tldr pages

Simplified and community-driven man pages

Generated on Mon Aug 9 14:44:51 2021

A ndr oid

am

Android activity manager.

More information: https://developer.android.com/studio/command-line/adb#_am

- Start a specific activity:

```
am start -n {{com.android.settings/.Settings}}
```

- Start an activity and pass data to it:

```
am start -a {{android.intent.action.VIEW}} -d {{tel:123}}
```

- Start an activity matching a specific action and category:

```
am start -a {{android.intent.action.MAIN}} -c
{{android.intent.category.HOME}}
```

- Convert an intent to an URI:

```
am to-uri -a {{android.intent.action.VIEW}} -d {{tel:123}}
```

bugreport

Show an Android bug report.

This command can only be used through `adb shell`.

More information: https://android.googlesource.com/platform/frameworks/native/+/master/cmds/bugreport/.

- Show a complete bug report of an Android device:

  ```
  bugreport
  ```

bugreportz

Generate a zipped Android bug report.

This command can only be used through `adb shell`.

More information: https://android.googlesource.com/platform/frameworks/native/+/master/cmds/bugreportz/

- Generate a complete zipped bug report of an Android device:

  ```
  bugreportz
  ```

- Show the progress of a running `bugreportz` operation:

  ```
  bugreportz -p
  ```

- Show the version of `bugreportz`:

  ```
  bugreportz -v
  ```

- Display help:

  ```
  bugreportz -h
  ```

cmd

Android service manager.

More information: https://cs.android.com/android/platform/superproj ect/+ / master:frameworks/native/cmds/cmd/.

- List every running service:

  ```
  cm d - l
  ```

- Call a specific service:

  ```
  cm d {{alarm }}
  ```

- Call a service with arguments:

  ```
  cm d {{v ib rator}} {{v ib rate 3 0 0 }}
  ```

dalvikvm

Android Java virtual machine.

More information: https://source.android.com/devices/tech/dalvik.

- Start a Java program:

```
dalvikvm -classpath {{path/to/file.jar}} {{classname}}
```

dumpsys

Provide information about Android system services.

This command can only be used through `adb shell`.

More information: https://developer.android.com/studio/command-line/dumpsys.

- Get diagnostic output for all system services:

  ```
  dumpsys
  ```

- Get diagnostic output for a specific system service:

  ```
  dumpsys {{service}}
  ```

- List all services `dumpsys` can give information about:

  ```
  dumpsys -l
  ```

- List service-specific arguments for a service:

  ```
  dumpsys {{service}} -h
  ```

- Exclude a specific service from the diagnostic output:

  ```
  dumpsys --skip {{service}}
  ```

- Specify a timeout period in seconds (defaults to 10 s):

  ```
  dumpsys -t {{seconds}}
  ```

getprop

Show information about Android system properties.

More information: https://manned.org/getprop.

- Display information about Android system properties:

```
getprop
```

- Display information about a specific property:

```
getprop {{prop}}
```

- Display the SDK API level:

```
getprop {{ro.build.version.sdk}}
```

- Display the Android version:

```
getprop {{ro.build.version.release}}
```

- Display the Android device model:

```
getprop {{ro.vendor.product.model}}
```

- Display the OEM unlock status:

```
getprop {{ro.oem_unlock_supported}}
```

- Display the MAC address of the Android's WiFi card:

```
getprop {{ro.boot.wifimaddr}}
```

input

Send event codes or touchscreen gestures to an Android device.

This command can only be used through `adb shell`

More information: https://developer.android.com/reference/android/view/KeyEvent.html#constants. 1

- Send an event code for a single character to an Android device:

  ```
  input keyevent {{event_code}}
  ```

- Send a text to an Android device (`%s` represents spaces):

  ```
  input text "{{text}}"
  ```

- Send a single tap to an Android device:

  ```
  input tap {{x_pos}} {{y_pos}}
  ```

- Send a swipe gesture to an Android device:

  ```
  input swipe {{x_start}} {{y_start}} {{x_end}} {{y_end}} {{duration_in_ms}}
  ```

- Send a long press to an Android device using a swipe gesture:

  ```
  input swipe {{x_pos}} {{y_pos}} {{x_pos}} {{y_pos}} {{duration_in_ms}}
  ```

logcat

Dump a log of system messages.

More information: https://developer.android.com/studio/command-line/logcat.

- Display system logs:

  ```
  log cat
  ```

- Write system logs to a file:

  ```
  log cat - f {{path/to/file}}
  ```

- Display lines that match a regular expression:

  ```
  log cat - - reg ex {{reg u lar_expression}}
  ```

pm

Show information about apps on an Android device.

More information: https://developer.android.com/studio/command-line/adb#pm

- Print a list of all installed apps:

```
pm list packages
```

- Print a list of all installed system apps:

```
pm list packages -s
```

- Print a list of all installed 3rd-Party apps:

```
pm list packages -3
```

- Print a list of apps matching specific keywords:

```
pm list packages {{keywords}}
```

- Print the path of the APK of a specific app:

```
pm path {{app}}
```

settings

G et information about the Android O S .

More information: https://adbinstaller.com/commands/adb-shell-settings-5 b6 7 0 d5 ee7 9 5 8 1 7.8 a2 9 5 5 5 3 6

- Display a list of settings in the `g lob al` namespace:

  ```
  setting s list {{g lob al}}
  ```

- G et the value of a specific setting:

  ```
  setting s g et {{g lob al}} {{airplane_m ode_on}}
  ```

- S et the value of a setting:

  ```
  setting s pu t {{system }} {{screen_b rig htness}} {{4 2 }}
  ```

- Delete a specific setting:

  ```
  setting s delete {{secu re}} {{screensav er_enab led}}
  ```

wm

Show information about the screen of an Android device.

This command can only be used through `adb shell`

More information: https://adbinstaller.com/commands/adb-shell-wm-5b67 2b17 e7958 17 8a2955538

- Display the physical size of an Android device's screen:

  ```
  wm {{size}}
  ```

- Display the physical density of an Android device's screen:

  ```
  wm {{density}}
  ```

Common

2to3

Automated Python 2 to 3 code conversion.

More information: https://docs.python.org/3/library/2to3.html

- Display the changes that would be performed without performing them (dry-run):

```
2to3 {{path/to/file.py}}
```

- Convert a Python 2 file to Python 3:

```
2to3 --write {{path/to/file.py}}
```

- Convert specific Python 2 language features to Python 3:

```
2to3 --write {{path/to/file.py}} --fix={{raw_input}} --fix={{print}}
```

- Convert all Python 2 language features except the specified ones to Python 3:

```
2to3 --write {{path/to/file.py}} --nofix={{has_key}} --nofix={{isinstance}}
```

- Display a list of all available language features that can be converted from Python 2 to Python 3:

```
2to3 --list-fixes
```

- Convert all Python 2 files in a directory to Python 3:

```
2to3 --output-dir={{path/to/python3_directory}} --write-unchanged-files --nobackups {{path/to/python2_directory}}
```

- Run 2to3 with multiple threads:

```
2to3 --processes={{4}} --output-dir={{path/to/python3_directory}} --write --nobackups --no-diff {{path/to/python2_directory}}
```

7 z

File archiver with a high compression ratio.

More information: https://www.7-zip.org

- [a]rchive a file or directory:

```
7z a {{path/to/archive.7z}} {{path/to/file_or_directory}}
```

- Encrypt an existing archive (including filenames):

```
7z a {{path/to/encrypted.7z}} -p{{password}} -mhe=on {{path/to/archive.7z}}
```

- Extract an archive preserving the original directory structure:

```
7z x {{path/to/archive.7z}}
```

- Extract an archive to a specific directory:

```
7z x {{path/to/archive.7z}} -o{{path/to/output}}
```

- Extract an archive to stdout:

```
7z x {{path/to/archive.7z}} -so
```

- [a]rchive using a specific archive type:

```
7z a -t{{7z|zip|gzip|bzip2|lzip}} {{path/to/archive.7z}} {{path/to/file_or_directory}}
```

- [l]ist the contents of an archive:

```
7z l {{path/to/archive.7z}}
```

- List available archive types:

```
7z i
```

7za

File archiver with a high compression ratio.

Similar to `7z` except that it supports fewer file types but is cross-platform.

More information: https://www.7-zip.org

- [a]rchive a file or directory:

  ```
  7za a {{path/to/archive.7z}} {{path/to/file_or_directory}}
  ```

- Encrypt an existing archive (including file names):

  ```
  7za a {{path/to/encrypted.7z}} -p{{password}} -mhe=on {{path/to/archive.7z}}
  ```

- Extract an archive preserving the original directory structure:

  ```
  7za x {{path/to/archive.7z}}
  ```

- Extract an archive to a specific directory:

  ```
  7za x {{path/to/archive.7z}} -o{{path/to/output}}
  ```

- Extract an archive to stdout:

  ```
  7za x {{path/to/archive.7z}} -so
  ```

- [a]rchive using a specific archive type:

  ```
  7za a -t{{7z|zip|gzip|bzip2|lzip}} {{path/to/archive.7z}} {{path/to/file_or_directory}}
  ```

- [l]ist the contents of an archive:

  ```
  7za l {{path/to/archive.7z}}
  ```

- List available archive types:

  ```
  7za i
  ```

7zr

File archiver with a high compression ratio.

Similar to `7z` except that it only supports `.7z` files.

More information: https://www.7-zip.org

- [a]rchive a file or directory:

```
7zr a {{path/to/archive.7z}} {{path/to/file_or_directory}}
```

- Encrypt an existing archive (including file names):

```
7zr a {{path/to/encrypted.7z}} -p{{password}} -mhe=on {{path/to/archive.7z}}
```

- Extract an archive preserving the original directory structure:

```
7zr x {{path/to/archive.7z}}
```

- Extract an archive to a specific directory:

```
7zr x {{path/to/archive.7z}} -o{{path/to/output}}
```

- Extract an archive to stdout:

```
7zr x {{path/to/archive.7z}} -so
```

- [l]ist the contents of an archive:

```
7zr l {{path/to/archive.7z}}
```

- List available archive types:

```
7zr i
```

[

Evaluate condition.

Returns 0 if the condition evaluates to true, 1 if it evaluates to false.

More information: https://www.gnu.org/software/coreutils/test.

- Test if a given variable is equal to a given string:

```
[ "{{$VARIABLE}}" == "{{/bin/zsh}}" ]
```

- Test if a given variable is empty:

```
[ -z "{{$GIT_BRANCH}}" ]
```

- Test if a file exists:

```
[ -f "{{path/to/file}}" ]
```

- Test if a directory does not exist:

```
[ ! -d "{{path/to/directory}}" ]
```

- If-else statement:

```
[ {{condition}} ] && {{echo "true"}} || {{echo "false"}}
```

aapt

Android Asset Packaging Tool.

Compile and package an Android app's resources.

More information: https://elinux.org/Android_aapt

- List files contained in an APK archive:

  ```
  aapt list {{path/to/app.apk}}
  ```

- Display an app's metadata (version, permissions, etc.):

  ```
  aapt dumpbadging {{path/to/app.apk}}
  ```

- Create a new APK archive with files from the specified directory:

  ```
  aapt package -F {{path/to/app.apk}} {{path/to/directory}}
  ```

ab

Apache HTTP server benchmarking tool.

More information: https://httpd.apache.org/docs/current/programs/ab.html.

- Execute 100 HTTP GET requests to a given URL:

```
ab -n {{100}} {{url}}
```

- Execute 100 HTTP GET requests, in concurrent batches of 10, to a URL:

```
ab -n {{100}} -c {{10}} {{url}}
```

- Execute 100 HTTP POST requests to a URL, using a JSON payload from a file:

```
ab -n {{100}} -T {{application/json}} -p {{path/to/file.json}} {{url}}
```

- Use HTTP [K]eep Alive, i.e. perform multiple requests within one HTTP session:

```
ab -k {{url}}
```

- Set the maximum number of seconds to spend for benchmarking:

```
ab -t {{60}} {{url}}
```

abduco

Terminal session manager.

More information: http://www.brain-dump.org/projects/abduco/

- List sessions:

  ```
  abduco
  ```

- Attach to a session, creating it if it doesn't exist:

  ```
  abduco -A {{name}} {{bash}}
  ```

- Attach to a session with `dvtm`, creating it if it doesn't exist:

  ```
  abduco -A {{name}}
  ```

- Detach from a session:

  ```
  Ctrl + \
  ```

- Attach to a session in read-only mode:

  ```
  abduco -Ar {{name}}
  ```

ack

A search tool like grep, optimized for developers.

See also: `rg`, which is much faster.

More information: https://beyondgrep.com/documentation.

- Search for files containing a string or regular expression in the current directory recursively:

  ```
  ack "{{search_pattern}}"
  ```

- Search for a case-insensitive pattern:

  ```
  ack --ignore-case "{{search_pattern}}"
  ```

- Search for lines matching a pattern, printing [o]nly the matched text and not the rest of the line:

  ```
  ack -o "{{search_pattern}}"
  ```

- Limit search to files of a specific type:

  ```
  ack --type={{ruby}} "{{search_pattern}}
  ```

- Do not search in files of a specific type:

  ```
  ack --type=no{{ruby}} "{{search_pattern}}
  ```

- Count the total number of matches found:

  ```
  ack --count --no-filename "{{search_pattern}}"
  ```

- Print the file names and the number of matches for each file only:

  ```
  ack --count --files-with-matches "{{search_pattern}}"
  ```

- List all values that can be used with `--type`:

  ```
  ack --help-types
  ```

acme.sh --dns

Use a DNS-01 challenge to issue a TLS certificate.

More information: https://github.com/acmesh-official/acme.sh/wiki.

- Issue a certificate using an automatic DNS API mode:

```
acme.sh --issue --dns {{gnd_gd}} --domain {{example.com}}
```

- Issue a wildcard certificate (denoted by an asterisk) using an automatic DNS API mode:

```
acme.sh --issue --dns {{dns_namesilo}} --domain
{{example.com}} --domain {{*.example.com}}
```

- Issue a certificate using a DNS alias mode:

```
acme.sh --issue --dns {{dns_cf}} --domain {{example.com}} --
challenge-alias {{alias-for-example-validation.com}}
```

- Issue a certificate while disabling automatic Cloudflare/Google DNS polling after the DNS record is added by specifying a custom wait time in seconds:

```
acme.sh --issue --dns {{dns_namecheap}} --domain
{{example.com}} --dnssleep {{300}}
```

- Issue a certificate using a manual DNS mode:

```
acme.sh --issue --dns --domain {{example.com}} --yes-I-know
dns-manual-mode-enough-go-ahead-please
```

acme.sh

Shell script implementing ACME client protocol, an alternative to certbot.

See also **acme.sh dns**

More information: https://github.com/acmesh-official/acme.sh.

- Issue a certificate using webroot mode:

```
acme.sh --issue --domain {{example.com}} --webroot {{/path/to/webroot}}
```

- Issue a certificate for multiple domains using standalone mode using port 80:

```
acme.sh --issue --standalone --domain {{example.com}} --domain {{www.example.com}}
```

- Issue a certificate using standalone TLS mode using port 443:

```
acme.sh --issue --alpn --domain {{example.com}}
```

- Issue a certificate using a working Nginx configuration:

```
acme.sh --issue --nginx --domain {{example.com}}
```

- Issue a certificate using a working Apache configuration:

```
acme.sh --issue --apache --domain {{example.com}}
```

- Issue a wildcard (*) certificate using an automatic DNS API mode:

```
acme.sh --issue --dns {{dns_cf}} --domain {{*.example.com}}
```

- Install certificate files into the specified locations (useful for automatic certificate renewal):

```
acme.sh --install-cert -d {{example.com}} --key-file {{/path/to/example.com.key}} --fullchain-file {{/path/to/example.com.cer}} --reloadcmd {{"systemctl force-reload nginx"}}
```

act

Execute GitHub Actions locally using Docker.

More information: https://github.com/nektos/act.

- List the available actions:

  ```
  act -l
  ```

- Run the default event:

  ```
  act
  ```

- Run a specific event:

  ```
  act {{event_type}}
  ```

- Run a specific action:

  ```
  act -a {{action_id}}
  ```

- Do not actually run the actions (i.e. a dry run):

  ```
  act -n
  ```

- Show verbose logs:

  ```
  act -v
  ```

adb install

Android Debug Bridge Install: Push packages to an Android emulator instance or connected Android devices.

More information: https://developer.android.com/studio/command-line/adb.

- Push an Android application to an emulator/device:

```
adb install {{path/to/file.apk}}
```

- Reinstall an existing app, keeping its data:

```
adb install -r {{path/to/file.apk}}
```

- Grant all permissions listed in the app manifest:

```
adb install -g {{path/to/file.apk}}
```

- Quickly update an installed package by only updating the parts of the APK that changed:

```
adb install --fastdeploy {{path/to/file.apk}}
```

adb reverse

Android Debug Bridge Reverse: reverse socket connections from an Android emulator instance or connected Android devices.

More information: https://developer.android.com/studio/command-line/adb.

- List all reverse socket connections from emulators and devices:

```
adb reverse --list
```

- Reverse a TCP port from an emulator or device to localhost:

```
adb reverse tcp:{{remote_port}} tcp:{{local_port}}
```

- Remove a reverse socket connections from an emulator or device:

```
adb reverse --remove tcp:{{remote_port}}
```

- Remove all reverse socket connections from all emulators and devices:

```
adb reverse --remove-all
```

adb shell

Android Debug Bridge Shell: Run remote shell commands on an Android emulator instance or connected Android devices.

More information: https://developer.android.com/studio/command-line/adb.

- Start a remote interactive shell on the emulator/device:

```
adb shell
```

- Get all the properties from emulator or device:

```
adb shell getprop
```

- Revert all runtime permissions to their default:

```
adb shell pm reset-permissions
```

- Revoke a dangerous permission for an application:

```
adb shell pm revoke {{package}} {{permission}}
```

- Trigger a key event:

```
adb shell input keyevent {{keycode}}
```

- Clear the data of an application on an emulator or device:

```
adb shell pm clear {{package}}
```

- Start an activity on emulator/device:

```
adb shell am start -n {{package}}/{{activity}}
```

- Start the home activity on an emulator or device:

```
adb shell am start -W -c android.intent.category.HOME -a
android.intent.action.MAIN
```

adb

Android Debug Bridge: communicate with an Android emulator instance or connected Android devices.

More information: https://developer.android.com/studio/command-line/adb.

- Check whether the adb server process is running and start it:

```
adb start-server
```

- Terminate the adb server process:

```
adb kill-server
```

- Start a remote shell in the target emulator/device instance:

```
adb shell
```

- Push an Android application to an emulator/device:

```
adb install -r {{path/to/file.apk}}
```

- Copy a file/directory from the target device:

```
adb pull {{path/to/device_file_or_directory}} {{path/to/local_destination_directory}}
```

- Copy a file/directory to the target device:

```
adb push {{path/to/local_file_or_directory}} {{path/to/device_destination_directory}}
```

- Get a list of connected devices:

```
adb devices
```

AdG uardH ome

A network-wide software for blocking ads & tracking.

More information: https://github.com/AdguardT eam/AdG uardH ome

- R un AdG uard H ome:

  ```
  A dG u ardH om e
  ```

- R un AdG uard H ome with a specific config:

  ```
  A dG u ardH om e - - config  {{path/to/A dG u ardH om e. yam l}}
  ```

- S et the work directory for data to be stored in:

  ```
  A dG u ardH om e - - w ork - dir {{path/to/directory}}
  ```

- Install or uninstall AdG uard H ome as a service:

  ```
  A dG u ardH om e - - serv ice {{install| u ninstall}}
  ```

- S tart the AdG uard H ome service:

  ```
  A dG u ardH om e - - serv ice start
  ```

- R eload the configuration for the AdG uard H ome service:

  ```
  A dG u ardH om e - - serv ice reload
  ```

- S top or restart the AdG uard H ome service:

  ```
  A dG u ardH om e - - serv ice {{stop| restart}}
  ```

ag

The Silver Searcher. Like ack, but aims to be faster.

More information: https://github.com/ggreer/the_silver_searcher

- Find files containing "foo", and print the line matches in context:

```
ag {{foo}}
```

- Find files containing "foo" in a specific directory:

```
ag {{foo}} {{path/to/directory}}
```

- Find files containing "foo", but only list the filenames:

```
ag -l {{foo}}
```

- Find files containing "FOO" case-insensitively, and print only the match, rather than the whole line:

```
ag -i -o {{FOO}}
```

- Find "foo" in files with a name matching "bar":

```
ag {{foo}} -G {{bar}}
```

- Find files whose contents match a regular expression:

```
ag '{{^bar|z)$}}'
```

- Find files with a name matching "foo":

```
ag -g {{foo}}
```

age

A simple, modern and secure file encryption tool.

More information: https://age-encryption.org.

- Generate an encrypted file that can be decrypted with a passphrase:

  ```
  age --passphrase --output {{path/to/encrypted_file}} {{path/to/unencrypted_file}}
  ```

- Generate a key pair, saving the private key to an unencrypted file and printing the public key to stdout:

  ```
  age-keygen -o {{path/to/file}}
  ```

- Encrypt a file with one or more public keys that are entered as literals:

  ```
  age --recipient {{public_key_1}} --recipient {{public_key_2}} {{path/to/unencrypted_file}} --output {{path/to/encrypted_file}}
  ```

- Encrypt a file with one or more public keys that are specified in a recipients file:

  ```
  age --recipients-file {{path/to/recipients_file}} {{path/to/unencrypted_file}} --output {{path/to/encrypted_file}}
  ```

- Decrypt a file with a passphrase:

  ```
  age --decrypt --output {{path/to/decrypted_file}} {{path/to/encrypted_file}}
  ```

- Decrypt a file with a private key file:

  ```
  age --decrypt --identity {{path/to/private_key_file}} --output {{path/to/decrypted_file}} {{path/to/encrypted_file}}
  ```

airmon-ng

Activate monitor mode on wireless network devices.

More information: https://www.aircrack-ng.org/doku.php?_id=_airmon-ng

- List wireless devices and their statuses:

```
su do airm on- ng
```

- T urn on monitor mode for a specific device:

```
su do airm on- ng  start {{w lan0 }}
```

- Kill disturbing processes that use wireless devices:

```
su do airm on- ng  check  k ill
```

- T urn off monitor mode for a specific network interface:

```
su do airm on- ng  stop {{w lan0 m on}}
```

airpaste

Share messages and files on the same network using mDNS.

More information: https://github.com/mafintosh/airpaste.

- Wait for a message and display it when received:

```
airpaste
```

- Send text:

```
echo {{text}} | airpaste
```

- Send a file:

```
airpaste < {{path/to/file}}
```

- Receive a file:

```
airpaste > {{path/to/file}}
```

- Create or join a channel:

```
airpaste {{channel_name}}
```

aj son

Executes JSONPath on JSON objects.

