close)
```

View file/folder in default application

```
open <file/folder name>

# view current directory in Finder
open .

# specify an application to use
open <file name> -a TextEdit
```

Create a file

```
touch <file name>
```

Set or append to a file

```
# set the file's contents
echo 'hi' > file.txt

# append to file's contents
echo 'hi' >> file.txt
```

Create a directory

```
mkdir <folder name>
# make intermediary directories as needed
mkdir -p parent/child/grandchild
```

Remove a file

```
# Note, this permanently deletes a file
rm <file name>

# Remove a folder and it's contents, recursively
rm -rf <folder name>
```

Move a file

```
mv <target> <destination>
# for example, to rename a file
mv a.js b.js
# move all files in a folder to another folder
mv lib/* src
```

Copy a file

```
cp <target> <destination>

# copy everything recursively from one folder to another
cp -R src/* lib
```

Find Files and Folders with `find`

```
# find all the PNGs in a folder
find <path> -name "*.png"
# find all the JPGs (case insensitive) in a folder
find <path> -iname "*.jpg"
# find only directories
find <path> -type d
# delete all matching files
find <path> -name "*.built.js" -delete
# execute an arbitrary action on each match
# remember `{}` will be replaced with the file name
find <path> -name "*.png" -exec pngquant {} \;
```

Search for text with `grep`

```
grep <pattern> <target file or glob>

# Useful flags
# --color     (colorizes matches)
# -n          (show line numbers)
# -C <number> (show N lines above/below match for context)
# -e          (regex search)
```

Make HTTP requests in bash with `curl`

```
curl <url>
# Useful flags
# -i    (show response headers only)
# -L    (follow redirects)
# -H    (header flag)
# -X    (set HTTP method)
# -d    (request body)
# -o    (output to a file)
```

Create and run bash scripts

```
echo 'echo Hello World' > script.sh
chmod u+x script.sh #give permissions to execute a file
./script.sh
```

Store and Use Values with bash Variables

```
# no spaces between name, =, and value
var=123
echo $var
# to make it accessible to all child processes of current
shell, export it
export var
# this deletes the variable
unset var
```

```
# To see all environment variables
env
```

Simple Function

```
#!/bin/bash
# Define a function greet

greet () {
   echo "Hello World!"
}

# Call the function greet
greet
```

Understand exit statuses

```
# Exit status 0: success
# Exit status other than 0: failure


# To test on the exit status of a command:

if command;then
   echo 'success'
else
   echo 'failure'
fi
```

```
# Get the last run command's exit status

ls
# will be 0 if it ran successfully, 1 - 255 for an error
echo $?
```

Use Conditional Statements

```
# Some conditional primaries that can be used in the if expression:
#   =, !=      string (in)equality
#   -eq, -ne   numeric (in)equality
#   -lt, -gt   less/greater than
#   -z         check variable is not set
#   -e         check file/folder exists

if [[ $USER = 'cameronnokes' ]]; then
  echo "true"
else
  echo "false"
fi
```

```
# Conditionals can be used inline in a more ternary-like format

[[ $USER = 'cameronnokes' ]] && echo "yes" || echo "no"
```

Chain Commands with Pipes and Redirect Output

```
# ps ax will list all running processes
ps ax | grep Chrome | less

# get the file size after uglify + gzip
uglifyjs -c -m -- index.js | gzip -9 | wc -c
```

```
# redirect stdout to a file
ls > ls.txt

# append stdout to a file
echo "hi" >> ls.txt
```