More information: https://github.com/spyzhov/ajson

- Read JSON from a file and execute a specified JSONPath expression:

  ```
  ajson '{{$..json[?(@.path)]}}' {{path/to/file.json}}
  ```

- Read JSON from stdin and execute a specified JSONPath expression:

  ```
  cat {{path/to/file.json}} | ajson '{{$..json[?(@.path)]}}'
  ```

- Read JSON from a URL and evaluate a specified JSONPath expression:

  ```
  ajson '{{avg($..price)}}' '{{https://example.com/api/}}'
  ```

- Read some simple JSON and calculate a value:

  ```
  echo '{{3}}' | ajson '{{2 * pi * $}}'
  ```

alacritty

Cross-platform, GPU-accelerated terminal emulator.

More information: https://github.com/alacritty/alacritty.

- Open a new Alacritty window:

  ```
  alacritty
  ```

- Run in a specific directory:

  ```
  alacritty --working-directory {{path/to/directory}}
  ```

- Run a command in a new Alacritty window:

  ```
  alacritty -e {{command}}
  ```

- Specify alternative configuration file (defaults to `$XDG_CONFIG_HOME/alacritty/alacritty.yml`):

  ```
  alacritty --config-file {{path/to/config.yml}}
  ```

- Run with live config reload enabled (can also be enabled by default in `alacritty.yml`):

  ```
  alacritty --live-config-reload --config-file {{path/to/config.yml}}
  ```

alex

A tool that catches insensitive, inconsiderate writing.

It helps you find gender favouring, polarising, race related, religion inconsiderate, or other uneq ual phrasing in text.

More information: https://github.com/get-alex/alex.

- Analyz e text from stdin:

  ```
  echo {{H is netw ork look s g ood}} | alex - - stdin
  ```

- Analyz e all files in the current directory:

  ```
  alex
  ```

- Analyz e a specific file:

  ```
  alex {{textfile. m d}}
  ```

- Analyz e all Markdown files except `exam ple. m d`:

  ```
  alex * . m d ! {{exam ple. m d}}
  ```

alias

Creates aliases -- words that are replaced by a command string.

Aliases expire with the current shell session unless defined in the shell's configuration file, e.g. `~/.bash_rc`

More information: https://tldp.org/LDP/abs/html/aliases.html

- List all aliases:

  ```
  alias
  ```

- Create a generic alias:

  ```
  alias {{word}}="{{command}}"
  ```

- View the command associated to a given alias:

  ```
  alias {{word}}
  ```

- Remove an aliased command:

  ```
  unalias {{word}}
  ```

- Turn `rm` into an interactive command:

  ```
  alias {{rm}}="{{rm -i}}"
  ```

- Create `la` as a shortcut for `ls -a`

  ```
  alias {{la}}="{{ls -a}}"
  ```

amass db

Interact with an Amass database.

More information: https://github.com/OWASP/Amass/blob/master/doc/user_guide.md#the-db-subcommand

- List all performed enumerations in the database:

  ```
  amass db -dir {{path/to/database_directory}} -list
  ```

- Show results for a specified enumeration index and domain name:

  ```
  amass db -dir {{path/to/database_directory}} -d {{domain_name}} -enum {{index_from_list}} -show
  ```

- List all found subdomains of a domain within an enumeration:

  ```
  amass db -dir {{path/to/database_directory}} -d {{domain_name}} -enum {{index_from_list}} -names
  ```

- Show a summary of the found subdomains within an enumeration:

  ```
  amass db -dir {{path/to/database_directory}} -d {{domain_name}} -enum {{index_from_list}} -summary
  ```

amass enum

Find subdomains of a domain.

More information: https://github.com/OWASP/Amass/blob/master/doc/user_guide.md#the-enum-subcommand

- Passively find subdomains of a domain:

```
amass enum -passive -d {{domain_name}}
```

- Find subdomains of a domain and actively verify them attempting to resolve the found subdomains:

```
amass enum -active -d {{domain_name}} -p {{80,443,8080}}
```

- Do a brute force search for subdomains:

```
amass enum -brute -d {{domain_name}}
```

- Save the results to a text file:

```
amass enum -o {{output_file}} -d {{domain_name}}
```

- Save the results to a database:

```
amass enum -o {{output_file}} -dir {{path/to/database_directory}}
```

amass intel

Collect open source intel on an organisation like root domains and AS N s.

More information: [https://github.com/O WAS P /Amass/blob/master/doc/user_ guide.md# the-intel-subcommand](https://github.com/OWASP/Amass/blob/master/doc/user_guide.md#the-intel-subcommand)

- F ind root domains in an IP address range:

  ```
  am ass intel - addr {{1 9 2 . 1 6 8 . 0 . 1 - 2 5 4 }}
  ```

- U se active recon methods:

  ```
  am ass intel - activ e - addr {{1 9 2 . 1 6 8 . 0 . 1 - 2 5 4 }}
  ```

- F ind root domains related to a domain:

  ```
  am ass intel - w hois - d {{dom ain_nam e}}
  ```

- F ind AS N s belonging to an organisation:

  ```
  am ass intel - org {{org anisation_nam e}}
  ```

- F ind root domains belonging to a given Autonomous S ystem N umber:

  ```
  am ass intel - asn {{asn}}
  ```

- S ave results to a text file:

  ```
  am ass intel - o {{ou tpu t_file}} - w hois - d {{dom ain_nam e}}
  ```

amass track

Track differences between enumerations of the same domain.

More information: https://github.com/OWASP/Amass/blob/master/doc/user_guide.md#the-track-subcommand

- Show the difference between the last two enumerations of the specified domain:

  ```
  amass track -dir {{path/to/database_directory}} -d {{domain_name}} -last 2
  ```

- Show the difference between a certain point in time and the last enumeration:

  ```
  amass track -dir {{path/to/database_directory}} -d {{domain_name}} -since {{01/02 15:04:05 2006 MST}}
  ```

amass viz

Visualize gathered information in a network graph.

More information: https://github.com/OWASP/Amass/blob/master/doc/user_guide.md#the-viz-subcommand

- Generate a D3.js visualization based on database data:

  ```
  amass viz -d3 -dir {{path/to/database_directory}}
  ```

- Generate a DOT file based on database data:

  ```
  amass viz -dot -dir {{path/to/database_directory}}
  ```

- Generate a Gephi Graph Exchange XML Format (GEXF) file based on database data:

  ```
  amass viz -gexf -dir {{path/to/database_directory}}
  ```

- Generate a Graphistry JSON file based on database data:

  ```
  amass viz -graphistry -dir {{path/to/database_directory}}
  ```

- Generate a Maltego CSV file based on database data:

  ```
  amass viz -maltego -dir {{path/to/database_directory}}
  ```

amass

In-depth Attack Surface Mapping and Asset Discovery tool.

More information: https://github.com/OWASP/Amass

- Check the Amass version:

  ```
  amass -version
  ```

- Show general help:

  ```
  amass -help
  ```

- Show help on an Amass subcommand (like `intel`, `enum`, etc.):

  ```
  amass -help {{subcommand}}
  ```

- Execute an Amass subcommand:

  ```
  amass {{subcommand}}
  ```

androguard

Reverse engineering tool for Android applications. Written in Python.

More information: https://github.com/androguard/androguard.

- Display Android app manifest:

  ```
  androguard axml {{path/to/app.apk}}
  ```

- Display app metadata (version and app ID):

  ```
  androguard apkid {{path/to/app.apk}}
  ```

- Decompile Java code from an app:

  ```
  androguard decompile {{path/to/app.apk}} --output {{path/to/directory}}
  ```

ansible-doc

Display information on modules installed in Ansible libraries.

Display a terse listing of plugins and their short descriptions.

More information: https://docs.ansible.com/ansible/latest/cli/ansible-doc.html.

- List available action plugins (modules):

  ```
  ansible-doc --list
  ```

- List available plugins of a specific type:

  ```
  ansible-doc --type {{plugin_type}} --list
  ```

- Show information about a specific action plugin (module):

  ```
  ansible-doc {{plugin_name}}
  ```

- Show information about a plugin with a specific type:

  ```
  ansible-doc --type {{plugin_type}} {{plugin_name}}
  ```

- Show the playbook snippet for action plugin (modules):

  ```
  ansible-doc --snippet {{plugin_name}}
  ```

- Show information about an action plugin (module) as JSON:

  ```
  ansible-doc --json {{plugin_name}}
  ```

ansible-galaxy

Create and manage Ansible roles.

More information: https://docs.ansible.com/ansible/latest/cli/ansible-galaxy.html.

- Install a role:

  ```
  ansible-galaxy install {{username}}.{{role_name}}
  ```

- Remove a role:

  ```
  ansible-galaxy remove {{username}}.{{role_name}}
  ```

- List installed roles:

  ```
  ansible-galaxy list
  ```

- Search for a given role:

  ```
  ansible-galaxy search {{role_name}}
  ```

- Create a new role:

  ```
  ansible-galaxy init {{role_name}}
  ```

- Get information about a user role:

  ```
  ansible-galaxy role info {{username}}.{{role_name}}
  ```

- Get information about a collection:

  ```
  ansible-galaxy collection info {{username}}.
  {{collection_name}}
  ```

ansible-inventory

Display or dump an Ansible inventory.

See also: `ansible`.

More information: https://docs.ansible.com/ansible/latest/cli/ansible-inventory.html.

- Display the default inventory:

  ```
  ansible-inventory --list
  ```

- Display a custom inventory:

  ```
  ansible-inventory --list --inventory {{path/to/file_or_script_or_directory}}
  ```

- Display the default inventory in YAML:

  ```
  ansible-inventory --list --yaml
  ```

- Dump the default inventory to a file:

  ```
  ansible-inventory --list --output {{path/to/file}}
  ```

ansible-playbook

Execute tasks defined in playbook on remote machines over SSH.

More information: https://docs.ansible.com/ansible/latest/cli/ansible-playbook.html.

- Run tasks in playbook:

```
ansible-playbook {{playbook}}
```

- Run tasks in playbook with custom host inventory:

```
ansible-playbook {{playbook}} -i {{inventory_file}}
```

- Run tasks in playbook with extra variables defined via the command-line:

```
ansible-playbook {{playbook}} -e "{{variable1}}={{value1}} {{variable2}}={{value2}}"
```

- Run tasks in playbook with extra variables defined in a json file:

```
ansible-playbook {{playbook}} -e "@{{variables.json}}"
```

- Run tasks in playbook for the given tags:

```
ansible-playbook {{playbook}} --tags {{tag1,tag2}}
```

- Run tasks in a playbook starting at a specific task:

```
ansible-playbook {{playbook}} --start-at {{task_name}}
```

ansible-pull

Pull ansible playbooks from a VCS repo and executes them for the local host.

More information: https://docs.ansible.com/ansible/latest/cli/ansible-pull.html.

- Pull a playbook from a VCS and execute a default local.yml playbook:

  ```
  ansible-pull -U {{repository_url}}
  ```

- Pull a playbook from a VCS and execute a specific playbook:

  ```
  ansible-pull -U {{repository_url}} {{playbook}}
  ```

- Pull a playbook from a VCS at a specific branch and execute a specific playbook:

  ```
  ansible-pull -U {{repository_url}} -C {{branch}} {{playbook}}
  ```

- Pull a playbook from a VCS, specify hosts file and execute a specific playbook:

  ```
  ansible-pull -U {{repository_url}} -i {{hosts_file}}
  {{playbook}}
  ```

ansible-vault

Encrypts & decrypts values, data structures and files within Ansible projects.

More information: https://docs.ansible.com/ansible/latest/user_guide/vault.html#_id17

- Create a new encrypted vault file with a prompt for a password:

  ```
  ansible-vault create {{vault_file}}
  ```

- Create a new encrypted vault file using a vault key file to encrypt it:

  ```
  ansible-vault create --vault-password-file={{password_file}} {{vault_file}}
  ```

- Encrypt an existing file using an optional password file:

  ```
  ansible-vault encrypt --vault-password-file={{password_file}} {{vault_file}}
  ```

- Encrypt a string using Ansible's encrypted string format, displaying interactive prompts:

  ```
  ansible-vault encrypt_string
  ```

- View an encrypted file, using a password file to decrypt:

  ```
  ansible-vault view --vault-password-file={{password_file}} {{vault_file}}
  ```

- Re-key already encrypted vault file with a new password file:

  ```
  ansible-vault rekey --vault-password-file={{old_password_file}} --new-vault-password-file={{new_password_file}} {{vault_file}}
  ```

ansible

Manage groups of computers remotely over SSH.

Use the `/etc/ansible/hosts` file to add new groups/hosts.

More information: https://www.ansible.com/.

- List hosts belonging to a group:

```
ansible {{group}} --list-hosts
```

- Pinging a group of hosts by invoking the ping module:

```
ansible {{group}} -m ping
```

- Display facts about a group of hosts by invoking the setup module:

```
ansible {{group}} -m setup
```

- Execute a command on a group of hosts by invoking command module with arguments:

```
ansible {{group}} -m command -a '{{my_command}}'
```

- Execute a command with administrative privileges:

```
ansible {{group}} --become --ask-become-pass -m command -a '{{my_command}}'
```

- Execute a command using a custom inventory file:

```
ansible {{group}} -i {{inventory_file}} -m command -a '{{my_command}}'
```

- List the groups in an inventory:

```
ansible localhost -m debug -a '{{var=groups.keys()}}'
```

ansiweather

A shell script for displaying the current weather conditions in your terminal.

More information: https://github.com/fcambus/ansiweather.

- Display a forecast using metric units for the next five days for Rzeszow, Poland:

  `ansiweather -u {{metric}} -f {{5}} -l {{Rzeszow,PL}}`

- Display a forecast showing symbols and daylight data for your current location:

  `ansiweather -s {{true}} -d {{true}}`

- Display a forecast showing wind and humidity data for your current location:

  `ansiweather -w {{true}} -h {{true}}`

ant

Apache Ant.

Tool for building and managing Java-based projects.

More information: https://ant.apache.org.

- Build a project with default build file `build.xml`

  `ant`

- Build a project using build file other than `build.xml`

  `ant -f {{buildfile.xml}}`

- Print information on possible targets for this project:

  `ant -p`

- Print debugging information:

  `ant -d`

- Execute all targets that do not depend on fail target(s):

  `ant -k`

apg

Creates arbitrarily complex random passwords.

More information: https://manned.org/apg.

- Create random passwords (default password length is 8):

  ```
  apg
  ```

- Create a password with at least 1 symbol (S), 1 number (N), 1 uppercase (C), 1 lowercase (L):

  ```
  apg -M SNCL
  ```

- Create a password with 16 characters:

  ```
  apg -m {{16}}
  ```

- Create a password with maximum length of 16:

  ```
  apg -x {{16}}
  ```

- Create a password that doesn't appear in a dictionary (the dictionary file has to be provided):

  ```
  apg -r {{dictionary_file}}
  ```

apktool

Reverse engineer APK files.

More information: https://ibotpeaches.github.io/Apktool/.

- Decode an APK file:

  ```
  apktool d {{file.apk}}
  ```

- Build an APK file from a directory:

  ```
  apktool b {{path/to/directory}}
  ```

- Install and store frameworks:

  ```
  apktool if {{framework.apk}}
  ```

apm

Atom editor Package Manager.

See `atom`.

More information: https://github.com/atom/apm.

- Install packages from http://atom.io/packages and themes from http://atom.io/themes:

  `apm install {{package_name}}`

- Remove packages/themes:

  `apm remove {{package_name}}`

- Upgrade packages/themes:

  `apm upgrade {{package_name}}`

apropos

Search the manual pages for names and descriptions.

More information: https://manned.org/apropos.

- Search for a keyword using a regular expression:

```
apropos {{regular_expression}}
```

- Search without restricting the output to the terminal width:

```
apropos -l {{regular_expression}}
```

- Search for pages that contain all of the expressions given:

```
apropos {{regular_expression_1}} -a {{regular_expression_2}}
-a {{regular_expression_3}}
```

ar

Create, modify, and extract from archives (`.a`, `.so`, `.o`).

More information: https://manned.org/ar.

- Extract all members from an archive:

  `ar -x {{path/to/file.a}}`

- List the members of an archive:

  `ar -t {{path/to/file.a}}`

- Replace or add files to an archive:

  `ar -r {{path/to/file.a}} {{path/to/file1.o}} {{path/to/file2.o}}`

- Insert an object file index (equivalent to using `ranlib`):

  `ar -s {{path/to/file.a}}`

- Create an archive with files and an accompanying object file index:

  `ar -rs {{path/to/file.a}} {{path/to/file1.o}} {{path/to/file2.o}}`

arc

Arcanist: A CLI for Phabricator.

More information: https://secure.phabricator.com/book/phabricator/article/arcanist/.

- Send the changes to Differential for review:

  ```
  arc diff
  ```

- Show pending revision information:

  ```
  arc list
  ```

- Update Git commit messages after review:

  ```
  arc am end
  ```

- Push Git changes:

  ```
  arc land
  ```

arch

Display the name of the system architecture.

See also: `uname`.

More information: https://www.gnu.org/software/coreutils/arch.

- Display the system's architecture:

  ```
  arch
  ```

arduino-builder

A command-line tool for compiling arduino sketches.

DEPRECATIONWARNING: This tool is being phased out in favor of `arduino`.

More information: https://github.com/arduino/arduino-builder.

- Compile a sketch:

  `arduino-builder -compile {{path/to/sketch.ino}}`

- Specify the debug level (1 to 10, defaults to 5):

  `arduino-builder -debug-level {{level}}`

- Specify a custom build directory:

  `arduino-builder -build-path {{path/to/build_directory}}`

- Use a build option file, instead of specifying `--hardware`, `--tools`, etc. manually every time:

  `arduino-builder -build-options-file {{path/to/build.options.json}}`

- Enable verbose mode:

  `arduino-builder -verbose {{true}}`

arduino

Arduino Studio - Integrated Development Environment for the Arduino platform.

More information: https://github.com/arduino/Arduino/blob/master/build/shared/manpage.adoc.

- Build a sketch:

  ```
  arduino --verify {{path/to/file.ino}}
  ```

- Build and upload a sketch:

  ```
  arduino --upload {{path/to/file.ino}}
  ```

- Build and upload a sketch to an Arduino Nano with an Atmega328p CPU, connected on port `/dev/ttyACM0`:

  ```
  arduino --board {{arduino:avr:nano:cpu=atmega328p}} --port {{/dev/ttyACM0}} --upload {{path/to/file.ino}}
  ```

- Set the preference `name` to a given `value`:

  ```
  arduino --pref {{name}}={{value}}
  ```

- Build a sketch, put the build results in the build directory, and reuse any previous build results in that directory:

  ```
  arduino --pref build.path={{path/to/build_directory}} --verify {{path/to/file.ino}}
  ```

- Save any (changed) preferences to `preferences.txt`:

  ```
  arduino --save-prefs
  ```

aria2

This command is an alias of `aria2c`.

- View documentation for the updated command:

  `tldr aria2c`

aria2 c

Fast download utility.

Supports HTTP(S), FTP, SFTP, BitTorrent, and Metalink.

More information: https://aria2.github.io

- Download a URI to a file:

```
aria2c {{url}}
```

- Download the file pointed to by the specified URI with the specified output name:

```
aria2c --out={{filename}} {{url}}
```

- Download multiple files in parallel:

```
aria2c --force-sequential {{url_1}} {{url_2}}
```

- Download from multiple sources each URI pointing to the same file:

```
aria2c {{url_1}} {{url_2}}
```

- Download the URIs listed in a file with limited parallel downloads:

```
aria2c --input-file={{filename}} --max-concurrent-downloads={{number_of_downloads}}
```

- Download with multiple connections:

```
aria2c --split={{number_of_connections}} {{url}}
```

- FTP download with username and password:

```
aria2c --ftp-user={{username}} --ftp-passwd={{password}} {{url}}
```

- Limit download speed in bytes/s:

```
aria2c --max-download-limit={{speed}} {{url}}
```

arp

Show and manipulate your system's ARP cache.

More information: https://manned.org/arp.

- Show the current ARP table:

```
arp -a
```

- Clear the entire cache:

```
sudo arp -a -d
```

- Delete a specific entry:

```
arp -d {{address}}
```

- Create an entry in the ARP table:

```
arp -s {{address}} {{mac_address}}
```

arping

Discover and probe hosts in a network using the ARP protocol.

Useful for MAC address discovery.

More information: https://github.com/Thomas Habets/arping

- Ping a host by ARP request packets:

```
arping {{host_ip}}
```

- Ping a host on a specific interface:

```
arping -I {{interface}} {{host_ip}}
```

- Ping a host and stop at the first reply:

```
arping -f {{host_ip}}
```

- Ping a host a specific number of times:

```
arping -c {{count}} {{host_ip}}
```

- Broadcast ARP request packets to update neighbours' ARP caches:

```
arping -U {{ip_to_broadcast}}
```

- Detect duplicated IP addresses in the network by sending ARP requests with a 3 seconds timeout:

```
arping -D -w {{3}} {{ip_to_check}}
```

asar

A file archiver for the Electron platform.

More information: https://github.com/electron/asar.

- Archive a file or directory:

  ```
  asar pack {{path/to/file_or_directory}} {{archived.asar}}
  ```

- Extract an archive:

  ```
  asar extract {{archived.asar}}
  ```

- Extract a specific file from an archive:

  ```
  asar extract-file {{archived.asar}} {{file}}
  ```

- List the contents of an archive file:

  ```
  asar list {{archived.asar}}
  ```

asciinema

Record and replay terminal sessions, and optionally share them on asciinema.org.

More information: https://asciinema.org/.

- Associate the local install of `asciinema` with an asciinema.org account:

  ```
  asciinema auth
  ```

- Make a new recording (once finished, user will be prompted to upload it or save it locally):

  ```
  asciinema rec
  ```

- Make a new recording and save it to a local file:

  ```
  asciinema rec {{path/to/file}}.cast
  ```

- Replay a terminal recording from a local file:

  ```
  asciinema play {{path/to/file}}.cast
  ```

- Replay a terminal recording hosted on asciinema.org:

  ```
  asciinema play https://asciinema.org/a/{{cast_id}}
  ```

- Make a new recording, limiting any idle time to at most 2.5 seconds:

  ```
  asciinema rec -i {{2.5}}
  ```

- Print the full output of a locally saved recording:

  ```
  asciinema cat {{path/to/file}}.cast
  ```

- Upload a locally saved terminal session to asciinema.org:

  ```
  asciinema upload {{path/to/file}}.cast
  ```

asdf

Command-line interface for managing versions of different packages.

More information: https://asdf-vm.com.

- List all available plugins:

  ```
  asdf plu g in- list- all
  ```

- Install a plugin:

  ```
  asdf plu g in- add {{nam e}}
  ```

- List all available versions for a package:

  ```
  asdf list- all {{nam e}}
  ```

- Install a specific version of a package:

  ```
  asdf install {{nam e}} {{v ersion}}
  ```

- S et global version for a package:

  ```
  asdf g lob al {{nam e}} {{v ersion}}
  ```

- S et local version for a package:

  ```
  asdf local {{nam e}} {{v ersion}}
  ```

assimp

Command-line client for the Open Asset Import Library.

Supports loading of 40+ 3D file formats, and exporting to several popular 3D formats.

More information: http://www.assimp.org/.

- List all supported import formats:

  ```
  assimp listext
  ```

- List all supported export formats:

  ```
  assimp listexport
  ```

- Convert a file to one of the supported output formats, using the default parameters:

  ```
  assimp export {{input_file.stl}} {{output_file.obj}}
  ```

- Convert a file using custom parameters (the dox_cmd.h file in assimp's source code lists available parameters):

  ```
  assimp export {{input_file.stl}} {{output_file.obj}} {{parameters}}
  ```

- Display a summary of a 3D file's contents:

  ```
  assimp info {{path/to/file}}
  ```

- List all supported subcommands ("verbs"):

  ```
  assimp help
  ```

- Get help on a specific subcommand (e.g. the parameters specific to it):

  ```
  assimp {{subcommand}} --help
  ```

astronomer

T ool that detects illegitimate stars from bot accounts on G itH ub proj ects.

More information: https://github.com/U llaakut/astronomer

- S can a repository:

  ```
  astronom er {{tldr- pag es/tldr- node- client}}
  ```

- S can the maximum amount of stars in the repository:

  ```
  astronom er {{tldr- pag es/tldr- node- client}} - - stars {{5 0 }}
  ```

- S can a repository including comparative reports:

  ```
  astronom er {{tldr- pag es/tldr- node- client}} - - v erb ose
  ```

astyle

Source code indenter, formatter, and beautifier for the C, C++, C# and Java programming languages.

Upon running, a copy of the original file is created with an ".orig" appended to the original file name.

More information: http://astyle.sourceforge.net/.

- Apply the default style of 4 spaces per indent and no formatting changes:

  ```
  astyle {{source_file}}
  ```

- Apply the Java style with attached braces:

  ```
  astyle --style=java {{path/to/file}}
  ```

- Apply the allman style with broken braces:

  ```
  astyle --style=allman {{path/to/file}}
  ```

- Apply a custom indent using spaces. Choose between 2 and 20 spaces:

  ```
  astyle --indent=spaces={{number_of_spaces}} {{path/to/file}}
  ```

- Apply a custom indent using tabs. Choose between 2 and 20 tabs:

  ```
  astyle --indent=tab={{number_of_tabs}} {{path/to/file}}
  ```

at

Execute commands once at a later time.

Service atd (or atrun) should be running for the actual executions.

More information: https://man.archlinux.org/man/at.1.

- Execute commands from standard input in 5 minutes (press `Ctrl + D` when done):

  ```
  at now + 5 minutes
  ```

- Execute a command from standard input at 10:00 AM today:

  ```
  echo "{{./make_db_backup.sh}}" | at 1000
  ```

- Execute commands from a given file next Tuesday:

  ```
  at -f {{path/to/file}} 9:30 PM Tue
  ```

atom

A cross-platform pluggable text editor.

Plugins are managed by `apm`

More information: https://atom.io/.

- Open a file or directory:

  ```
  atom {{path/to/file_or_directory}}
  ```

- Open a file or directory in a new window:

  ```
  atom -n {{path/to/file_or_directory}}
  ```

- Open a file or directory in an existing window:

  ```
  atom --add {{path/to/file_or_directory}}
  ```

- Open Atom in safe mode (does not load any additional packages):

  ```
  atom --safe
  ```

- Prevent Atom from forking into the background, keeping Atom attached to the terminal:

  ```
  atom --foreground
  ```

- Wait for Atom window to close before returning (useful for Git commit editor):

  ```
  atom --wait
  ```

atoum

A simple, modern and intuitive unit testing framework for PHP.

More information: http://atoum.org.

- Initialise a configuration file:

  ```
  atoum --init
  ```

- Run all tests:

  ```
  atoum
  ```

- Run tests using the specified configuration file:

  ```
  atoum -c {{path/to/file}}
  ```

- Run a specific test file:

  ```
  atoum -f {{path/to/file}}
  ```

- Run a specific directory of tests:

  ```
  atoum -d {{path/to/directory}}
  ```

- Run all tests under a specific namespace:

  ```
  atoum -ns {{namespace}}
  ```

- Run all tests with a specific tag:

  ```
  atoum -t {{tag}}
  ```

- Load a custom bootstrap file before running tests:

  ```
  atoum --bootstrap-file {{path/to/file}}
  ```

atq

Show jobs scheduled by `at` or `batch` commands.

More information: https://man.archlinux.org/man/at.1.

- Show the current user's scheduled jobs:

  ```
  atq
  ```

- Show jobs from queue named 'a' (queues have single-character names):

  ```
  atq -q {{a}}
  ```

- Show jobs of all users (run as super user):

  ```
  sudo atq
  ```

atrm

Remove jobs scheduled by `at` or `batch` commands.

To find job numbers use `atq`.

More information: https://man.archlinux.org/man/at.1.

- Remove job number 10:

  `atrm {{10}}`

- Remove many jobs, separated by spaces:

  `atrm {{15}} {{17}} {{22}}`

autoflake

A tool to remove unused imports and variables from Python code.

More information: https://github.com/myint/autoflake

- Remove unused variables from a single file and display the diff:

  ```
  autoflake --remove-unused-variables {{file.py}}
  ```

- Remove unused imports from multiple files and display the diffs:

  ```
  autoflake --remove-all-unused-imports {{file1.py}} {{file2.py}} {{file3.py}}
  ```

- Remove unused variables from a file, overwriting the file:

  ```
  autoflake --remove-unused-variables --in-place {{file.py}}
  ```

- Remove unused variables recursively from all files in a directory, overwriting each file:

  ```
  autoflake --remove-unused-variables --in-place --recursive {{path/to/directory}}
  ```

autojump

Quickly jump among the directories you visit the most.

Aliases like j or jc are provided for even less typing.

More information: https://github.com/wting/autojump

- Jump to a directory that contains the given pattern:

  ```
  j {{pattern}}
  ```

- Jump to a sub-directory (child) of the current directory that contains the given pattern:

  ```
  j c {{pattern}}
  ```

- Open a directory that contains the given pattern in the operating system file manager:

  ```
  j o {{pattern}}
  ```

- Remove non-existing directories from the autojump database:

  ```
  j --purge
  ```

- Show the entries in the autojump database:

  ```
  j -s
  ```

autopep8

Format Python code according to the PEP 8 style guide.

More information: https://github.com/hhatto/autopep8.

- Format a file to stdout, with a custom maximum line length:

  ```
  autopep8 {{path/to/file.py}} --max-line-length {{length}}
  ```

- Format a file, displaying a diff of the changes:

  ```
  autopep8 --diff {{path/to/file}}
  ```

- Format a file in-pace and save the changes:

  ```
  autopep8 --in-place {{path/to/file.py}}
  ```

- Recursively format all files in a directory in-place and save changes:

  ```
  autopep8 --in-place --recursive {{path/to/directory}}
  ```

autossh

Run, monitor and restart SSH connections.

Auto-reconnects to keep port forwarding tunnels up. Accepts all `ssh` flags.

More information: https://www.harding.motd.ca/autossh.

- Start an SSH session, restarting when a monitoring port fails to return data:

```
autossh -M {{monitor_port}} "{{ssh_command}}"
```

- Forward a local port to a remote one, restarting when necessary:

```
autossh -M {{monitor_port}} -L {{local_port}}:localhost:{{remote_port}} {{user}}@{{host}}
```

- Fork `autossh` into the background before executing `ssh` and don't open a remote shell:

```
autossh -f -M {{monitor_port}} -N "{{ssh_command}}"
```

- Run in the background, with no monitoring port, and instead send SSH keep-alive packets every 10 seconds to detect failure:

```
autossh -f -M 0 -N -o "ServerAliveInterval 10" -o "ServerAliveCountMax 3" "{{ssh_command}}"
```

- Run in the background, with no monitoring port and no remote shell, exiting if the port forward fails:

```
autossh -f -M 0 -N -o "ServerAliveInterval 10" -o "ServerAliveCountMax 3" -o ExitOnForwardFailure=yes -L {{local_port}}:localhost:{{remote_port}} {{user}}@{{host}}
```

- Run in the background, logging `autossh` debug output and `ssh` verbose output to files:

```
AUTOSSH_DEBUG=1 AUTOSSH_LOGFILE={{path/to/autossh_log_file.log}} autossh -f -M {{monitor_port}} -v -E {{path/to/ssh_log_file.log}} {{ssh_command}}
```

avrdude

Driver program for Atmel AVR microcontrollers programming.

More information: https://www.nongnu.org/avrdude/.

- Read AVR microcontroller:

```
avrdude -p {{AVR_device}} -c {{programmer}} -U flash:r:{{file.hex}}:i
```

- Write AVR microcontroller:

```
avrdude -p {{AVR_device}} -c {{programmer}} -U flash:w:{{file.hex}}
```

- List available AVR devices:

```
avrdude -p \?
```

- List available AVR programmers:

```
avrdude -c \?
```

awk

A versatile programming language for working on files.

More information: https://github.com/onetrueawk/awk.

- Print the fifth column (a.k.a. field) in a space-separated file:

```
awk '{print $5}' {{filename}}
```

- Print the second column of the lines containing "foo" in a space-separated file:

```
awk '/{{foo}}/ {print $2}' {{filename}}
```

- Print the last column of each line in a file, using a comma (instead of space) as a field separator:

```
awk -F ',' '{print $NF}' {{filename}}
```

- Sum the values in the first column of a file and print the total:

```
awk '{s+=$1} END {print s}' {{filename}}
```

- Print every third line starting from the first line:

```
awk 'NR%3==1' {{filename}}
```

- Print different values based on conditions:

```
awk '{if ($1 == "foo") print "Exact match foo"; else if ($
"bar") print "Partial match bar"; else print "Baz"}'
{{filename}}
```

- Print all lines where the 10th column value equals the specified value:

```
awk '($10 == value)'
```

- Print all the lines which the 10th column value is between a min and a max:

```
awk '($10 >= min_value && $10 <= max_value)'
```

aws cur

Create, q uery, and delete AWS usage report definitions.

More information: https://awscli.amaz onaws.com/v2 /documentation/api/latest/reference/cur/index.html.

- Create an AWS cost and usage report definition from a J S O N file:

```
aw s cu r pu t- report- definition - - report- definition file: //
{{path/to/report- definition. j son}}
```

- List usage report definitions defined for the logged in account:

```
aw s cu r describ e- report- definitions
```

- Delete a usage report definition:

```
aw s cu r - - reg ion {{aw s_reg ion}} delete- report- definition - -
report- nam e {{report}}
```

aws ec2

CLI for AWS EC2.

Provides secure and resizable computing capacity in the AWS cloud to enable faster development and deployment of applications.

More information: https://awscli.amazonaws.com/v2/documentation/api/latest/reference/ec2/index.html

- Show list of all available EC2 commands:

```
aws ec2 help
```

- Show help for specific EC2 subcommand:

```
aws ec2 {{subcommand}} help
```

- Display information about a specific instance:

```
aws ec2 describe-instances --instance-ids {{instance_id}}
```

- Display information about all instances:

```
aws ec2 describe-instances
```

- Display information about all EC2 volumes:

```
aws ec2 describe-volumes
```

- Delete an EC2 volume:

```
aws ec2 delete-volume --volume-id {{volume_id}}
```

- Create a snapshot from an EC2 volume:

```
aws ec2 create-snapshot --volume-id {{volume_id}}
```

- List available AMIs (Amazon Machine Images):

```
aws ec2 describe-images
```

aws ecr

Push, pull, and manage container images.

More information: https://awscli.amazonaws.com/v2/documentation/api/latest/reference/ecr/index.html.

- Authenticate Docker with the default registry (username is AWS):

```
aws ecr get-login-password --region {{region}} | {{docker login}} --username AWS --password-stdin {{aws_account_id}}.dkr.ecr.{{region}}.amazonaws.com
```

- Create a repository:

```
aws ecr create-repository --repository-name {{repository}} --image-scanning-configuration scanOnPush={{true|false}} --region {{region}}
```

- Tag a local image for ECR:

```
docker tag {{container_name}}:{{tag}} {{aws_account_id}}.dkr.ecr.{{region}}.amazonaws.com/{{container_name}}:{{tag}}
```

- Push an image to a repository:

```
docker push {{aws_account_id}}.dkr.ecr.{{region}}.amazonaws.com/{{container_name}}:{{tag}}
```

- Pull an image from a repository:

```
docker pull {{aws_account_id}}.dkr.ecr.{{region}}.amazonaws.com/{{container_name}}:{{tag}}
```

- Delete an image from a repository:

```
aws ecr batch-delete-image --repository-name {{repository}} --image-ids imageTag={{latest}}
```

- Delete a repository:

```
aws ecr delete-repository --repository-name {{repository}} --force
```

- List images within a repository:

```
aws ecr list-images --repository-name {{repository}}
```

aws glue

CLI for AWS Glue.

Defines the public endpoint for the AWS Glue service.

More information: https://docs.aws.amazon.com/cli/latest/reference/glue/

- List jobs:

  ```
  aws glue list-jobs
  ```

- Start a job:

  ```
  aws glue start-job-run --job-name {{job_name}}
  ```

- Start running a workflow:

  ```
  aws glue start-workflow-run --name {{workflow_name}}
  ```

- List triggers:

  ```
  aws glue list-triggers
  ```

- Start a trigger:

  ```
  aws glue start-trigger --name {{trigger_name}}
  ```

- Create a dev endpoint:

  ```
  aws glue create-dev-endpoint --endpoint-name {{name}} --role-
  arn {{role_arn_used_by_endpoint}}
  ```

aws-google-auth

Command-line tool to acquire AWS temporary (STS) credentials using Google Apps as a federated (Single Sign-On) provider.

More information: https://github.com/cevoaustralia/aws-google-auth.

- Log in with Google SSO using the IDP and SP identifiers and set the credentials duration to one hour:

```
aws-google-auth -u {{example@example.com}} -I
{{$GOOGLE_IDP_ID}} -S {{$GOOGLE_SP_ID}} -d {{3600}}
```

- Log in [a]sking which role to use (in case of several available SAML roles):

```
aws-google-auth -u {{example@example.com}} -I
{{$GOOGLE_IDP_ID}} -S {{$GOOGLE_SP_ID}} -d {{3600}}
```

- Resolve aliases for AWS accounts:

```
aws-google-auth -u {{example@example.com}} -I
{{$GOOGLE_IDP_ID}} -S {{$GOOGLE_SP_ID}} -d {{3600}}
resolve-aliases
```

- Show help information:

```
aws-google-auth -h
```

aws iam

CLI for AWS IAM.

More information: https://awscli.amazonaws.com/v2/documentation/api/latest/reference/iam/index.html.

- Show `aws iam` help page (including all available iam commands):

  ```
  aws iam help
  ```

- List users:

  ```
  aws iam list-users
  ```

- List policies:

  ```
  aws iam list-policies
  ```

- List groups:

  ```
  aws iam list-groups
  ```

- Get users in a group:

  ```
  aws iam get-group --group-name {{group_name}}
  ```

- Describe an IAM policy:

  ```
  aws iam get-policy --policy-arn arn:aws:iam::aws:policy/{{policy_name}}
  ```

- List access keys:

  ```
  aws iam list-access-keys
  ```

- List access keys for a specific user:

  ```
  aws iam list-access-keys --user-name {{user_name}}
  ```

aws kinesis

Official AWS CLI for Amazon Kinesis streaming data services.

More information: https://docs.aws.amazon.com/cli/latest/reference/kinesis/index.html#cli-aws-kinesis

- Show all streams in the account:

```
aws kinesis list-streams
```

- Write one record to a Kinesis stream:

```
aws kinesis put-record --stream-name {{name}} --partition-key {{key}} --data {{base64_encoded_message}}
```

- Write a record to a Kinesis stream with inline base64 encoding:

```
aws kinesis put-record --stream-name {{name}} --partition-key {{key}} --data "$(echo "{{my raw message}}" | base64)"
```

- List the shards available on a stream:

```
aws kinesis list-shards --stream-name {{name}}
```

- Get a shard iterator for reading from the oldest message in a stream's shard:

```
aws kinesis get-shard-iterator --shard-iterator-type TRIM_HORIZON --stream-name {{name}} --shard-id {{id}}
```

- Read records from a shard, using a shard iterator:

```
aws kinesis get-records --shard-iterator {{iterator}}
```

aws lambda

CLI for AWS lambda.

More information: https://docs.aws.amazon.com/cli/latest/reference/lambda/

- Run a function:

```
aws lambda invoke --function-name {{name}} {{path/to/response}}.json
```

- Run a function with an input payload in JSON format:

```
aws lambda invoke --function-name {{name}} --payload {{json}}
{{path/to/response}}.json
```

- List functions:

```
aws lambda list-functions
```

- Display the configuration of a function:

```
aws lambda get-function-configuration --function-name
{{name}}
```

- List function aliases:

```
aws lambda list-aliases --function-name {{name}}
```

- Display the reserved concurrency configuration for a function:

```
aws lambda get-function-concurrency --function-name {{name}}
```

- List which AWS services can invoke the function:

```
aws lambda get-policy --function-name {{name}}
```

aws q uicksight

CLI for AWS Q uickS ight.

Access Q uickS ight entities.

More information: https://docs.aws.amaz on.com/cli/latest/reference/q uicksight/

- List datasets:

```
aw s q u ick sig ht list- data- sets - - aw s- accou nt- id
{{aw s_accou nt_id}}
```

- List users:

```
aw s q u ick sig ht list- u sers - - aw s- accou nt- id {{aw s_accou nt_id}
- - nam espace defau lt
```

- List groups:

```
aw s q u ick sig ht list- g rou ps - - aw s- accou nt- id
{{aw s_accou nt_id}} - - nam espace defau lt
```

- List dashboards:

```
aw s q u ick sig ht list- dashb oards - - aw s- accou nt- id
{{aw s_accou nt_id}}
```

- Display detailed information about a dataset:

```
aw s q u ick sig ht describ e- data- set - - aw s- accou nt- id
{{aw s_accou nt_id}} - - data- set- id {{data_set_id}}
```

- Display who has access to the dataset and what kind of actions they can perform on the dataset:

```
aw s q u ick sig ht describ e- data- set- perm issions - - aw s- accou nt-
{{aw s_accou nt_id}} - - data- set- id {{data_set_id}}
```

aws s3

CLI for AWS S 3 - provides storage through web services interfaces.

More information: https://aws.amaz on.com/cli

- S how files in a bucket:

  ```
  aw s s3 ls {{b u ck et_nam e}}
  ```

- S ync files and directories from local to bucket:

  ```
  aw s s3 sync {{path/to/files}} s3 : //{{b u ck et_nam e}}
  ```

- S ync files and directories from bucket to local:

  ```
  aw s s3 sync s3 : //{{b u ck et_nam e}} {{path/to/targ et}}
  ```

- S ync files and directories with exclusions:

  ```
  aw s s3 sync {{path/to/files}} s3 : //{{b u ck et_nam e}} - - exclu de
  {{path/to/file}} - - exclu de {{path/to/directory}}/*
  ```

- R emove file from bucket:

  ```
  aw s s3 rm s3 : //{{b u ck et}}/{{path/to/file}}
  ```

- P review changes only:

  ```
  aw s s3 {{any_com m and}} - - dryru n
  ```

aws s3 api

Create and delete Amazon S3 buckets and edit bucket properties.

More information: https://awscli.amazonaws.com/v2/documentation/api/latest/reference/s3api/index.html

- Create a bucket:

```
aws s3 api create-bucket --bucket {{bucket_name}}
```

- Delete a bucket:

```
aws s3 api delete-bucket --bucket {{bucket_name}}
```

- List buckets:

```
aws s3 api list-buckets
```

- List the objects inside of a bucket and only show each object's key and size:

```
aws s3 api list-objects --bucket {{bucket_name}} --query
' {{Contents[].{Key: Key, Size: Size}}}'
```

- Add an object to a bucket:

```
aws s3 api put-object --bucket {{bucket_name}} --key
{{object_key}} --body {{path/to/file}}
```

- Download object from a bucket (The output file is always the last argument):

```
aws s3 api get-object --bucket {{bucket_name}} --key
{{object_key}} {{path/to/output_file}}
```

- Apply an Amazon S3 bucket policy to a specified bucket:

```
aws s3 api put-bucket-policy --bucket {{bucket_name}} --po
file://{{path/to/bucket_policy.json}}
```

- Download the Amazon S3 bucket policy from a specified bucket:

```
aws s3 api get-bucket-policy --bucket {{bucket_name}} --q
Policy --output {{json|table|text|yaml|yaml-stream}} >
{{path/to/bucket_policy}}
```

aws secretsmanager

Store, manage, and retrieve secrets.

More information: https://docs.aws.amazon.com/cli/latest/reference/secretsmanager/

- Show secrets stored by the secrets manager in the current account:

  ```
  aws secretsmanager list-secrets
  ```

- Create a secret:

  ```
  aws secretsmanager create-secret --name {{name}} --description "{{secret_description}}" --secret-string {{secret}}
  ```

- Delete a secret:

  ```
  aws secretsmanager delete-secret --secret-id {{name_or_arn}}
  ```

- View details of a secret except for secret text:

  ```
  aws secretsmanager describe-secret --secret-id {{name_or_arn}}
  ```

- Retrieve the value of a secret (to get the latest version of the secret omit `--version-stage`):

  ```
  aws secretsmanager get-secret-value --secret-id {{name_or_arn}} --version-stage {{version_of_secret}}
  ```

- Rotate the secret immediately using a Lambda function:

  ```
  aws secretsmanager rotate-secret --secret-id {{name_or_arn}} --rotation-lambda-arn {{arn_of_lambda_function}}
  ```

- Rotate the secret automatically every 30 days using a Lambda function:

  ```
  aws secretsmanager rotate-secret --secret-id {{name_or_arn}} --rotation-lambda-arn {{arn_of_lambda_function}} --rotation-rules AutomaticallyAfterDays={{30}}
  ```

aws-vault

A vault for securely storing and accessing AWS credentials in development environments.

More information: https://github.com/99designs/aws-vault

- Add credentials to the secure keystore:

  ```
  aws-vault add {{profile}}
  ```

- Execute a command with AWS credentials in the environment:

  ```
  aws-vault exec {{profile}} -- {{aws s3 ls}}
  ```

- Open a browser window and login to the AWS Console:

  ```
  aws-vault login {{profile}}
  ```

- List profiles, along with their credentials and sessions:

  ```
  aws-vault list
  ```

- Rotate AWS credentials:

  ```
  aws-vault rotate {{profile}}
  ```

- Remove credentials from the secure keystore:

  ```
  aws-vault remove {{profile}}
  ```

aws

The official CLI tool for Amazon Web Services.

Wizard, SSO, Resource Autocompletion, and YAML options are v2 only.

More information: https://aws.amazon.com/cli

- Configure the AWS Command-line:

  ```
  aws configure wizard
  ```

- Configure the AWS Command-line using SSO:

  ```
  aws configure sso
  ```

- See help text for the AWS command:

  ```
  aws {{command}} help
  ```

- Get the caller identity (used to troubleshoot permissions):

  ```
  aws sts get-caller-identity
  ```

- List AWS resources in a region and output in yaml:

  ```
  aws dynamodb list-tables --region {{us-east-1}} --output ya
  ```

- Use auto prompt to help with a command:

  ```
  aws iam create-user --cli-auto-prompt
  ```

- Get an interactive wizard for an AWS resource:

  ```
  aws dynamodb wizard {{new_table}}
  ```

- Generate a JSON CLI Skeleton (useful for infrastructure as code):

  ```
  aws dynamodb update-table --generate-cli-skeleton
  ```

awslogs

Q ueries groups, streams and events from Amaz on CloudWatch logs.

More information: https://github.com/j orgebastida/awslogs

- List log groups:

  ```
  aw slog s  g rou ps
  ```

- List existing streams for the specified group:

  ```
  aw slog s  stream s {{/v ar/log /syslog }}
  ```

- G et logs for any streams in the specified group between 1 and 2 hours ago:

  ```
  aw slog s  g et {{/v ar/log /syslog }} - - start= ' {{2 h  ag o}}'  - -
  end= ' {{1 h  ag o}}'
  ```

- G et logs that match a specific CloudWatch Logs F ilter pattern:

  ```
  aw slog s  g et {{/aw s/lam b da/m y_lam b da_g rou p}} - - filter-
  pattern= ' {{E R R O R }}'
  ```

- Watch logs for any streams in the specified group:

  ```
  aw slog s  g et {{/v ar/log /syslog }} A L L  - - w atch
  ```

axel

Download accelerator.

S upports H T T P , H T T P S , and F T P .

More information: https://github.com/axel-download-accelerator/axel.

- Download a U R L to a file:

```
axel {{u rl}}
```

- Download and specify filename:

```
axel {{u rl}} - o {{filenam e}}
```

- Download with multiple connections:

```
axel - n {{connections_nu m }} {{u rl}}
```

- S earch for mirrors:

```
axel - S {{m irrors_nu m }} {{u rl}}
```

- Limit download speed (bytes per second):

```
axel - s {{speed}} {{u rl}}
```

az account

Manage Azure subscription information.

Part of `az`, the command-line client for Microsoft Azure.

More information: https://docs.microsoft.com/cli/azure/account

- Print a list of subscriptions for the logged in account:

  ```
  az account list
  ```

- Set a `subscription` to be the currently active subscription:

  ```
  az account set --subscription {{subscription_id}}
  ```

- List supported regions for the currently active subscription:

  ```
  az account list-locations
  ```

- Print an access token to be used with `MS Graph API`:

  ```
  az account get-access-token --resource-type {{ms-graph}}
  ```

- Print details of the currently active subscription in a specific format:

  ```
  az account show --output {{json|tsv|table|yaml}}
  ```

az login

Log in to Azure.

Part of `az`, the command-line client for Microsoft Azure.

More information: https://docs.microsoft.com/cli/azure/reference-index#az_login

- Log in interactively:

```
az login
```

- Log in with a service principal using a client secret:

```
az login --service-principal --username {{http://azure-cli-service-principal}} --password {{secret}} --tenant {{someone.onmicrosoft.com}}
```

- Log in with a service principal using a client certificate:

```
az login --service-principal --username {{http://azure-cli-service-principal}} --password {{path/to/cert.pem}} --tenant {{someone.onmicrosoft.com}}
```

- Log in using a VM's system assigned identity:

```
az login --identity
```

- Log in using a VM's user assigned identity:

```
az login --identity --username /subscriptions/{{subscription_id}}/resourcegroups/{{my_rg}}/providers/Microsoft.Managedidentity/userAssignedIdentities/{{my_id}}
```

az logout

Log out from an Azure subscription.

Part of **`azure-cli`**

More information: https://docs.microsoft.com/cli/azure/reference-index#az_logout

- Log out from the active account:

  ```
  az logout
  ```

- Log out a specific username:

  ```
  az logout --username {{alias@somedomain.com}}
  ```

az vm

Manage virtual machines in Azure.

Part of `az`, the command-line client for Microsoft Azure.

More information: https://docs.microsoft.com/cli/azure/vm.

- List details of availible Virtual Machines:

```
az vm list
```

- Create an `Ubuntu Server 18.04 LTS` Virtual Machine and generate ssh keys:

```
az vm create --resource-group {{rg}} --name {{vm_name}} --
image {{Canonical:Ubuntu Server:18.04-LTS:latest}} --adm
user {{azureuser}} --generate-ssh-keys
```

- Stop a Virtual Machine:

```
az vm stop --resource-group {{rg}} --name {{vm_name}}
```

- Deallocate a Virtual Machine:

```
az vm deallocate --resource-group {{rg}} --name {{vm_name}}
```

- Start a Virtual Machine:

```
az vm start --resource-group {{rg}} --name {{vm_name}}
```

- Restart a Virtual Machine:

```
az vm restart --resource-group {{rg}} --name {{vm_name}}
```

- List VM images availiable in the Azure Marketplace:

```
az vm image list
```

az

The official CLI tool for Microsoft Azure.

More information: https://docs.microsoft.com/cli/azure

- Log in to Azure:

  ```
  az login
  ```

- Manage azure subscription information:

  ```
  az account
  ```

- List all Azure Managed Disks:

  ```
  az disk list
  ```

- List all Azure virtual machines:

  ```
  az vm list
  ```

- Manage Azure Kubernetes Services:

  ```
  az aks
  ```

- Manage Azure Network resources:

  ```
  az network
  ```

b2sum

Calculate BLAKE2 cryptographic checksums.

More information: https://www.gnu.org/software/coreutils/b2sum

- Calculate the BLAKE2 checksum for a file:

```
b2sum {{filename1}}
```

- Calculate BLAKE2 checksums for multiple files:

```
b2sum {{filename1}} {{filename2}}
```

- Read a file of BLAKE2 sums and filenames and verify all files have matching checksums:

```
b2sum -c {{filename.b2}}
```

- Calculate the BLAKE2 checksum from stdin:

```
{{somecommand}} | b2sum
```

babel

A transpiler which converts code from JavaScript ES6/ES7 syntax to ES5 syntax.

More information: https://babelj s.io/

- Transpile a specified input file and output to stdout:

  ```
  babel {{path/to/file}}
  ```

- Transpile a specified input file and output to a specific file:

  ```
  babel {{path/to/input_file}} --output-file {{path/to/output_file}}
  ```

- Transpile the input file every time it is changed:

  ```
  babel {{path/to/input_file}} --watch
  ```

- Transpile a whole directory of files:

  ```
  babel {{path/to/input_directory}}
  ```

- Ignore specified comma-separated files in a directory:

  ```
  babel {{path/to/input_directory}} --ignore {{ignored_files}}
  ```

- Transpile and output as minified JavaScript:

  ```
  babel {{path/to/input_file}} --minified
  ```

- Choose a set of presets for output formatting:

  ```
  babel {{path/to/input_file}} --presets {{presets}}
  ```

- Output all available options:

  ```
  babel --help
  ```

badblocks

Search a device for bad blocks.

Some usages of badblocks can cause destructive actions, such as erasing all data on a disk, including the partition table.

More information: https://manned.org/badblocks.

- Search a disk for bad blocks by using a non-destructive read-only test:

```
sudo badblocks {{/dev/sdX}}
```

- Search an unmounted disk for bad blocks with a non-destructive read-write test:

```
sudo badblocks -n {{/dev/sdX}}
```

- Search an unmounted disk for bad blocks with a destructive write test:

```
sudo badblocks -w {{/dev/sdX}}
```

- Search an unmounted disk for bad blocks with a destructive write test and show verbose status:

```
sudo badblocks -sv w {{/dev/sdX}}
```

- Search an unmounted disk in destructive mode and output found blocks to a file:

```
sudo badblocks -o {{path/to/file}} -w {{/dev/sdX}}
```

- Search an unmounted disk in destructive mode with improved speed using 4K block size and 64K block count:

```
sudo badblocks -w -b {{4096}} -c {{65536}} {{/dev/sdX}}
```

balena

Interact with the balenaCloud, openB alena and the balena AP I from the command-line.

More information: https://www.balena.io/docs/reference/cli/.

- Log in to the balenaCloud account:

  ```
  b alena log in
  ```

- Create a balenaCloud or openB alena application:

  ```
  b alena app create {{app_nam e}}
  ```

- List all balenaCloud or openB alena applications within the account:

  ```
  b alena apps
  ```

- List all devices associated with the balenaCloud or openB alena account:

  ```
  b alena dev ices
  ```

- F lash a balenaO S image to a local drive:

  ```
  b alena local flash {{path/to/b alenaos. im g }} - - driv e
  {{driv e_location}}
  ```

bandwhich

Display the current network utilization by process, connection or remote IP/hostname.

More information: https://github.com/imsnif/bandwhich.

- Show the remote addresses table only:

  ```
  bandwhich --addresses
  ```

- Show DNS queries:

  ```
  bandwhich --show-dns
  ```

- Show total (cumulative) usage:

  ```
  bandwhich --total-utilization
  ```

- Show the network utilization for a specific network interface:

  ```
  bandwhich --interface {{eth0}}
  ```

- Show DNS queries with a given DNS server:

  ```
  bandwhich --show-dns --dns-server {{dns_server_ip}}
  ```

banner

Print the given argument as a large ASCII art.

More information: https://man.archlinux.org/man/banner.1.

- Print the text message as a large banner (quotes are optional):

```
banner "{{Hello World}}"
```

- Print the text message as a banner with a width of 50 characters:

```
banner -w {{50}} "{{Hello World}}"
```

- Read text from stdin:

```
banner
```

base32

Encode or decode file or standard input to/from Base32, to standard output.

More information: https://www.gnu.org/software/coreutils/base32

- Encode a file:

```
base32 {{filename}}
```

- Decode a file:

```
base32 --decode {{filename}}
```

- Encode from stdin:

```
{{some command}} | base32
```

- Decode from stdin:

```
{{some command}} | base32 --decode
```

base64

Encode or decode file or standard input to/from Base64, to standard output.

More information: https://www.gnu.org/software/coreutils/base64

- Encode the contents of a file as base64 and write the result to stdout:

```
base64 {{filename}}
```

- Decode the base64 contents of a file and write the result to stdout:

```
base64 --decode {{filename}}
```

- Encode from stdin:

```
{{some command}} | base64
```

- Decode from stdin:

```
{{some command}} | base64 --decode
```

basename

R emove leading directory portions from a path.

More information: https://www.gnu.org/software/coreutils/basename.

- S how only the file name from a path:

  ```
  b asenam e {{path/to/file}}
  ```

- S how only the rightmost directory name from a path:

  ```
  b asenam e {{path/to/directory/}}
  ```

- S how only the file name from a path, with a suffix removed:

  ```
  b asenam e {{path/to/file}} {{su ffix}}
  ```

bash

Bourne-Again Shell, an `sh`-compatible command-line interpreter.

See also `history expand` for history expansion.

More information: https://gnu.org/software/bash/.

- Start an interactive shell session:

  ```
  bash
  ```

- Execute a command and then exit:

  ```
  bash -c "{{command}}"
  ```

- Execute a script:

  ```
  bash {{path/to/script.sh}}
  ```

- Execute a script, printing each command before executing it:

  ```
  bash -x {{path/to/script.sh}}
  ```

- Execute commands from a script, stopping at the first error:

  ```
  bash -e {{path/to/script.sh}}
  ```

- Read and execute commands from stdin:

  ```
  bash -s
  ```

- Print the Bash version (`$BASH_VERSION` contains the version without license information):

  ```
  bash --version
  ```

bashmarks

Save and jump to commonly used directories using 1 character commands.

More information: https://github.com/huyng/bashmarks.

- List available bookmarks:

  ```
  l
  ```

- Save the current directory as "bookmark_name":

  ```
  s {{bookmark_name}}
  ```

- Go to a bookmarked directory:

  ```
  g {{bookmark_name}}
  ```

- Print a bookmarked directory's contents:

  ```
  p {{bookmark_name}}
  ```

- Delete a bookmark:

  ```
  d {{bookmark_name}}
  ```

bastet

Clone of the game Tetris in the terminal.

More information: https://fph.altervista.org/prog/bastet.html.

- Start a tetris game:

  ```
  bastet
  ```

- Navigate the piece horizontally:

  ```
  {{Left|Right}} arrow key
  ```

- Rotate the piece clockwise or counterclockwise:

  ```
  {{Spacebar|Up arrow key}}
  ```

- Soft drop the piece:

  ```
  Down arrow key
  ```

- Hard drop the piece:

  ```
  Enter
  ```

- Pause the game:

  ```
  p
  ```

- Quit the game:

  ```
  Ctrl + C
  ```

bat

Print and concatenate files.

A `cat` clone with syntax highlighting and Git integration.

More information: https://github.com/sharkdp/bat.

- Print the contents of a file to the standard output:

  `bat {{file}}`

- Concatenate several files into the target file:

  `bat {{file1}} {{file2}} > {{target_file}}`

- Append several files into the target file:

  `bat {{file1}} {{file2}} >> {{target_file}}`

- Number all output lines:

  `bat -n {{file}}`

- Syntax highlight a json file:

  `bat --language=json {{file.json}}`

- Display all supported languages:

  `bat --list-languages`

batch

Execute commands at a later time when the system load levels permit.

Service atd (or atrun) should be running for the actual executions.

More information: https://man.archlinux.org/man/at.1.

- Execute commands from standard input (press `Ctrl + D` when done):

  `batch`

- Execute a command from standard input:

  `echo "{{./make_db_backup.sh}}" | batch`

- Execute commands from a given file:

  `batch -f {{path/to/file}}`

bc

An arbitrary precision calculator language.

More information: https://manned.org/bc.

- Start `bc` in interactive mode using the standard math library:

```
bc -l
```

- Calculate the result of an expression:

```
bc <<< "(1 + 2) * 2 ^ 2"
```

- Calculate the result of an expression and force the number of decimal places to `10`:

```
bc <<< "scale=10; 5 / 3"
```

- Calculate the result of an expression with sine and cosine using `mathlib`:

```
bc -l <<< "s(1) + c(1)"
```

beanstalkd

A simple and generic work-queue server.

More information: https://beanstalkd.github.io/.

- Start beanstalkd, listening on port 1 1 3 0 0 :

  ```
  beanstalkd
  ```

- Start beanstalkd listening on a custom port and address:

  ```
  beanstalkd -l {{ip_address}} -p {{port_number}}
  ```

- Persist work queues by saving them to disk:

  ```
  beanstalkd -b {{path/to/persistence_directory}}
  ```

- Sync to the persistence directory every 5 0 0 milliseconds:

  ```
  beanstalkd -b {{path/to/persistence_directory}} -f {{500}}
  ```

bedtools

A swiss-army knife of tools for genomic-analysis tasks.

Used to intersect, group, convert and count data in BAM, BED, GFF/GTF, VCF format.

More information: https://bedtools.readthedocs.io/en/latest/.

- Intersect two files with respect to the sequences' strand and save the result to {{path/to/output_file}}:

```
bedtools intersect -a {{path/to/file_1}} -b {{path/to/file_2}} -s > {{path/to/output_file}}
```

- Intersect two files with a left outer join, i.e. report each feature from {{file_1}} and NULL if no overlap with {{file_2}}:

```
bedtools intersect -a {{path/to/file_1}} -b {{path/to/file_2}} -lof > {{path/to/output_file}}
```

- Using more efficient algorithm to intersect two pre-sorted files:

```
bedtools intersect -a {{path/to/file_1}} -b {{path/to/file_2}} -sorted > {{path/to/output_file}}
```

- Group file {{path/to/file}} based on the first three and the fifth column and summarize the sixth column by summing it up:

```
bedtools groupby -i {{path/to/file}} -c 1-3,5 -g 6 -o sum
```

- Convert bam-formatted file to a bed-formatted one:

```
bedtools bamtobed -i {{path/to/file}}.bam > {{path/to/file}}.bed
```

- Find for all features in {{file_1}}.bed the closest one in {{file_2}}.bed and write distance in an extra column (input files must be sorted):

```
bedtools closest -a {{path/to/file_1}}.bed -b {{path/to/file_2}}.bed -d
```

behat

A PHP framework for Behaviour-Driven Development.

More information: https://behat.org.

- Initialise a new Behat project:

  ```
  behat --init
  ```

- Run all tests:

  ```
  behat
  ```

- Run all tests from the specified suite:

  ```
  behat --suite={{suite_name}}
  ```

- Run tests with a specific output formatter:

  ```
  behat --format {{pretty|progress}}
  ```

- Run tests and output results to a file:

  ```
  behat --out {{path/to/file}}
  ```

- Display a list of definitions in your test suites:

  ```
  behat --definitions
  ```

berks

Chef cookbook dependency manager.

More information: https://docs.chef.io/berkshelf.html.

- Install cookbook dependencies into a local repo:

  ```
  b erk s install
  ```

- U pdate a specific cookbook and its dependencies:

  ```
  b erk s u pdate {{cook b ook }}
  ```

- U pload a cookbook to the Chef server:

  ```
  b erk s u pload {{cook b ook }}
  ```

- V iew the dependencies of a cookbook:

  ```
  b erk s conting ent {{cook b ook }}
  ```

betty

Use natural language to execute commands.

More information: https://github.com/pickhardt/betty.

- Ask Betty something:

```
betty {{what time is it}}
```

- Ask Betty version:

```
betty version
```

- Download a file:

```
betty download {{https://example.com/file.png}} to {{file.png}}
```

- Compress a file or directory to one of the support archive formats:

```
betty {{zip}} {{path/to/file_or_directory}}
```

- Extract an archive into the current directory:

```
betty {{unzip}} {{archive.tar.gz}}
```

- Extract an archive into a directory:

```
betty unarchive {{archive.tar.gz}} to {{directory}}
```

- Play Spotify:

```
betty play {{Spotify}}
```

- Drive Betty to madness:

```
betty go crazy
```

bfg

Remove large files or passwords from Git history like git-filter-branch.

Note: if your repository is connected to a remote, you will need to force push to it.

More information: https://rtyley.github.io/bfg-repo-cleaner/.

- Remove a file with sensitive data but leave the latest commit untouched:

  ```
  bfg --delete-files {{file_with_sensitive_data}}
  ```

- Remove all text mentioned in the specified file wherever it can be found in the repository's history:

  ```
  bfg --replace-text {{path/to/file.txt}}
  ```

bg

Resumes jobs that have been suspended (e.g. using `Ctrl+Z`), and keeps them running in the background.

More information: https://manned.org/bg.

- Resume the most recently suspended job and run it in the background:

  `bg`

- Resume a specific job (use `jobs -l` to get its ID) and run it in the background:

  `bg %{{job_id}}`

bison

GNU parser generator.

More information: https://www.gnu.org/software/bison/.

- Compile a bison definition file:

```
bison {{path/to/file.y}}
```

- Compile in debug mode, which causes the resulting parser to write additional information to the standard output:

```
bison --debug {{path/to/file.y}}
```

- Specify the output filename:

```
bison --output {{path/to/output.c}} {{path/to/file.y}}
```

- Be verbose when compiling:

```
bison --verbose
```

bitcoin-cli

Command-line client to interact with the Bitcoin daemon via RPC calls.

Uses the configuration defined in `bitcoin.conf`

More information: https://en.bitcoin.it/wiki/Running_Bitcoin#Command-line_arguments

- Send a transaction to a given address:

  ```
  bitcoin-cli sendtoaddress "{{address}}" {{amount}}
  ```

- Generate one or more blocks:

  ```
  bitcoin-cli generate {{num_blocks}}
  ```

- Print high-level information about the wallet:

  ```
  bitcoin-cli getwalletinfo
  ```

- List all outputs from previous transactions available to fund outgoing transactions:

  ```
  bitcoin-cli listunspent
  ```

- Export the wallet information to a text file:

  ```
  bitcoin-cli dumpwallet "{{path/to/file}}"
  ```

black

A Python auto code formatter.

More information: https://github.com/psf/black.

- Auto-format a file or entire directory:

  ```
  black {{path/to/file_or_directory}}
  ```

- Format the code passed in as a string:

  ```
  black -c {{path/to/file_or_directory}}
  ```

- Output a diff for each file on stdout:

  ```
  black --diff {{path/to/file_or_directory}}
  ```

- Return the status without writing the files back:

  ```
  black --check {{path/to/file_or_directory}}
  ```

- Auto-format a file or directory emitting exclusively error messages to stderr:

  ```
  black --quiet {{path/to/file_or_directory}}
  ```

blackfire

A command-line profiling tool for PHP.

More information: https://blackfire.io.

- Initialise and configure the Blackfire client:

  ```
  blackfire config
  ```

- Launch the Blackfire agent:

  ```
  blackfire agent
  ```

- Launch the Blackfire agent on a specific socket:

  ```
  blackfire agent --socket="{{tcp://127.0.0.1:8307}}"
  ```

- Run the profiler on a specific program:

  ```
  blackfire run {{php path/to/file.php}}
  ```

- Run the profiler and collect 10 samples:

  ```
  blackfire --samples={{10}} run {{php path/to/file.php}}
  ```

- Run the profiler and output results as JSON:

  ```
  blackfire --json run {{php path/to/file.php}}
  ```

- Upload a profiler file to the Blackfire web service:

  ```
  blackfire upload {{path/to/file}}
  ```

- View the status of profiles on the Blackfire web service:

  ```
  blackfire status
  ```

blender

Command-line interface to the Blender 3D computer graphics application.

Arguments are executed in the order they are given.

More information: https://docs.blender.org/manual/en/latest/advanced/command_line/

- Render all frames of an animation in the background, without loading the UI (output is saved to `/tmp`):

  ```
  blender -b {{filename}}.blend -a
  ```

- Render an animation using a specific image naming pattern, in a path relative (`//`) to the .blend file:

  ```
  blender -b {{filename}}.blend -o //{{render/frame_###.png}} -a
  ```

- Render the 10th frame of an animation as a single image, saved to an existing directory (absolute path):

  ```
  blender -b {{filename}}.blend -o {{/path/to/output_directory}} -f {{10}}
  ```

- Render the second last frame in an animation as a JPEG image, saved to an existing directory (relative path):

  ```
  blender -b {{filename}}.blend -o //{{output_directory}} -F {{JPEG}} -f {{-2}}
  ```

- Render the animation of a specific scene, starting at frame 10 and ending at frame 500:

  ```
  blender -b {{filename}}.blend -S {{scene_name}} -s {{10}} -e {{500}} -a
  ```

- Render an animation at a specific resolution, by passing a Python expression:

  ```
  blender -b {{filename}}.blend --python-expr '{{import bpy; bpy.data.scenes[0].render.resolution_percentage = 25}}' -a
  ```

- Start an interactive Blender session in the terminal with a python console (do `import bpy` after starting):

  ```
  blender -b --python-console
  ```

blockout2

Tetris like game in 3D.

More information: http://www.blockout.net/blockout2/

- Start a new game:

  `blockout2`

- Navigate the current piece on a 2D plane:

  `{{Up|Down|Left|Right}} arrow keys`

- Rotate the piece on its axis:

  `{{Q|W|E|A|S|D}}`

- Hard drop the current piece:

  `Spacebar`

- Pause/unpause the game:

  `p`

bmaptool

Create or copy block maps intelligently (designed to be faster than **cp** or **dd**).

More information: https://source.tiz en.org/documentation/reference/bmaptool

- Create a blockmap from image file:

  ```
  b m aptool create - o {{b lock m ap. b m ap}} {{sou rce. im g }}
  ```

- Copy an image file into sdb:

  ```
  b m aptool copy - - b m ap {{b lock m ap. b m ap}} {{sou rce. im g }} {{/de
  sdb }}
  ```

- Copy a compressed image file into sdb:

  ```
  b m aptool copy - - b m ap {{b lock m ap. b m ap}} {{sou rce. im g . g z}} {
  dev /sdb }}
  ```

- Copy an image file into sdb without using a blockmap:

  ```
  b m aptool copy - - nob m ap {{sou rce. im g }} {{/dev /sdb }}
  ```

boot

Build tooling for the Clojure programming language.

More information: https://github.com/boot-clj/boot

- Start a REPL session either with the project or standalone:

  ```
  boot repl
  ```

- Build a single uberjar

  ```
  boot jar
  ```

- Learn about a command:

  ```
  boot clj s --help
  ```

- Generate scaffolding for a new project based on a template:

  ```
  boot --dependencies boot/new new --template {{template_name}}
  --name {{project_name}}
  ```

- Build for development (if using the boot/new template):

  ```
  boot dev
  ```

- Build for production (if using the boot/new template):

  ```
  boot prod
  ```

bootctl

Control EFI firmware boot settings and manage boot loader.

More information: https://man.archlinux.org/man/bootctl.1.

- Show information about the system firmware and the bootloaders:

  ```
  sudo bootctl status
  ```

- Set a flag to boot into the system firmware on the next boot (similar to `sudo systemctl reboot --firmware-setup`):

  ```
  sudo bootctl reboot-to-firmware true
  ```

- Specify the path to the EFI system partition (defaults `/efi/`, `/boot/` or `/boot/efi`):

  ```
  sudo bootctl --esp-path={{/path/to/efi_system_partition/}}
  ```

- Show all available bootloader entries:

  ```
  sudo bootctl list
  ```

- Install `systemd-boot` into the EFI system partition:

  ```
  sudo bootctl install
  ```

- Remove all installed versions of `systemd-boot` from the EFI system partition:

  ```
  sudo bootctl remove
  ```

borg

Deduplicating backup tool.

Creates local or remote backups that are mountable as filesystems.

More information: https://borgbackup.readthedocs.io/en/stable/usage/general.html.

- Initialise a (local) repository:

```
b org  init {{path/to/repo_directory}}
```

- B ackup a directory into the repository, creating an archive called " Monday" :

```
b org  create - - prog ress {{path/to/repo_directory}}: : {{M onday}}
{{path/to/sou rce_directory}}
```

- List all archives in a repository:

```
b org  list {{path/to/repo_directory}}
```

- E xtract a specific directory from the " Monday" archive in a remote repository, excluding all `* . ex` files:

```
b org  extract {{u ser}}@ {{host}}: {{path/to/repo_directory}}: :
{{M onday}} {{path/to/targ et_directory}} - - exclu de ' {{* . ext}}'
```

- P rune a repository by deleting all archives older than 7 days, listing changes:

```
b org  pru ne - - k eep- w ithin {{7 d}} - - list {{path/to/
repo_directory}}
```

- Mount a repository as a F U S E filesystem:

```
b org  m ou nt {{path/to/repo_directory}}: : {{M onday}} {{path/to/
m ou ntpoint}}
```

- Display help on creating archives:

```
b org  create - - help
```

bosh

Command-line tool to deploy and manage the bosh director.

More information: https://bosh.io/docs/cli-v2 /

- Create local alias for director:

  ```
  b osh alias- env  {{env ironm ent_nam e}} - e {{ip_address| u rl}} - -
  ca- cert {{ca_certificate}}
  ```

- List environments:

  ```
  b osh env ironm ents
  ```

- Log in to the director:

  ```
  b osh log in - e {{env ironm ent}}
  ```

- List deployments:

  ```
  b osh - e {{env ironm ent}} deploym ents
  ```

- List environment virtual machines:

  ```
  b osh - e {{env ironm ent}} v m s - d {{deploym ent}}
  ```

- S sh into virtual machine:

  ```
  b osh - e {{env ironm ent}} ssh {{v irtu al_m achine}} - d
  {{deploym ent}}
  ```

- U pload stemcell:

  ```
  b osh - e {{env ironm ent}} u pload- stem cell {{stem cell_file| u rl}}
  ```

- S how current cloud config:

  ```
  b osh - e {{env ironm ent}} clou d- config
  ```

bower

A package manager optimized for front-end web development.

A package can be a GitHub user/repo shorthand, a Git endpoint, a URL or a registered package.

More information: https://bower.io/.

- Install a project's dependencies, listed in its bower.json:

  `bower install`

- Install one or more packages to the bower_components directory:

  `bower install {{package}} {{package}}`

- Uninstall packages locally from the bower_components directory:

  `bower uninstall {{package}} {{package}}`

- List local packages and possible updates:

  `bower list`

- Display help information about a bower command:

  `bower help {{command}}`

- Create a `bower.json` file for your package:

  `bower init`

- Install a specific dependency version, and add it to `bower.json`:

  `bower install {{local_name}}={{package}}#{{version}} --save`

box

A P H P application for building and managing P hars.

More information: https://github.com/box-proj ect/box

- Compile a new P har file:

  ```
  b ox com pile
  ```

- Compile a new P har file using a specific config file:

  ```
  b ox com pile - c {{path/to/config }}
  ```

- Display information about the P H AR P H P extension:

  ```
  b ox info
  ```

- Display information about a specific P har file:

  ```
  b ox info {{path/to/phar_file}}
  ```

- V alidate the first found config file in the working directory:

  ```
  b ox v alidate
  ```

- V erify the signature of a specific P har file:

  ```
  b ox v erify {{path/to/phar_file}}
  ```

- Display all available commands and options:

  ```
  b ox help
  ```

brew

Package manager for macOS and Linux.

More information: https://brew.sh.

- Install the latest stable version of a formula or cask (use `--dev` for development versions):

  ```
  brew install {{formula}}
  ```

- List all installed formulae and casks:

  ```
  brew list
  ```

- Upgrade an installed formula or cask (if none is given, all installed formulae/casks are upgraded):

  ```
  brew upgrade {{formula}}
  ```

- Fetch the newest version of Homebrew and of all formulae and casks from the Homebrew source repository:

  ```
  brew update
  ```

- Show formulae and casks that have a more recent version available:

  ```
  brew outdated
  ```

- Search for available formulae (i.e. packages) and casks (i.e. native packages):

  ```
  brew search {{text}}
  ```

- Display information about a formula or a cask (version, installation path, dependencies, etc.):

  ```
  brew info {{formula}}
  ```

- Check the local Homebrew installation for potential problems:

  ```
  brew doctor
  ```

B rotli

Compress/uncompress files with brotli compression.

More information: https://github.com/google/brotli.

- Compress a file, creating a compressed version next to the file:

```
b rotli {{file. ext}}
```

- Decompress a file, creating an uncompressed version next to the file:

```
b rotli - d {{file. ext}}. b r
```

- Compress a file specifying the output filename:

```
b rotli {{file. ext}} - o {{com pressed_file. ext. b r}}
```

- Decompress a brotli file specifying the output filename:

```
b rotli - d {{com pressed_file. ext. b r}} - o {{file. ext}}
```

- S pecify the compression level. 1 = F astest (Worst), 1 1 = S lowest (B est):

```
b rotli - q  {{1 1 }} {{file. ext}} - o {{com pressed_file. ext. b r}}
```

browser-sync

Starts local web server that updates browser on file changes.

More information: https://browsersync.io/docs/command-line.

- Start a server from a specific directory:

```
browser-sync start --server {{path/to/directory}} --files {{path/to/directory}}
```

- Start a server from local directory, watching all css files in some directory:

```
browser-sync start --server --files '{{path/to/directory/*.css}}'
```

- Create configuration file:

```
browser-sync init
```

- Start browser-sync from config file:

```
browser-sync start --config {{config_file}}
```

bshell

A G U I tool for browsing for S S H /V N C servers on the local network.

S ee also: **`bs s h`** and **`bvn c`**.

More information: https://linux.extremeoverclocking.com/man/1 /bssh.

- B rowse for both S S H and V N C servers:

  ```
  b shell
  ```

- B rowse for S S H servers only:

  ```
  b shell - - ssh
  ```

- B rowse for V N C servers only:

  ```
  b shell - - v nc
  ```

- B rowse for both S S H and V N C servers in a specified domain:

  ```
  b shell - - dom ain= {{dom ain}}
  ```

bssh

A G U I tool for browsing for S S H /V N C servers on the local network.

S ee also **bvn c** and **bs h ell**.

More information: https://linux.extremeoverclocking.com/man/1 /bssh

- B rowse for S S H servers:

  ```
  b ssh
  ```

- B rowse for V N C servers:

  ```
  b ssh - - v nc
  ```

- B rowse for both S S H and V N C servers:

  ```
  b ssh - - shell
  ```

- B rowse for S S H servers in a specified domain:

  ```
  b ssh - - dom ain= {{dom ain}}
  ```

btm

An alternative to **`top`**

Aims to be lightweight, cross-platform and more graphical than **`top`**

More information: https://github.com/ClementTsang/bottom

- Show the default layout (cpu, memory, temperatures, disk, network, and processes):

  ```
  btm
  ```

- Enable basic mode, removing charts and condensing data (similar to `top`):

  ```
  btm --basic
  ```

- Use big dots instead of small ones in charts:

  ```
  btm --dot_marker
  ```

- Show also battery charge and health status:

  ```
  btm --battery
  ```

- Refresh every 250 milliseconds and show the last 30 seconds in the charts:

  ```
  btm --rate 250 --default_time_value 30000
  ```

buku

Command-line browser-independent bookmark manager.

More information: https://github.com/j arun/B uku

- Display all bookmarks matching " keyword" and with " privacy" tag:

```
b u k u {{k eyw ord}} - - stag {{priv acy}}
```

- Add bookmark with tags " search engine" and " privacy" :

```
b u k u - - add {{https: //exam ple. com }} {{search eng ine}},
{{priv acy}}
```

- Delete a bookmark:

```
b u k u - - delete {{b ook m ark _id}}
```

- O pen editor to edit a bookmark:

```
b u k u - - w rite {{b ook m ark _id}}
```

- R emove " search engine" tag from a bookmark:

```
b u k u - - u pdate {{b ook m ark _id}} - - tag {{- }} {{search eng ine}}
```

bundle

Dependency manager for the Ruby programming language.

More information: https://bundler.io/man/bundle.1.html

- Install all gems defined in the `Gemfile` expected in the working directory:

  ```
  bundle install
  ```

- Execute a command in the context of the current bundle:

  ```
  bundle exec {{command}} {{arguments}}
  ```

- Update all gems by the rules defined in the `Gemfile` and regenerate `Gemfile.lock`:

  ```
  bundle update
  ```

- Update one or more specific gem(s) defined in the `Gemfile`:

  ```
  bundle update {{gem_name}} {{gem_name}}
  ```

- Update one or more specific gems(s) defined in the `Gemfile` but only to the next patch version:

  ```
  bundle update --patch {{gem_name}} {{gem_name}}
  ```

- Update all gems within the given group in the `Gemfile`:

  ```
  bundle update --group {{development}}
  ```

- List installed gems in the `Gemfile` with newer versions available:

  ```
  bundle outdated
  ```

- Create a new gem skeleton:

  ```
  bundle gem {{gem_name}}
  ```

bundletool dump

Command-line tool to manipulate Android Application Bundles.

More information: https://developer.android.com/studio/command-line/bundletool.

- Display the `AndroidManifest.xml` of the base module:

  ```
  bundletool dump manifest --bundle={{path/to/bundle.aab}}
  ```

- Display a specific value from the `AndroidManifest.xml` using XPath:

  ```
  bundletool dump manifest --bundle={{path/to/bundle.aab}} --
  xpath={{/manifest/@android:versionCode}}
  ```

- Display the `AndroidManifest.xml` of a specific module:

  ```
  bundletool dump manifest --bundle={{path/to/bundle.aab}} --
  module={{name}}
  ```

- Display all the resources in the application bundle:

  ```
  bundletool dump resources --bundle={{path/to/bundle.aab}}
  ```

- Display the configuration for a specific resource:

  ```
  bundletool dump resources --bundle={{path/to/bundle.aab}} --
  resource={{type/name}}
  ```

- Display the configuration and values for a specific resource using the ID:

  ```
  bundletool dump resources --bundle={{path/to/bundle.aab}} --
  resource={{0x7f0e013a}} --values
  ```

- Display the contents of the bundle configuration file:

  ```
  bundletool dump config --bundle={{path/to/bundle.aab}}
  ```

bundletool validate

Command-line tool to manipulate Android Application B undles.

More information: https://developer.android.com/studio/command-line/bundletool.

- V erify a bundle and display detailed information about it:

```
b u ndletool v alidate - - b u ndle= {{path/to/b u ndle. aab }}
```

bundletool

Command-line tool to manipulate Android Application Bundles.

More information: https://developer.android.com/studio/command-line/bundletool.

- Display help for a subcommand:

```
bundletool help {{subcommand}}
```

- Generate APKs from an application bundle (prompts for keystore password):

```
bundletool build-apks --bundle={{path/to/bundle.aab}} --
ks={{path/to/key.keystore}} --ks-key-alias={{key_alias}} --
output={{path/to/file.apks}}
```

- Generate APKs from an application bundle giving the keystore password:

```
bundletool build-apks --bundle={{path/to/bundle.aab}} --
ks={{path/to/key.keystore}} --ks-key-alias={{key_alias}} —ks
pass={{pass:the_password}} --output={{path/to/file.apks}}
```

- Generate APKs including only one single APK for universal usage:

```
bundletool build-apks --bundle={{path/to/bundle.aab}} --
mode={{universal}} --ks={{path/to/key.keystore}} --ks-key-
alias={{key_alias}} --output={{path/to/file.apks}}
```

- Install the right combination of APKs to an emulator or device:

```
bundletool install-apks --apks={{path/to/file.apks}}
```

- Estimate the download size of an application:

```
bundletool get-size total --apks={{path/to/file.apks}}
```

- Generate a device specification JSON file for an emulator or device:

```
bundletool get-device-spec --output={{path/to/file.json}}
```

- Verify a bundle and display detailed information about it:

```
bundletool validate --bundle={{path/to/bundle.aab}}
```

bup

Backup system based on the Git packfile format, providing incremental saves and global deduplication.

More information: https://github.com/bup/bup.

- Initialize a backup repository in the specified local directory:

```
bup -d {{path/to/repository}} init
```

- Prepare a given directory before taking a backup:

```
bup -d {{path/to/repository}} index {{path/to/directory}}
```

- Backup a directory to the repository:

```
bup -d {{path/to/repository}} save -n {{backup_name}} {{path/to/directory}}
```

- Show the backup snapshots currently stored in the repository:

```
bup -d {{path/to/repository}} ls
```

- Restore a specific backup snapshot to a target directory:

```
bup -d {{path/to/repository}} restore -C {{path/to/target_directory}} {{backup_name}}
```

buz z phrase

N ode.j s command-line tool to output a random buz z phrase.

More information: [https://github.com/atomantic/buz z phrase](https://github.com/atomantic/buzzphrase)

- G enerate a string of three random phrases containing an adj ective, a past tense verb and a plural noun:

  ```
  b u zzphrase
  ```

- O utput a phrase formatted as [i] mperative verb + past tense [v] erb + [a] dj ective + plural [N] oun:

  ```
  b u zzphrase {{' {i} {v } {a} {N }' }}
  ```

- O utput 4 phrases formatted as present participle [V] erb + [a] dj ective + singular [n] oun + [f] inal:

  ```
  b u zzphrase {{4 ' {V } {a} {n} {f}' }}
  ```

bvnc

A G U I tool for browsing for S S H /V N C servers on the local network.

S ee also: **bs s h** and **bs h ell**.

More information: https://linux.extremeoverclocking.com/man/1 /bssh.

- B rowse for V N C servers:

  ```
  b v nc
  ```

- B rowse for S S H servers:

  ```
  b v nc - - ssh
  ```

- B rowse for both V N C and S S H servers:

  ```
  b v nc - - shell
  ```

- B rowse for V N C servers in a specified domain:

  ```
  b v nc - - dom ain= {{dom ain}}
  ```

bw

A CLI to access and manage a Bitwarden vault.

More information: https://help.bitwarden.com/article/cli/.

- Log in to a Bitwarden user account:

  ```
  bw login
  ```

- Log out of a Bitwarden user account:

  ```
  bw logout
  ```

- Search and display items from Bitwarden vault:

  ```
  bw list items --search {{github}}
  ```

- Display a particular item from Bitwarden vault:

  ```
  bw get item {{github}}
  ```

- Create a folder in Bitwarden vault:

  ```
  {{echo -n '{"name":"My Folder1"}' | base64}} | bw create
  folder
  ```

bz ip2

A block-sorting file compressor.

More information: http://bz ip.org

- Compress a file:

  ```
  b zip2 {{path/to/file_to_com press}}
  ```

- Decompress a file:

  ```
  b zip2 - d {{path/to/com pressed_file. b z2 }}
  ```

- Decompress a file to standard output:

  ```
  b zip2 - dc {{path/to/com pressed_file. b z2 }}
  ```

c9 9

Compiles C programs according to the IS O C standard.

More information: https://manned.org/c9 9

- Compile source file(s) and create an executable:

```
c9 9 {{file. c}}
```

- Compile source file(s) and create an executable with a custom name:

```
c9 9 - o {{execu tab le_nam e}} {{file. c}}
```

- Compile source file(s) and create obj ect file(s):

```
c9 9 - c {{file. c}}
```

- Compile source file(s), link with obj ect file(s), and create an executable:

```
c9 9 {{file. c}} {{file. o}}
```

cabal

Command-line interface to the H askell package infrastructure (Cabal).

Manage H askell proj ects and Cabal packages from the H ackage package repository.

More information: https://cabal.readthedocs.io/en/latest/intro.html.

- S earch and list packages from H ackage:

  ```
  cab al list {{search_string }}
  ```

- S how information about a package:

  ```
  cab al info {{pack ag e_nam e}}
  ```

- Download and install a package:

  ```
  cab al install {{pack ag e_nam e}}
  ```

- Create a new H askell proj ect in the current directory:

  ```
  cab al init
  ```

- B uild the proj ect in the current directory:

  ```
  cab al b u ild
  ```

- R un tests of the proj ect in the current directory:

  ```
  cab al test
  ```

caddy

A powerful, enterprise-ready, open source web server with automatic HTTPS, written in Go.

More information: https://caddyserver.com.

- Start Caddy in the foreground:

  ```
  caddy run
  ```

- Start Caddy with the specified Caddyfile:

  ```
  caddy run --config {{path/to/Caddyfile}}
  ```

- Start Caddy in the background:

  ```
  caddy start
  ```

- Stop a background Caddy process:

  ```
  caddy stop
  ```

- Run a simple file server on the specified port with a browsable interface:

  ```
  caddy file-server --listen :{{8000}} --browse
  ```

- Run a reverse proxy server:

  ```
  caddy reverse-proxy --from :{{80}} --to localhost:{{8000}}
  ```

cake

The command-line processor for the CakePHP framework.

More information: https://cakephp.org.

- Display basic information about the current app and available commands:

  ```
  cake
  ```

- Display a list of available routes:

  ```
  cake routes
  ```

- Clear configuration caches:

  ```
  cake cache clear_all
  ```

- Build the metadata cache:

  ```
  cake schema_cache build --connection {{connection}}
  ```

- Clear the metadata cache:

  ```
  cake schema_cache clear
  ```

- Clear a single cache table:

  ```
  cake schema_cache clear {{table_name}}
  ```

- Start a development web server (defaults to port 8765):

  ```
  cake server
  ```

- Start a REPL interactive shell instance:

  ```
  cake console
  ```

calendar

Display upcoming events from a calendar file.

More information: https://www.commandlinux.com/man-page/man1/calendar.1.html

- Show events for today and tomorrow (or the weekend on Friday) from the default calendar:

  ```
  calendar
  ```

- Look [A]head, showing events for the next 30 days:

  ```
  calendar -A {{30}}
  ```

- Look [B]ack, showing events for the previous 7 days:

  ```
  calendar -B {{7}}
  ```

- Show events from a custom calendar [f]ile:

  ```
  calendar -f {{path/to/file}}
  ```

calibre-server

A server application that can be used to distribute e-books over a network.

Note: e-books must already be imported into the library using the GUI or the `calibredb` CLI.

More information: https://manual.calibre-ebook.com/generated/en/calibre-server.html.

- Start a server to distribute e-books. Access at http://localhost:8080:

  ```
  calibre-server
  ```

- Start server on different port. Access at http://localhost:port:

  ```
  calibre-server --port {{port}}
  ```

- Password protect the server:

  ```
  calibre-server --username {{username}} --password {{password}}
  ```

calibredb

Tool to manipulate the your e-book database.

Part of the Calibre e-book library.

More information: https://manual.calibre-ebook.com/generated/en/calibredb.html.

- List e-books in the library with additional information:

  ```
  calibredb list
  ```

- Search for e-books displaying additional information:

  ```
  calibredb list --search {{search_term}}
  ```

- Search for just ids of e-books:

  ```
  calibredb search {{search_term}}
  ```

- Add one or more e-books to the library:

  ```
  calibredb add {{file1 file2 …}}
  ```

- Recursively add all e-books under a directory to the library:

  ```
  calibredb add -r {{path/to/directory}}
  ```

- Remove one or more e-books from the library. You need the e-book IDs (see above):

  ```
  calibredb remove {{id1 id2 …}}
  ```

carbon-now

Create beautiful images of code.

More information: https://github.com/mixn/carbon-now-cli.

- Create an image from a file using default settings:

  ```
  carbon-now {{file}}
  ```

- Create an image from a text in clipboard using default settings:

  ```
  carbon-now --from-clipboard
  ```

- Create an image from standard input using default settings:

  ```
  {{input}} | carbon-now
  ```

- Create images interactively for custom settings and optionally save a preset:

  ```
  carbon-now -i {{file}}
  ```

- Create images from previously saved preset:

  ```
  carbon-now -p {{preset}} {{file}}
  ```

- Start at a specified line of text:

  ```
  carbon-now -s {{line}} {{file}}
  ```

- End at a specific line of text:

  ```
  carbon-now -e {{line}} {{file}}
  ```

- Open image in a browser instead of saving:

  ```
  carbon-now --open {{file}}
  ```

cargo build

Compile a local package and all of its dependencies.

More information: https://doc.rust-lang.org/cargo/commands/cargo-build.html.

- Build the package or packages defined by the `Cargo.toml` manifest file in the local path:

  ```
  cargo build
  ```

- Build artifacts in release mode, with optimizations:

  ```
  cargo build --release
  ```

- Require that `Cargo.lock` is up to date:

  ```
  cargo build --locked
  ```

- Build all packages in the workspace:

  ```
  cargo build --workspace
  ```

- Build a specific package:

  ```
  cargo build --package {{package}}
  ```

- Build only the specified binary:

  ```
  cargo build --bin {{name}}
  ```

- Build only the specified test target:

  ```
  cargo build --test {{testname}}
  ```

cargo clippy

A collection of lints to catch common mistakes and improve your Rust code.

More information: https://github.com/rust-lang/rust-clippy.

- Run checks over the code in the current directory:

  ```
  cargo clippy
  ```

- Require that `Cargo.lock` is up to date:

  ```
  cargo clippy --locked
  ```

- Run checks on all packages in the workspace:

  ```
  cargo clippy --workspace
  ```

- Run checks for a package:

  ```
  cargo clippy --package {{package}}
  ```

- Treat warnings as errors:

  ```
  RUSTFLAGS="-Dwarnings" cargo clippy --  -D warnings
  ```

- Run checks and ignore warnings:

  ```
  cargo clippy --  -A warnings
  ```

- Apply Clippy suggestion automatically (experimental and only supported on the nightly channel):

  ```
  cargo clippy --fix -Z unstable-options
  ```

cargo doc

Build and view Rust package documentation offline.

More information: https://doc.rust-lang.org/cargo/commands/cargo-doc.html.

- Build and view the default package documentation in the browser:

  ```
  cargo doc --open
  ```

- Build documentation without accessing the network:

  ```
  cargo doc --offline
  ```

- View a particular package's documentation:

  ```
  cargo doc --open --package {{package}}
  ```

- View a particular package's documentation offline:

  ```
  cargo doc --open --offline --package {{package}}
  ```

cargo rustc

Compile a Rust package, and pass extra options to the compiler.

More information: https://doc.rust-lang.org/cargo/commands/cargo-rustc.html.

- Build the package or packages defined by the `Cargo.toml` manifest file in the current working directory:

  ```
  cargo rustc
  ```

- Build artifacts in release mode, with optimizations:

  ```
  cargo rustc --release
  ```

- Compile with architecture-specific optimizations for the current CPU:

  ```
  cargo rustc --release -- -C target-cpu=native
  ```

- Compile with speed optimization:

  ```
  cargo rustc -- -C opt-level={{1|2|3}}
  ```

- Compile with [s]ize optimizations (`z` also turns off loop vectorization):

  ```
  cargo rustc -- -C opt-level={{s|z}}
  ```

- Check if your package uses unsafe code:

  ```
  cargo rustc --lib -- -D unsafe-code
  ```

- Build a specific package:

  ```
  cargo rustc --package {{package}}
  ```

- Build only the specified binary:

  ```
  cargo rustc --bin {{name}}
  ```

cargo test

Execute the unit and integration tests of a Rust package.

More information: https://doc.rust-lang.org/cargo/commands/cargo-test.html.

- Only run tests containing a specific string in their names:

  ```
  cargo test {{testname}}
  ```

- Set the number of simultaneous running test cases:

  ```
  cargo test --  --test-threads={{count}}
  ```

- Require that `Cargo.lock` is up to date:

  ```
  cargo test --locked
  ```

- Test artifacts in release mode, with optimizations:

  ```
  cargo test --release
  ```

- Test all packages in the workspace:

  ```
  cargo test --workspace
  ```

- Run tests for a package:

  ```
  cargo test --package {{package}}
  ```

cargo

Rust package manager.

Manage Rust projects and their module dependencies (crates).

More information: https://crates.io/.

- Search for crates:

  ```
  cargo search {{search_string}}
  ```

- Install a crate:

  ```
  cargo install {{crate_name}}
  ```

- List installed crates:

  ```
  cargo install --list
  ```

- Create a new binary or library Rust project in the current directory:

  ```
  cargo init --{{bin|lib}}
  ```

- Create a new binary or library Rust project in the specified directory:

  ```
  cargo new {{path/to/directory}} --{{bin|lib}}
  ```

- Build the Rust project in the current directory:

  ```
  cargo build
  ```

- Build using a specific number of threads (default is the number of CPU cores):

  ```
  cargo build -j {{jobs}}
  ```

case

Branch based on the value of an expression.

More information: https://manned.org/case.

- Match a variable against string literals to decide which command to run:

```
case {{$ tocou nt}} in {{w ords}}) {{w c -w R E A D M E }}; ;
{{lines}}) {{w c -l R E A D M E }}; ; esac
```

- Combine patterns with |, use * as a fallback pattern:

```
case {{$ tocou nt}} in {{[ w W ] | w ords}}) {{w c -w R E A D M E }}; ;
{{[ lL ] | lines}}) {{w c -l R E A D M E }}; ; * ) {{echo " w hat?" }}
esac
```

cat

Print and concatenate files.

More information: https://www.gnu.org/software/coreutils/cat.

- Print the contents of a file to the standard output:

  `cat {{file}}`

- Concatenate several files into the target file:

  `cat {{file1}} {{file2}} > {{target_file}}`

- Append several files into the target file:

  `cat {{file1}} {{file2}} >> {{target_file}}`

- Number all output lines:

  `cat -n {{file}}`

- Display non-printable and whitespace characters (with `M-`prefix if non-ASCII):

  `cat -v -t -e {{file}}`

cd

Change the current working directory.

More information: https://man.archlinux.org/man/cd.n.

- G o to the given directory:

  ```
  cd {{path/to/directory}}
  ```

- G o to home directory of current user:

  ```
  cd
  ```

- G o up to the parent of the current directory:

  ```
  cd . .
  ```

- G o to the previously chosen directory:

  ```
  cd -
  ```

cdk

A CLI for AWS Cloud Development Kit (CDK).

More information: https://docs.aws.amazon.com/cdk/latest/guide/cli.html

- List the stacks in the app:

  ```
  cdk ls
  ```

- Synthesize and print the CloudFormation template for the specified stack(s):

  ```
  cdk synth {{stack_name}}
  ```

- Deploy a space-separated list of stacks:

  ```
  cdk deploy {{stack_name}}
  ```

- Destroy a space-separated list of stacks:

  ```
  cdk destroy {{stack_name}}
  ```

- Compare the specified stack with the deployed stack or a local CloudFormation template:

  ```
  cdk diff {{stack_name}}
  ```

- Create a new CDK project in the current directory for a specified language:

  ```
  cdk init -l {{language_name}}
  ```

- Open the CDK API reference in your browser:

  ```
  cdk doc
  ```

cf

Command-line tool to manage apps and services on Cloud Foundry.

More information: https://docs.cloudfoundry.org.

- Push an app using the default settings:

  ```
  cf push {{app_name}}
  ```

- View the services available from your organization:

  ```
  cf marketplace
  ```

- Create a service instance:

  ```
  cf create-service {{service}} {{plan}} {{service_name}}
  ```

- Connect an application to a service:

  ```
  cf bind-service {{app_name}} {{service_name}}
  ```

- Run a script whose code is included in the app, but runs independently:

  ```
  cf run-task {{app_name}} "{{script_command}}" --name {{task_name}}
  ```

- Start an interactive SSH session with a VM hosting an app:

  ```
  cf ssh {{app_name}}
  ```

- View a dump of recent app logs:

  ```
  cf logs {{app_name}} --recent
  ```

chars

Display names and codes for various ASCII and Unicode characters and code points.

More information: https://github.com/antifuchs/chars.

- Look up a character by its value:

  ```
  chars ' {{ß }}'
  ```

- Look up a character by its Unicode code point:

  ```
  chars {{U + 1 F 6 3 C }}
  ```

- Look up possible characters given an ambiguous code point:

  ```
  chars {{1 0 }}
  ```

- Look up a control character:

  ```
  chars " {{^ C }}"
  ```

chcon

Change S E Linux security context of a file or files/directories.

More information: https://www.gnu.org/software/coreutils/chcon.

- V iew security context of a file:

```
ls - lZ  {{path/to/file}}
```

- Change the security context of a target file, using a reference file:

```
chcon - - reference= {{reference_file}} {{targ et_file}}
```

- Change the full S E Linux security context of a file:

```
chcon {{u ser}}: {{role}}: {{type}}: {{rang e/lev el}} {{filenam e}}
```

- Change only the user part of S E Linux security context:

```
chcon - u  {{u ser}} {{filenam e}}
```

- Change only the role part of S E Linux security context:

```
chcon - r {{role}} {{filenam e}}
```

- Change only the type part of S E Linux security context:

```
chcon - t {{type}} {{filenam e}}
```

- Change only the range/level part of S E Linux security context:

```
chcon - l {{rang e/lev el}} {{filenam e}}
```

cheat

Create and view interactive cheat sheets on the command-line.

More information: https://github.com/cheat/cheat.

- S how example usage of a command:

  ```
  cheat {{com m and}}
  ```

- E dit the cheat sheet for a command:

  ```
  cheat - e {{com m and}}
  ```

- List the available cheat sheets:

  ```
  cheat - l
  ```

- S earch available the cheat sheets for a specified command name:

  ```
  cheat - s {{com m and}}
  ```

- G et the current cheat version:

  ```
  cheat - v
  ```

Chezmoi

A multi-machine dotfile manager, written in Go.

More information: https://chezmoi.io

- Initialize chezmoi on your machine:

  ```
  chezmoi init
  ```

- Tell chezmoi to manage a dotfile:

  ```
  chezmoi add {{path/to/file}}
  ```

- Edit the source state of a tracked dotfile:

  ```
  chezmoi edit {{path/to/file}}
  ```

- See changes chezmoi would make:

  ```
  chezmoi diff
  ```

- Apply the changes:

  ```
  chezmoi -v apply
  ```

- Set chezmoi up on another machine by downloading existing dotfiles from a Git repository:

  ```
  chezmoi init {{https://example.com/path/to/repository.git}}
  ```

- Fetch the latest changes from a remote repository:

  ```
  chezmoi update
  ```

chgrp

Change group ownership of files and directories.

More information: https://www.gnu.org/software/coreutils/chgrp.

- Change the owner group of a file/directory:

```
chg rp {{g rou p}} {{path/to/file_or_directory}}
```

- R ecursively change the owner group of a directory and its contents:

```
chg rp - R  {{g rou p}} {{path/to/directory}}
```

- Change the owner group of a symbolic link:

```
chg rp - h {{g rou p}} {{path/to/sym link }}
```

- Change the owner group of a file/directory to match a reference file:

```
chg rp - - reference= {{path/to/reference_file}} {{path/to/
file_or_directory}}
```

chisel

Create TCP tunnels. Includes both client and server.

More information: https://github.com/jpillora/chisel

- Run a Chisel server:

  ```
  chisel server
  ```

- Run a Chisel server listening to a specific port:

  ```
  chisel server -p {{server_port}}
  ```

- Run a chisel server that accepts authenticated connections using username and password:

  ```
  chisel server --auth {{username}}:{{password}}
  ```

- Connect to a Chisel server and tunnel a specific port to a remote server and port:

  ```
  chisel client {{server_ip}}:{{server_port}} {{local_port}}:{{remote_server}}:{{remote_port}}
  ```

- Connect to a Chisel server and tunnel a specific host and port to a remote server and port:

  ```
  chisel client {{server_ip}}:{{server_port}} {{local_host}}:{{local_port}}:{{remote_server}}:{{remote_port}}
  ```

- Connect to a Chisel server using username and password authentication:

  ```
  chisel client --auth {{username}}:{{password}} {{server_ip}}:{{server_port}} {{local_port}}:{{remote_server}}:{{remote_port}}
  ```

chmod

Change the access permissions of a file or directory.

More information: https://www.gnu.org/software/coreutils/chmod.

- Give the [u]ser who owns a file the right to e[x]ecute it:

  ```
  chmod u+x {{file}}
  ```

- Give the [u]ser rights to [r]ead and [w]rite to a file/directory:

  ```
  chmod u+rw {{file_or_directory}}
  ```

- Remove e[x]ecutable rights from the [g]roup:

  ```
  chmod g-x {{file}}
  ```

- Give [a]ll users rights to [r]ead and e[x]ecute:

  ```
  chmod a+rx {{file}}
  ```

- Give [o]thers (not in the file owner's group) the same rights as the [g]roup:

  ```
  chmod o=g {{file}}
  ```

- Remove all rights from [o]thers:

  ```
  chmod o= {{file}}
  ```

- Change permissions recursively giving [g]roup and [o]thers the ability to [w]rite:

  ```
  chmod -R g+w,o+w {{directory}}
  ```

choose

A human-friendly and fast alternative to cut and (sometimes) awk.

More information: https://github.com/theryangeary/choose.

- Print the 5th item from a line (starting from 0):

  ```
  choose {{4}}
  ```

- Print the first, 3rd, and 5th item from a line, where items are separated by ':' instead of whitespace:

  ```
  choose --field-separator '{{:}}' {{0}} {{2}} {{4}}
  ```

- Print everything from the 2nd to 5th item on the line, including the 5th:

  ```
  choose {{1}}:{{4}}
  ```

- Print everything from the 2nd to 5th item on the line, excluding the 5th:

  ```
  choose --exclusive {{1}}:{{4}}
  ```

- Print the beginning of the line to the 3rd item:

  ```
  choose :{{2}}
  ```

- Print all items from the beginning of the line until the 3rd item (exclusive):

  ```
  choose --exclusive :{{2}}
  ```

- Print all items from the 3rd to the end of the line:

  ```
  choose {{2}}:
  ```

- Print the last item from a line:

  ```
  choose {{-1}}
  ```

chown

Change user and group ownership of files and directories.

More information: https://www.gnu.org/software/coreutils/chown.

- Change the owner user of a file/directory:

```
chown {{user}} {{path/to/file_or_directory}}
```

- Change the owner user and group of a file/directory:

```
chown {{user}}:{{group}} {{path/to/file_or_directory}}
```

- Recursively change the owner of a directory and its contents:

```
chown -R {{user}} {{path/to/directory}}
```

- Change the owner of a symbolic link:

```
chown -h {{user}} {{path/to/symlink}}
```

- Change the owner of a file/directory to match a reference file:

```
chown --reference={{path/to/reference_file}} {{path/to/file_or_directory}}
```

chroma

Chroma is a general-purpose syntax highlighting library and corresponding command, for Go.

More information: https://github.com/alecthomas/chroma.

- Highlight a source file with python lexer and output to terminal:

```
chroma --lexer="{{python}}" {{source_file}}
```

- Highlight a source file with Go lexer and output to a HTML file:

```
chroma --lexer="{{go}}" --formatter="{{html}}" {{source_file}} > {{html_file}}
```

- Highlight a source file with C++ lexer and output to an SVG image, using the Monokai style:

```
chroma --lexer="{{c++}}" --formatter="{{svg}}" --style="{{monokai}}" {{source_file}} > {{svg_file}}
```

chromium

Open-source web browser from Google.

More information: https://chromium.org.

- Open a file:

```
chromium {{path/to/file.html}}
```

- Open an URL:

```
chromium {{example.com}}
```

- Open in incognito mode:

```
chromium --incognito {{example.com}}
```

- Open in a new window:

```
chromium --new-window {{example.com}}
```

- Open in app mode (without toolbars, URL bar, buttons, etc.):

```
chromium --app='{{https://example.com}}'
```

- Use a proxy server:

```
chromium --proxy-server="{{socks5://hostname:66}}" {{example.com}}
```

chroot

Run command or interactive shell with special root directory.

More information: https://www.gnu.org/software/coreutils/chroot.

- Run command as new root directory:

  ```
  chroot {{path/to/new/root}} {{command}}
  ```

- Specify user and group (ID or name) to use:

  ```
  chroot --user=spec={{user:group}}
  ```

chsh

Change the user's login shell.

More information: https://manned.org/chsh.

- Change the user's login shell:

```
chsh -s {{path/to/shell_binary}} {{username}}
```

circo

Render an image of a **circular** network graph from a **graphviz** file.

Layouts: **dot**, **neato**, **twopi**, **circo**, **fdp**, **sfdp**, **osage** & **patchwork**

More information: https://graphviz.org/doc/info/command.html

- Render a `png` image with a filename based on the input filename and output format (uppercase -O):

  ```
  circo -Tpng -O {{path/to/input.gv}}
  ```

- Render a `svg` image with the specified output filename (lowercase -o):

  ```
  circo -Tsvg -o {{path/to/image.svg}} {{path/to/input.gv}}
  ```

- Render the output in `ps`, `pdf`, `svgz`, `fig`, `png`, `gif`, `jpg`, `jpeg`, or `dot` format:

  ```
  circo -Tformat -O {{path/to/input.gv}}
  ```

- Render a `gif` image using stdin and stdout:

  ```
  echo "{{digraph {this -> that}}}" | circo -Tgif > {{path/to/image.gif}}
  ```

- Display help:

  ```
  circo -?
  ```

circup

The CircuitPython library updater.

More information: https://github.com/adafruit/circup.

- Interactively update modules on a device:

  `circup update`

- Install a new library:

  `circup install {{library_name}}`

- Search for a library:

  `circup show {{partial_name}}`

- List all libraries on a connected device in `requirements.txt` format:

  `circup freeze`

- Save all libraries on a connected device in the current directory:

  `circup freeze -r`

cksum

Calculates CRC checksums and byte counts of a file.

Note, on old UNIX systems the CRC implementation may differ.

More information: https://www.gnu.org/software/coreutils/cksum.

- Display a 32 bit checksum, size in bytes and filename:

```
cksum {{filename}}
```

clamscan

A command-line virus scanner.

More information: https://www.clamav.net.

- S can a file for vulnerabilities:

  ```
  clam scan {{path/to/file}}
  ```

- S can all files recursively in a specific directory:

  ```
  clam scan -r {{path/to/directory}}
  ```

- S can data from stdin:

  ```
  {{command}} | clam scan -
  ```

- S pecify a virus database file or directory of files:

  ```
  clam scan --database {{path/to/database_file_or_directory}}
  ```

- S can the current directory and output only infected files:

  ```
  clam scan --infected
  ```

- O utput the scan report to a log file:

  ```
  clam scan --log {{path/to/log_file}}
  ```

- Move infected files to a specific directory:

  ```
  clam scan --move {{path/to/quarantine_directory}}
  ```

- R emove infected files:

  ```
  clam scan --remove yes
  ```

clang-format

Tool to auto-format C/C++/Java/JavaScript/Objective-C/Protobuf/C# code.

More information: https://clang.llvm.org/docs/ClangFormat.html.

- Format a file and print the result to stdout:

  ```
  clang-format {{path/to/file}}
  ```

- Format a file in-place:

  ```
  clang-format -i {{path/to/file}}
  ```

- Format a file using a predefined coding style:

  ```
  clang-format --style={{LLVM|Google|Chromium|Mozilla|Webkit}} {{path/to/file}}
  ```

- Format a file using the `.clang-format` file in one of the parent directories of the source file:

  ```
  clang-format --style=file {{path/to/file}}
  ```

- Generate a custom `.clang-format` file:

  ```
  clang-format --style={{LLVM|Google|Chromium|Mozilla|Webkit}} --dump-config > {{.clang-format}}
  ```

clang

Compiler for C, C++, and Objective-C source files. Can be used as a drop-in replacement for GCC.

More information: https://clang.llvm.org/docs/ClangCommandLineReference.html

- Compile a source code file into an executable binary:

```
clang {{input_source.c}} -o {{output_executable}}
```

- Activate output of all errors and warnings:

```
clang {{input_source.c}} -Wall -o {{output_executable}}
```

- Include libraries located at a different path than the source file:

```
clang {{input_source.c}} -o {{output_executable}} -
I {{header_path}} -L {{library_path}} -l{{library_name}}
```

- Compile source code into LLVM Intermediate Representation (IR):

```
clang -S -emit-llvm {{file.c}} -o {{file.ll}}
```

clear

Clears the screen of the terminal.

More information: https://manned.org/clear.

- Clear the screen (equivalent to pressing Control-L in Bash shell):

```
clear
```

- Clear the screen but keep the terminal's scrollback buffer:

```
clear -x
```

- Indicate the type of terminal to clean (defaults to the value of the environment variable `TERM`):

```
clear -T {{type_of_terminal}}
```

- Show the version of `ncurses` used by `clear`:

```
clear -V
```

clementine

A modern music player and library organizer.

More information: https://www.clementine-player.org.

- Open Clementine:

  ```
  clementine
  ```

- Start playing a music file:

  ```
  clementine {{url/or/path/to/file.ext}}
  ```

- Toggle between pausing and playing:

  ```
  clementine --play-pause
  ```

- Stop playback:

  ```
  clementine --stop
  ```

- Skip to the next track:

  ```
  clementine --next
  ```

- Skip to the previous track:

  ```
  clementine --previous
  ```

- Load a playlist file:

  ```
  clementine --load {{path/to/playlist.ext}}
  ```

- Play the 5th track in the currently loaded playlist:

  ```
  clementine --play-track {{5}}
  ```

clj

Clojure tool to start a REPL or invoke a specific function with data.

All options can be defined in a `deps.edn` file.

More information: https://clojure.org/guides/deps_and_cli

- Start a REPL:

  ```
  clj
  ```

- Execute a function:

  ```
  clj -X {{namespace/function_name}}
  ```

- Run the main function of a specified namespace:

  ```
  clj -M -m {{namespace}} {{args}}
  ```

- Prepare a project by resolving dependencies, downloading libraries, and making / caching classpaths:

  ```
  clj -P
  ```

- Start an nREPL server with the CIDER middleware:

  ```
  clj -Sdeps '{:deps {nrepl {:mvn/version "0.7.0"} cider/cider
  nrepl {:mvn/version "0.25.2"}}}' -m nrepl.cmdline --
  middleware '["cider.nrepl/cider-middleware"]' --interactive
  ```

- Start a REPL for ClojureScript and open a web browser:

  ```
  clj -Sdeps '{:deps {org.clojure/clojurescript {:mvn/version
  "1.10.758"}}}' --main cljs.main --repl
  ```

cloc

Count, and compute differences of, lines of source code and comments.

More information: https://github.com/AlDanial/cloc.

- Count all the lines of code in a directory:

  ```
  cloc {{path/to/directory}}
  ```

- Count all the lines of code in a directory, displaying a progress bar during the counting process:

  ```
  cloc --progress=1 {{path/to/directory}}
  ```

- Compare 2 directory structures and count the differences between them:

  ```
  cloc --diff {{path/to/directory/one}} {{path/to/directory/two}}
  ```

- Ignore files that are ignored by VCS, such as files specified in `.gitignore`:

  ```
  cloc --vcs=git {{path/to/directory}}
  ```

- Count all the lines of code in a directory, displaying the results for each file instead of each language:

  ```
  cloc --by-file {{path/to/directory}}
  ```

clockwork-cli

A command-line interface for the Clockwork PHP debugging framework.

More information: https://github.com/ptrofimov/clockwork-cli.

- Monitor Clockwork logs for the current project:

  ```
  clockwork-cli
  ```

- Monitor Clockwork logs for a specific project:

  ```
  clockwork-cli {{path/to/directory}}
  ```

- Monitor Clockwork logs for multiple projects:

  ```
  clockwork-cli {{path/to/directory1 path/to/directory2 …}}
  ```

cloudflared

Command-line tool to create a persistent connection to the Cloudflare network.

More information: https://developers.cloudflare.com/argo-tunnel/

- Authenticate and associate the connection to a domain in the Cloudflare account:

  ```
  cloudflared tunnel login
  ```

- Establish a tunnel to a host in Cloudflare from the local server:

  ```
  cloudflared tunnel --hostname {{hostname}} localhost:{{port_number}}
  ```

- Establish a tunnel to a host in Cloudflare from the local server, without verifying the local server's certificate:

  ```
  cloudflared tunnel --hostname {{hostname}} localhost:{{port_number}} --no-tls-verify
  ```

- Save logs to a file:

  ```
  cloudflared tunnel --hostname {{hostname}} http://localhost:{{port_number}} --loglevel {{panic|fatal|error|warn|info|debug}} --logfile {{path/to/file}}
  ```

- Install cloudflared as a system service:

  ```
  cloudflared service install
  ```

cmake

Cross-platform build automation system, that generates recipes for native build systems.

More information: https://cmake.org/cmake/help/latest/manual/cmake.1.html

- Generate a build recipe in the current directory with `CMakeLists.txt` from a project directory:

  ```
  cmake {{path/to/project_directory}}
  ```

- Generate a build recipe, with build type set to `Release` with CMake variable:

  ```
  cmake {{path/to/project_directory}} -D {{CMAKE_BUILD_TYPE=Release}}
  ```

- Use a generated recipe in a given directory to build artifacts:

  ```
  cmake --build {{path/to/build_directory}}
  ```

- Install the build artifacts into `/usr/local/` and strip debugging symbols:

  ```
  cmake --install {{path/to/build_directory}} --strip
  ```

- Install the build artifacts using the custom prefix for paths:

  ```
  cmake --install {{path/to/build_directory}} --strip --prefix {{path/to/directory}}
  ```

- Run a custom build target:

  ```
  cmake --build {{path/to/build_directory}} --target {{target_name}}
  ```

cmark

Converts CommonMark Markdown formatted text to other formats.

More information: https://github.com/commonmark/cmark.

- Render a CommonMark Markdown file to HTML:

  ```
  cmark --to html {{filename.md}}
  ```

- Convert data from standard input to LaTeX:

  ```
  cmark --to latex
  ```

- Convert straight quotes to smart quotes:

  ```
  cmark --smart --to html {{filename.md}}
  ```

- Validate UTF-8 characters:

  ```
  cmark --validate-utf8 {{filename.md}}
  ```

cmatrix

Shows a scrolling Matrix like screen in the terminal.

More information: https://github.com/abishekvashok/cmatrix.

- Enable asynchronous scrolling:

  ```
  cmatrix -a
  ```

- Display red text:

  ```
  cmatrix -C {{red}}
  ```

- Enable rainbow mode:

  ```
  cmatrix -r
  ```

- Set screen update delay to 20 centiseconds (200 milliseconds):

  ```
  cmatrix -u {{20}}
  ```

cmp

Compare two files byte by byte.

More information: https://www.gnu.org/software/diffutils/manual/html_node/Invoking-cmp.html.

- Find the byte and line number of the first difference between two files:

  ```
  cmp {{path/to/file1}} {{path/to/file2}}
  ```

- Find the byte number and differing bytes of every difference:

  ```
  cmp -l {{path/to/file1}} {{path/to/file2}}
  ```

code

Visual Studio Code.

More information: https://github.com/microsoft/vscode.

- Open VS Code:

  ```
  code
  ```

- Open the current directory in VS Code:

  ```
  code .
  ```

- Open a file or directory in VS Code:

  ```
  code {{path/to/file_or_directory}}
  ```

- Open a file or directory in the currently open VS Code window:

  ```
  code --reuse-window {{path/to/file_or_directory}}
  ```

- Compare two files in VS Code:

  ```
  code -d {{file1}} {{file2}}
  ```

- Open VS Code with super user (sudo) permissions:

  ```
  sudo code {{path/to/file_or_directory}} --user-data-dir
  ```

codespell

Spellchecker for source code.

More information: https://github.com/codespell-project/codespell.

- Check for typos in all text files in the current directory, recursively:

  ```
  codespell
  ```

- Correct all typos found in-place:

  ```
  codespell --write-changes
  ```

- Skip files with names that match the specified pattern (accepts a comma-separated list of patterns using wildcards):

  ```
  codespell --skip "{{pattern}}"
  ```

- Use a custom dictionary file when checking (`--dictionary` can be used multiple times):

  ```
  codespell --dictionary {{path/to/file.txt}}
  ```

- Do not check words that are listed in the specified file:

  ```
  codespell --ignore-words {{path/to/file.txt}}
  ```

- Do not check the specified words:

  ```
  codespell --ignore-words-list {{words,to,ignore}}
  ```

- Print 3 lines of context around, before or after each match:

  ```
  codespell --{{context|before-context|after-context}} {{3}}
  ```

- Check file names for typos, in addition to file contents:

  ```
  codespell --check-filenames
  ```

coffee

Executes CoffeeScript scripts or compiles them into JavaScript.

More information: https://coffeescript.org#cli

- Run a script:

```
coffee {{path/to/file.coffee}}
```

- Compile to JavaScript and save to a file with the same name:

```
coffee --compile {{path/to/file.coffee}}
```

- Compile to JavaScript and save to a given output file:

```
coffee --compile {{path/to/file.coffee}} --output {{path/to/file.js}}
```

- Run interactive REPL:

```
coffee --interactive
```

- Watch script for changes and re-run script:

```
coffee --watch {{path/to/file.coffee}}
```

colordiff

A tool to colorize diff output.

The Perl script colordiff is a wrapper for `diff` and produces the same output but with pretty syntax highlighting. Colour schemes can be customized.

More information: https://github.com/kimmel/colordiff.

- Compare files:

  ```
  colordiff {{file1}} {{file2}}
  ```

- Output in two columns:

  ```
  colordiff -y {{file1}} {{file2}}
  ```

- Ignore case differences in file contents:

  ```
  colordiff -i {{file1}} {{file2}}
  ```

- Report when two files are the same:

  ```
  colordiff -s {{file1}} {{file2}}
  ```

- Ignore white spaces:

  ```
  colordiff -w {{file1}} {{file2}}
  ```

colorpicker

A minimalist X 1 1 colorpicker.

Any mouse gesture except left click will exit the program.

More information: https://github.com/ym1_2_3_4_/colorpicker

- Launch colorpicker and print the hexadecimal and R G B value of each clicked pixel to stdout:

  ```
  colorpick er
  ```

- O nly print the color of one clicked pixel and then exit:

  ```
  colorpick er - - one- shot
  ```

- P rint the color of each clicked pixel and q uit when a key is pressed:

  ```
  colorpick er - - q u it- on- k eypress
  ```

- O nly print the R G B value:

  ```
  colorpick er - - rg b
  ```

- O nly print the hexadecimal value:

  ```
  colorpick er - - hex
  ```

column

Format standard input or a file into multiple columns.

Columns are filled before rows; the default separator is a whitespace.

More information: https://manned.org/column.

- Format the output of a command for a 30 characters wide display:

```
printf "header1 header2\nbar foo\n" | column --output-width {{30}}
```

- Split columns automatically and auto-align them in a tabular format:

```
printf "header1 header2\nbar foo\n" | column --table
```

- Specify the column delimiter character for the `--table` option (e.g. `","` for csv) (defaults to whitespace):

```
printf "header1,header2\nbar,foo\n" | column --table --separator {{,}}
```

- Fill rows before filling columns:

```
printf "header1\nbar\nfoobar\n" | column --output-width {{30}} --fillrows
```

comm

Select or reject lines common to two files. Both files must be sorted.

More information: https://www.gnu.org/software/coreutils/comm.

- Produce three tab-separated columns: lines only in first file, lines only in second file and common lines:

```
comm {{file1}} {{file2}}
```

- Print only lines common to both files:

```
comm -12 {{file1}} {{file2}}
```

- Print only lines common to both files, reading one file from stdin:

```
cat {{file1}} | comm -12 - {{file2}}
```

- Get lines only found in first file, saving the result to a third file:

```
comm -23 {{file1}} {{file2}} > {{file1_only}}
```

- Print lines only found in second file, when the files aren't sorted:

```
comm -13 <(sort {{file1}}) <(sort {{file2}})
```

command

Command forces the shell to execute the program and ignore any functions, builtins and aliases with the same name.

More information: https://manned.org/command.

- E xecute the `ls` program literally, even if an `ls` alias exists:

  ```
  com m and {{ls}}
  ```

- Display the path to the executable or the alias definition of a specific command:

  ```
  com m and - v {{com m and_nam e}}
  ```

compare

View the difference between 2 images.

More information: https://imagemagick.org/script/compare.php.

- Compare 2 images:

```
compare {{image1.png}} {{image2.png}} {{diff.png}}
```

- Compare 2 images using a custom metric:

```
compare -verbose -metric {{PSNR}} {{image1.png}}
{{image2.png}} {{diff.png}}
```

complete

Provides argument autocompletion to shell commands.

More information: https://www.gnu.org/software/bash/manual/html_node/Programmable-Completion-Builtins.html

- Apply a function that performs autocompletion to a command:

  ```
  complete -F {{function}} {{command}}
  ```

- Apply a command that performs autocompletion to another command:

  ```
  complete -C {{autocomplete_command}} {{command}}
  ```

- Apply autocompletion without appending a space to the completed word:

  ```
  complete -o nospace -F {{function}} {{command}}
  ```

composer-req uire-checker

A CLI tool to analyse Composer dependencies for soft dependencies.

More information: https://github.com/maglnet/ComposerR eq uireChecker

- Analyse a Composer J S O N file:

  ```
  com poser- req u ire- check er check {{path/to/com poser. j son}}
  ```

- Analyse a Composer J S O N file with a specific configuration:

  ```
  com poser- req u ire- check er check - - config - file {{path/to/
  config . j son}} {{path/to/com poser. j son}}
  ```

composer

A package-based dependency manager for PHP projects.

More information: https://getcomposer.org/.

- Interactively create a `composer.json` file:

  ```
  composer init
  ```

- Add a package as a dependency for this project, adding it to `composer.json`:

  ```
  composer require {{user/package name}}
  ```

- Install all the dependencies in this project's `composer.json` and create `composer.lock`:

  ```
  composer install
  ```

- Uninstall a package from this project, removing it as a dependency from `composer.json`:

  ```
  composer remove {{user/package name}}
  ```

- Update all the dependencies in this project's `composer.json` and note versions in `composer.lock` file:

  ```
  composer update
  ```

- Update composer lock only after updating `composer.json` manually:

  ```
  composer update --lock
  ```

- Learn more about why a dependency can't be installed:

  ```
  composer why-not {{user/package name}}
  ```

- Update composer to its latest version:

  ```
  composer self-update
  ```

conan frogarian

Displays the conan frogarian.

More information: https://docs.conan.io/.

- Display the conan frogarian:

```
conan frog arian
```

conan

The open source, decentralized and cross-platform package manager to create and share all your native binaries.

More information: https://conan.io/.

- Install packages based on `conanfile.txt`

  ```
  conan install {{.}}
  ```

- Install packages and create configuration files for a specific generator:

  ```
  conan install -g {{generator}}
  ```

- Install packages, building from source:

  ```
  conan install {{.}} --build
  ```

- Search for locally installed packages:

  ```
  conan search {{package}}
  ```

- Search for remote packages:

  ```
  conan search {{package}} -r {{remote}}
  ```

- List remotes:

  ```
  conan remote list
  ```

conda create

Create new conda environments.

More information: https://docs.conda.io/projects/conda/en/latest/commands/create.html.

- Create a new environment named `py39`, and install Python 3.9 and numpy v1.11 or above in it:

```
conda create --yes --name {{py39}} python={{3.9}} "{{numpy>=1.11}}"
```

- Make exact copy of an environment:

```
conda create --clone {{py39}} --name {{py39-copy}}
```

- Create a new environment with a specified name and install a given package:

```
conda create --name {{envname}} {{package name}}
```

conda

P ackage, dependency and environment management for any programming language.

More information: https://github.com/conda/conda.

- Create a new environment, installing named packages into it:

  ```
  conda create - - nam e {{env ironm ent_nam e}} {{python= 3 . 9
  m atplotlib }}
  ```

- List all environments:

  ```
  conda info - - env s
  ```

- Load an environment:

  ```
  conda {{activ ate env ironm ent_nam e}}
  ```

- U nload an environment:

  ```
  conda {{deactiv ate}}
  ```

- Delete an environment (remove all packages):

  ```
  conda rem ov e - - nam e {{env ironm ent_nam e}} - - all
  ```

- Install packages into the current environment:

  ```
  conda install {{python= 3 . 4  nu m py}}
  ```

- List currently installed packages in current environment:

  ```
  conda list
  ```

- Delete unused packages and caches:

  ```
  conda clean - - all
  ```

consul-kv

Distributed key-value store with health checking and service discovery.

More information: https://learn.hashicorp.com/consul/getting-started/kv.

- Read a value from the key-value store:

  ```
  consul kv get {{key}}
  ```

- Store a new key-value pair:

  ```
  consul kv put {{key}} {{value}}
  ```

- Delete a key-value pair:

  ```
  consul kv delete {{key}}
  ```

consul

Distributed key-value store with health checking and service discovery.

More information: https://www.consul.io/commands.

- Check the Consul version:

  ```
  consul --version
  ```

- Show general help:

  ```
  consul --help
  ```

- Show help for a sub-command:

  ```
  consul {{sub-command}} --help
  ```

convert

ImageMagick image conversion tool.

More information: https://imagemagick.org/script/convert.php.

- Convert an image from JPG to PNG:

```
convert {{image.jpg}} {{image.png}}
```

- Scale an image 50% its original size:

```
convert {{image.png}} -resize 50% {{image2.png}}
```

- Scale an image keeping the original aspect ratio to a maximum dimension of 640x480:

```
convert {{image.png}} -resize 640x480 {{image2.png}}
```

- Horizontally append images:

```
convert {{image1.png}} {{image2.png}} {{image3.png}} +appen
{{image123.png}}
```

- Vertically append images:

```
convert {{image1.png}} {{image2.png}} {{image3.png}} -appen
{{image123.png}}
```

- Create a gif from a series of images with 100 ms delay between them:

```
convert {{image1.png}} {{image2.png}} {{image3.png}} -delay
{{10}} {{animation.gif}}
```

- Create an image with nothing but a solid background:

```
convert -size {{800x600}} "xc:{{#ff0000}}" {{image.png}}
```

- Create a favicon from several images of different sizes:

```
convert {{image1.png}} {{image2.png}} {{image3.png}}
{{image.ico}}
```

convmv

Convert filenames (NOT file content) from one encoding to another.

More information: https://www.j3e.de/linux/convmv/man/

- Test filename encoding conversion (don't actually change the filename):

```
convmv -f {{from_encoding}} -t {{to_encoding}} {{input_file}}
```

- Convert filename encoding and rename the file to the new encoding:

```
convmv -f {{from_encoding}} -t {{to_encoding}} --notest
{{input_file}}
```

copyq

Clipboard manager with advanced features.

More information: https://hluk.github.io/CopyQ/.

- Launch CopyQ to store clipboard history:

```
copyq
```

- Show current clipboard content:

```
copyq clipboard
```

- Insert raw text into the clipboard history:

```
copyq add -- {{text1}} {{text2}} {{text3}}
```

- Insert text containing escape sequences ('\n', '\t') into the clipboard history:

```
copyq add {{firstline\nsecondline}}
```

- Print the content of the first 3 items in the clipboard history:

```
copyq read 0 1 2
```

- Copy a file's contents into the clipboard:

```
copyq copy < {{file.txt}}
```

- Copy a JPEG image into the clipboard:

```
copyq copy image/jpeg < {{image.jpg}}
```

cordova

Mobile apps with HTML, CSS & JS.

More information: https://cordova.apache.org/docs/en/latest/guide/cli/.

- Create a cordova project:

  ```
  cordova create {{path}} {{package_name}} {{project_name}}
  ```

- Display the current workspace status:

  ```
  cordova info
  ```

- Add a cordova platform:

  ```
  cordova platform add {{platform}}
  ```

- Remove a cordova platform:

  ```
  cordova platform remove {{platform}}
  ```

- Add a cordova plugin:

  ```
  cordova plugin add {{pluginid}}
  ```

- Remove a cordova plugin:

  ```
  cordova plugin remove {{pluginid}}
  ```

cotton

Markdown test specification runner.

More information: https://github.com/chonla/cotton.

- Use a specific base URL:

  ```
  cotton -u {{base_url}} {{file}}.md
  ```

- Disable certificate verification (insecure mode):

  ```
  cotton -u {{base_url}} -i {{file}}.md
  ```

- Stop running when a test fails:

  ```
  cotton -u {{base_url}} -s {{file}}.md
  ```

couchdb

Command-line interface for Apache CouchDB database server.

More information: https://couchdb.apache.org.

- Start couchdb:

  ```
  couchdb
  ```

- Start couchdb interactive shell:

  ```
  couchdb -i
  ```

- Start couchdb as a background process:

  ```
  couchdb -b
  ```

- Kill the background process (Note: It will respawn if needed):

  ```
  couchdb -k
  ```

- Shutdown the background process:

  ```
  couchdb -d
  ```

cowsay

Print ASCII art (by default a cow) saying or thinking something.

More information: https://github.com/tnalpgge/rank-amateur-cowsay.

- Print an ASCII cow saying "hello, world":

  ```
  cowsay "{{hello, world}}"
  ```

- Print an ASCII cow saying text from stdin:

  ```
  echo "{{hello, world}}" | cowsay
  ```

- List all available art types:

  ```
  cowsay -l
  ```

- Print the specified ASCII art saying "hello, world":

  ```
  cowsay -f {{art}} "{{hello, world}}"
  ```

- Print a dead thinking ASCII cow:

  ```
  cowthink -d "{{I'm just a cow, not a great thinker...}}"
  ```

- Print an ASCII cow with custom eyes saying "hello, world":

  ```
  cowsay -e {{characters}} "{{hello, world}}"
  ```

cp

Copy files and directories.

More information: https://www.gnu.org/software/coreutils/cp.

- Copy a file to another location:

```
cp {{path/to/sou rce_file. ext}} {{path/to/targ et_file. ext}}
```

- Copy a file into another directory, keeping the filename:

```
cp {{path/to/sou rce_file. ext}} {{path/to/
targ et_parent_directory}}
```

- R ecursively copy a directory' s contents to another location (if the destination exists, the directory is copied inside it):

```
cp - R  {{path/to/sou rce_directory}} {{path/to/
targ et_directory}}
```

- Copy a directory recursively, in verbose mode (shows files as they are copied):

```
cp - v R  {{path/to/sou rce_directory}} {{path/to/
targ et_directory}}
```

- Copy text files to another location, in interactive mode (prompts user before overwriting):

```
cp - i {{* . txt}} {{path/to/targ et_directory}}
```

- F ollow symbolic links before copying:

```
cp - L  {{link }} {{path/to/targ et_directory}}
```

cpdf

CLI to manipulate existing PDF files in a variety of ways.

More information: https://www.coherentpdf.com/cpdfmanual/cpdfmanual.html.

- Select pages 1, 2, 3 and 6 from a source document and write those to a destination document:

  ```
  cpdf {{path/to/source_document.pdf}} {{1-3,6}} -o {{path/to/destination_document.pdf}}
  ```

- Merge two documents into a new one:

  ```
  cpdf -merge {{path/to/source_document_one.pdf}} {{path/to/source_document_two.pdf}} -o {{path/to/destination_document.pdf}}
  ```

- Show the bookmarks of a document:

  ```
  cpdf -list-bookmarks {{path/to/document.pdf}}
  ```

- Split a document into ten-page chunks, writing them to `chunk001.pdf`, `chunk002.pdf`, etc:

  ```
  cpdf -split {{path/to/document.pdf}} -o {{path/to/chunk%%%.pdf}} -chunk {{10}}
  ```

- Encrypt a document using 128 bit encryption, providing `fred` as owner password and `joe` as user password:

  ```
  cpdf -encrypt {{128bit}} {{fred}} {{joe}} {{path/to/source_document.pdf}} -o {{path/to/encrypted_document.pdf}}
  ```

- Decrypt a document using the owner password `fred`:

  ```
  cpdf -decrypt {{path/to/encrypted_document.pdf}} owner={{fred}} -o {{path/to/decrypted_document.pdf}}
  ```

- Show the annotations of a document:

  ```
  cpdf -list-annotations {{path/to/document.pdf}}
  ```

- Create a new document from an existing one with additional metadata:

  ```
  cpdf -set-metadata {{path/to/metadata.xml}} {{path/to/source_document.pdf}} -o {{path/to/destination_document.pdf}}
  ```

cpio

Copies files in and out of archives.

Supports the following archive formats: cpio's custom binary, old ASCII, new ASCII, crc, HPUX binary, HPUX old ASCII, old tar, and POSIX.1 tar.

More information: https://www.gnu.org/software/cpio.

- Take a list of file names from standard input and add them [o]into an archive in cpio's binary format:

```
echo "{{file1}} {{file2}} {{file3}}" | cpio -o > {{archive.cpio}}
```

- Copy all files and directories in a directory and add them [o]into an archive, in [v]erbose mode:

```
find {{path/to/directory}} | cpio -ov > {{archive.cpio}}
```

- P[i]ck all files from an archive, generating [d]irectories where needed, in [v]erbose mode:

```
cpio -idv < {{archive.cpio}}
```

cppcheck

A static analysis tool for C/C++ code.

Instead of syntax errors, it focuses on the types of bugs that compilers normally do not detect.

More information: http://cppcheck.sourceforge.net.

- Recursively check the current directory, showing progress on the screen and logging error messages to a file:

  ```
  cppcheck . 2> cppcheck.log
  ```

- Recursively check a given directory, and don't print progress messages:

  ```
  cppcheck --quiet {{path/to/directory}}
  ```

- Check a given file, specifying which tests to perform (by default only errors are shown):

  ```
  cppcheck --enable={{error|warning|style|performance|portability|information|all}} {{path/to/file.cpp}}
  ```

- List available tests:

  ```
  cppcheck --errorlist
  ```

- Check a given file, ignoring specific tests:

  ```
  cppcheck --suppress={{test_id1}} --suppress={{test_id2}} {{path/to/file.cpp}}
  ```

- Check the current directory, providing paths for include files located outside it (e.g. external libraries):

  ```
  cppcheck -I {{include/directory_1}} -I {{include/directory_2}} .
  ```

- Check a Microsoft Visual Studio project (`*.vcxproj`) or solution (`*.sln`):

  ```
  cppcheck --project={{path/to/project.sln}}
  ```

cppclean

Find unused code in C++ projects.

More information: https://github.com/myint/cppclean.

- Run in a project's directory:

  ```
  cppclean {{path/to/project}}
  ```

- Run on a project where the headers are in the `inc1/` and `inc2/` directories:

  ```
  cppclean {{path/to/project}} --include-path={{inc1}} --include-path={{inc2}}
  ```

- Run on a specific file `main.cpp`:

  ```
  cppclean {{main.cpp}}
  ```

- Run on the current directory, excluding the "build" directory:

  ```
  cppclean {{.}} --exclude={{build}}
  ```

cradle deploy

Manage Cradle deployments.

More information: https://cradlephp.github.io/docs/3_B_-Reference-Command-Line-Tools.html#_deploy

- Deploy Cradle to a server:

  ```
  cradle deploy production
  ```

- Deploy static assets to Amazon S3:

  ```
  cradle deploy s3
  ```

- Deploy static assets including the Yarn "components" directory:

  ```
  cradle deploy s3 --include-yarn
  ```

- Deploy static assets including the "upload" directory:

  ```
  cradle deploy s3 --include-upload
  ```

cradle elastic

Manage the Elasticsearch instances for a Cradle instance.

More information: https://cradlephp.github.io/docs/3.B.-Reference-Command-Line-Tools.html#elastic

- Truncate the Elasticsearch index:

  ```
  cradle elastic flush
  ```

- Truncate the Elasticsearch index for a specific package:

  ```
  cradle elastic flush {{package_name}}
  ```

- Submit the Elasticsearch schema:

  ```
  cradle elastic map
  ```

- Submit the Elasticsearch schema for a specific package:

  ```
  cradle elastic map {{package_name}}
  ```

- Populate the Elasticsearch indices for all packages:

  ```
  cradle elastic populate
  ```

- Populate the Elasticsearch indices for a specific package:

  ```
  cradle elastic populate {{package_name}}
  ```

cradle install

Installs the Cradle PHP framework components.

More information: https://cradlephp.github.io/docs/3.B.-Reference-Command-Line-Tools.html#install

- Install Cradle's components (User will be prompted for further details):

  ```
  cradle install
  ```

- Forcefully overwrite files:

  ```
  cradle install --force
  ```

- Skip running SQL migrations:

  ```
  cradle install --skip-sql
  ```

- Skip running package updates:

  ```
  cradle install --skip-versioning
  ```

- Use specific database details:

  ```
  cradle install -h {{hostname}} -u {{username}} -p {{password}}
  ```

cradle package

Manage packages for a Cradle instance.

More information: https://cradlephp.github.io/docs/3_B_-Reference-Command-Line-Tools.html#_package

- Display a list of available packages:

  ```
  cradle package list
  ```

- Search for a package:

  ```
  cradle package search {{package}}
  ```

- Install a package from Packagist:

  ```
  cradle package install {{package}}
  ```

- Install a specific version of a package:

  ```
  cradle package install {{package}} {{version}}
  ```

- Update a package:

  ```
  cradle package update {{package}}
  ```

- Update a package to a specific version:

  ```
  cradle package update {{package}} {{version}}
  ```

- Remove a specific package:

  ```
  cradle package remove {{package}}
  ```

cradle sql

Manage Cradle SQL databases.

More information: https://cradlephp.github.io/docs/3.B.-Reference-Command-Line-Tools.html#sql

- Rebuild the database schema:

  ```
  cradle sql build
  ```

- Rebuild the database schema for a specific package:

  ```
  cradle sql build {{package name}}
  ```

- Empty the entire database:

  ```
  cradle sql flush
  ```

- Empty the database tables for a specific package:

  ```
  cradle sql flush {{package name}}
  ```

- Populate the tables for all packages:

  ```
  cradle sql populate
  ```

- Populate the tables for a specific package:

  ```
  cradle sql populate {{package name}}
  ```

cradle

The Cradle PHP framework.

See `cradle-install`, `cradle-build`, `cradle-deploy` and other pages for additional information.

More information: https://cradlephp.github.io.

- Connect to a server:

  `cradle connect {{server_name}}`

- Display general help:

  `cradle help`

- Display help for a specific command:

  `cradle {{command}} help`

- Execute a Cradle command:

  `cradle {{command}}`

croc

Send and receive files easily and securely over any network.

More information: https://github.com/schollz/croc

- Send a file or directory:

```
croc send {{path/to/file_or_directory}}
```

- Send a file or directory with a specific passphrase:

```
croc send --code {{passphrase}} {{path/to/file_or_directory}}
```

- Receive a file or directory on receiving machine:

```
croc {{passphrase}}
```

- Send and connect over a custom relay:

```
croc --relay {{ip_to_relay}} send {{path/to/
file_or_directory}}
```

- Receive and connect over a custom relay:

```
croc --relay {{ip_to_relay}} {{passphrase}}
```

- Host a croc relay on the default ports:

```
croc relay
```

- Display parameters and options for a croc command:

```
croc {{send|relay}} --help
```

cronic

Bash script for wrapping cron jobs to prevent excess email sending.

More information: https://habilis.net/cronic/.

- Call a command and display its output if it returns a non-zero exit code:

  `cronic {{command}}`

crontab

Schedule cron jobs to run on a time interval for the current user.

Job definition format: "(min) (hour) (day_of_month) (month) (day_of_week) command_to_execute".

More information: https://manned.org/crontab.

- Edit the crontab file for the current user:

```
crontab -e
```

- Edit the crontab file for a specific user:

```
sudo crontab -e -u {{user}}
```

- Replace the current crontab with the contents of the given file:

```
crontab {{path/to/file}}
```

- View a list of existing cron jobs for current user:

```
crontab -l
```

- Remove all cron jobs for the current user:

```
crontab -r
```

- Sample job which runs at 10:00 every day (* means any value):

```
0 10 * * * {{command_to_execute}}
```

- Sample job which runs every minute on the 3rd of April:

```
* * 3 Apr * {{command_to_execute}}
```

- Sample job which runs a certain script at 02:30 every Friday:

```
30 2 * * Fri {{/absolute/path/to/script.sh}}
```

crunch

Wordlist generator.

More information: https://sourceforge.net/projects/crunch-wordlist/

- Output a list of words of length 1 to 3 with only lowercase characters:

  ```
  crunch {{1}} {{3}}
  ```

- Output a list of hexadecimal words of length 8:

  ```
  crunch {{8}} {{8}} {{0123456789abcdef}}
  ```

- Output a list of all permutations of abc (lengths are not processed):

  ```
  crunch {{1}} {{1}} -p {{abc}}
  ```

- Output a list of all permutations of the given strings (lengths are not processed):

  ```
  crunch {{1}} {{1}} -p {{abc}} {{def}} {{ghi}}
  ```

- Output a list of words generated according to the given pattern and a maximum number of duplicate letters:

  ```
  crunch {{5}} {{5}} {{abcde123}} -t {{@@@12}} -d 2@
  ```

- Write a list of words in chunk files of a given size, starting with the given string:

  ```
  crunch {{3}} {{5}} -o {{START}} -b {{10kb}} -s {{abc}}
  ```

- Write a list of words stopping with the given string and inverting the wordlist:

  ```
  crunch {{1}} {{5}} -o {{START}} -e {{abcde}} -i
  ```

- Write a list of words in compressed chunk files with a specified number of words:

  ```
  crunch {{1}} {{5}} -o {{START}} -c {{1000}} -z {{gzip|bzip|lzma|7z}}
  ```

cryfs

A cryptographic filesystem for the cloud.

More information: https://www.cryfs.org/.

- Mount an encrypted filesystem. T he initializ ation wiz ard will be started on the first execution:

  ```
  cryfs {{path/to/cipher_dir}} {{path/to/m ou nt_point}}
  ```

- U nmount an encrypted filesystem:

  ```
  cryfs- u nm ou nt {{path/to/m ou nt_point}}
  ```

- Automatically unmount after ten minutes of inactivity:

  ```
  cryfs - - u nm ou nt- idle {{1 0 }} {{path/to/cipher_dir}} {{path/to/
  m ou nt_point}}
  ```

- S how a list of supported ciphers:

  ```
  cryfs - - show - ciphers
  ```

crystal

Tool for managing Crystal source code.

More information: https://crystal-lang.org/reference/using_the_compiler

- Run a Crystal file:

  ```
  crystal {{path/to/file.cr}}
  ```

- Compile a file and all dependencies to a single executable:

  ```
  crystal build {{path/to/file.cr}}
  ```

- Start a local interactive server for testing the language:

  ```
  crystal play
  ```

- Create a project directory for a Crystal application:

  ```
  crystal init app {{application_name}}
  ```

- Display all help options:

  ```
  crystal help
  ```

csc

The Microsoft C# Compiler.

More information: https://docs.microsoft.com/dotnet/csharp/language-reference/compiler-options/command-line-building-with-csc-exe.

- Compile one or more C# files to a CIL executable:

```
csc {{path/to/input_file_a.cs}} {{path/to/input_file_b.cs}}
```

- Specify the output filename:

```
csc /out:{{path/to/filename}} {{path/to/input_file.cs}}
```

- Compile into a `.dll` library instead of an executable:

```
csc /target:library {{path/to/input_file.cs}}
```

- Reference another assembly:

```
csc /reference:{{path/to/library.dll}} {{path/to/
input_file.cs}}
```

- Embed a resource:

```
csc /resource:{{path/to/resource_file}} {{path/to/
input_file.cs}}
```

- Automatically generate XML documentation:

```
csc /doc:{{path/to/output.xml}} {{path/to/input_file.cs}}
```

- Specify an icon:

```
csc /win32icon:{{path/to/icon.ico}} {{path/to/input_file.cs}}
```

- Strongly-name the resulting assembly with a keyfile:

```
csc /keyfile:{{path/to/keyfile}} {{path/to/input_file.cs}}
```

csslint

A linter for CS S code.

More information: https://github.com/CS S Lint/csslint/wiki/Command-line-interface

- Lint a single CS S file:

  ```
  csslint {{file. css}}
  ```

- Lint multiple CS S files:

  ```
  csslint {{file1 . css}} {{file2 . css}} {{file3 . css}}
  ```

- List all possible style rules:

  ```
  csslint - - list- ru les
  ```

- S pecify certain rules as errors (which result in a non-z ero exit code):

  ```
  csslint - - errors= {{errors, u niv ersal- selector, im ports}}
  {{file. css}}
  ```

- S pecify certain rules as warnings:

  ```
  csslint - - w arning s= {{b ox- sizing , selector- m ax, floats}}
  {{file. css}}
  ```

- S pecify certain rules to completely ignore:

  ```
  csslint - - ig nore= {{ids, ru les- cou nt, shorthand}} {{file. css}}
  ```

csvclean

Finds and cleans common syntax errors in CSV files.

Included in csvkit.

More information: https://csvkit.readthedocs.io/en/latest/scripts/csvclean.html.

- Clean a CSV file:

  ```
  csv clean {{bad.csv}}
  ```

- List locations of syntax errors in a CSV file:

  ```
  csv clean -n {{bad.csv}}
  ```

csvcut

Filter and truncate CSV files. Like Unix `cut` command, but for tabular data.

Included in csvkit.

More information: https://csvkit.readthedocs.io/en/latest/scripts/csvcut.html.

- Print indices and names of all columns:

  ```
  csvcut -n {{data.csv}}
  ```

- Extract the first and third columns:

  ```
  csvcut -c {{1,3}} {{data.csv}}
  ```

- Extract all columns except the fourth one:

  ```
  csvcut -C {{4}} {{data.csv}}
  ```

- Extract the columns named "id" and "first name" (in that order):

  ```
  csvcut -c {{id,"first name"}} {{data.csv}}
  ```

csvformat

Convert a CSV file to a custom output format.

Included in csvkit.

More information: https://csvkit.readthedocs.io/en/latest/scripts/csvformat.html.

- Convert to a tab-delimited file (TSV):

  ```
  csvformat -T {{data.csv}}
  ```

- Convert delimiters to a custom character:

  ```
  csvformat -D "{{custom_character}}" {{data.csv}}
  ```

- Convert line endings to carriage return (^M) + line feed:

  ```
  csvformat -M "{{\r\n}}" {{data.csv}}
  ```

- Minimize use of quote characters:

  ```
  csvformat -U 0 {{data.csv}}
  ```

- Maximize use of quote characters:

  ```
  csvformat -U 1 {{data.csv}}
  ```

csvgrep

Filter CSV rows with string and pattern matching.

Included in csvkit.

More information: https://csvkit.readthedocs.io/en/latest/scripts/csvgrep.html.

- Find rows that have a certain string in column 1:

```
csvgrep -c {{1}} -m {{string_to_match}} {{data.csv}}
```

- Find rows in which columns 3 or 4 match a certain regular expression:

```
csvgrep -c {{3,4}} -r {{regular_expression}} {{data.csv}}
```

- Find rows in which the "name" column does NOT include the string "John Doe":

```
csvgrep -i -c {{name}} -m "{{John Doe}}" {{data.csv}}
```

csvkit

Manipulation toolkit for CSV files.

See the individual commands: `csvclean`, `csvcut`, `csvformat`, `csvgrep`, `csvlook`, `csvpy`, `csvsort`, `csvstat`

More information: https://csvkit.readthedocs.io/en/0.9.1/cli.html

- Run a command on a CSV file with a custom delimiter:

  ```
  {{cmd}} -d {{delimiter}} {{filename.csv}}
  ```

- Run a command on a CSV file with a tab as a delimiter (overrides -d):

  ```
  {{cmd}} -t {{filename.csv}}
  ```

- Run a command on a CSV file with a custom quote character:

  ```
  {{cmd}} -q {{quote_char}} {{filename.csv}}
  ```

- Run a command on a CSV file with no header row:

  ```
  {{cmd}} -H {{filename.csv}}
  ```

csvlook

Render a CSV file in the console as a fixed-width table.

Included in csvkit.

More information: https://csvkit.readthedocs.io/en/latest/scripts/csvlook.html.

- View a CSV file:

```
csvlook {{data.csv}}
```

csvpy

Loads a CSV file into a Python shell.

Included in csvkit.

More information: https://csvkit.readthedocs.io/en/latest/scripts/csvpy.html.

- Load a CSV file into a `CSVKitReader` object:

  ```
  csvpy {{data.csv}}
  ```

- Load a CSV file into a `CSVKitDictReader` object:

  ```
  csvpy --dict {{data.csv}}
  ```

csvsort

Sorts CSV files.

Included in csvkit.

More information: https://csvkit.readthedocs.io/en/latest/scripts/csvsort.html.

- Sort a CSV file by column 9:

  ```
  csvsort -c {{9}} {{data.csv}}
  ```

- Sort a CSV file by the "name" column in descending order:

  ```
  csvsort -r -c {{name}} {{data.csv}}
  ```

- Sort a CSV file by column 2, then by column 4:

  ```
  csvsort -c {{2,4}} {{data.csv}}
  ```

- Sort a CSV file without inferring data types:

  ```
  csvsort --no-inference -c {{columns}} {{data.csv}}
  ```

csvsql

Generate SQL statements for a CSV file or execute those statements directly on a database.

Included in csvkit.

More information: https://csvkit.readthedocs.io/en/latest/scripts/csvsql.html

- Generate a CREATE TABLE statement for a CSV file:

  ```
  csvsql {{path/to/data.csv}}
  ```

- Import a CSV file into an SQL database:

  ```
  csvsql --insert --db "{{mysql://user:password@host/database}}" {{data.csv}}
  ```

- Run an SQL query on a CSV file:

  ```
  csvsql --query "{{select * from 'data'}}" {{data.csv}}
  ```

csvstat

Print descriptive statistics for all columns in a CSV file.

Included in csvkit.

More information: https://csvkit.readthedocs.io/en/latest/scripts/csvstat.html.

- Show all stats for all columns:

  ```
  csvstat {{data.csv}}
  ```

- Show all stats for columns 2 and 4:

  ```
  csvstat -c {{2,4}} {{data.csv}}
  ```

- Show sums for all columns:

  ```
  csvstat --sum {{data.csv}}
  ```

- Show the max value length for column 3:

  ```
  csvstat -c {{3}} --len {{data.csv}}
  ```

- Show the number of unique values in the "name" column:

  ```
  csvstat -c {{name}} --unique {{data.csv}}
  ```

csvtool

Utility to filter and extract data from CSV formatted sources.

More information: https://github.com/maroofi/csvtool.

- Extract the second column from a CSV file:

```
csvtool --column {{2}} {{path/to/file.csv}}
```

- Extract the second and fourth columns from a CSV file:

```
csvtool --column {{2,4}} {{path/to/file.csv}}
```

- Extract lines from a CSV file where the second column exactly matches 'Foo':

```
csvtool --column {{2}} --search '{{^Foo$}}' {{path/to/file.csv}}
```

- Extract lines from a CSV file where the second column starts with 'Bar':

```
csvtool --column {{2}} --search '{{^Bar}}' {{path/to/file.csv}}
```

- Find lines in a CSV file where the second column ends with 'Baz' and then extract the third and sixth columns:

```
csvtool --column {{2}} --search '{{Baz$}}' {{path/to/file.csv}} | csvtool --no-header --column {{3,6}}
```

ctags

Generates an index (or tag) file of language objects found in source files for many popular programming languages.

More information: https://ctags.io/.

- Generate tags for a single file, and output them to a file named "tags" in the current directory, overwriting the file if it exists:

  ```
  ctags {{path/to/file}}
  ```

- Generate tags for all files in the current directory, and output them to a specific file, overwriting the file if it exists:

  ```
  ctags -f {{filename}} *
  ```

- Generate tags for all files in the current directory and all subdirectories:

  ```
  ctags --recurse
  ```

ctest

CMake test driver program.

More information: https://gitlab.kitware.com/cmake/community/wikis/doc/ctest/Testing-With-CTest

- Run all tests defined in the CMake project, executing 4 jobs at a time in parallel:

  ```
  ctest -j {{4}} --output-on-failure
  ```

- Show a list of available tests:

  ```
  ctest -N
  ```

- Run a single test based on its name, or filter on a regular expression:

  ```
  ctest --output-on-failure -R '^{{test_name}}$'
  ```

curl

Transfers data from or to a server.

Supports most protocols, including HTTP, FTP, and POP3.

More information: https://curl.se.

- Download the contents of an URL to a file:

```
curl {{http://example.com}} --output {{filename}}
```

- Download a file, saving the output under the filename indicated by the URL:

```
curl --remote-name {{http://example.com/filename}}
```

- Download a file, following location redirects, and automatically continuing (resuming) a previous file transfer:

```
curl --remote-name --location --continue-at - {{http://example.com/filename}}
```

- Send form-encoded data (POST request of `application/x-www-form-urlencoded`). Use `--data @file_name` or `--data @-` to read from STDIN:

```
curl --data {{'name=bob'}} {{http://example.com/form}}
```

- Send a request with an extra header, using a custom HTTP method:

```
curl --header {{'X-My-Header: 123'}} --request {{PUT}} {{http://example.com}}
```

- Send data in JSON format, specifying the appropriate content-type header:

```
curl --data {{'{"name": "bob"}'}} --header {{'Content-Type: application/json'}} {{http://example.com/users/1234}}
```

- Pass a username and password for server authentication:

```
curl --user myusernamе:mypassword {{http://example.com}}
```

- Pass client certificate and key for a resource, skipping certificate validation:

```
curl --cert {{client.pem}} --key {{key.pem}} --insecure {{https://example.com}}
```

cut

Cut out fields from stdin or files.

More information: https://www.gnu.org/software/coreutils/cut.

- Cut out the first sixteen characters of each line of stdin:

```
cut -c {{1-16}}
```

- Cut out the first sixteen characters of each line of the given files:

```
cut -c {{1-16}} {{file}}
```

- Cut out everything from the 3rd character to the end of each line:

```
cut -c {{3-}}
```

- Cut out the fifth field of each line, using a colon as a field delimiter (default delimiter is tab):

```
cut -d'{{:}}' -f{{5}}
```

- Cut out the 2nd and 10th fields of each line, using a semicolon as a delimiter:

```
cut -d'{{;}}' -f{{2,10}}
```

- Cut out the fields 3 through to the end of each line, using a space as a delimiter:

```
cut -d'{{ }}' -f{{3-}}
```

darkhttpd

Darkhttpd web server.

More information: https://unix4lyfe.org/darkhttpd

- Start server serving the specified document root:

  ```
  darkhttpd {{path/to/docroot}}
  ```

- Start server on specified port (port 80 by default if running as non-root user):

  ```
  darkhttpd {{path/to/docroot}} --port {{port}}
  ```

- Listen only on specified IP address (by default, the server listens on all interfaces):

  ```
  darkhttpd {{path/to/docroot}} --addr {{ip_address}}
  ```

dash

Debian Almquist Shell, a modern, POSIX-compliant implementation of sh (not Bash-compatible).

More information: https://manned.org/dash.

- Start an interactive shell session:

```
dash
```

- Execute a command and then exit:

```
dash -c "{{command}}"
```

- Execute a script:

```
dash {{path/to/script.sh}}
```

- Run commands from a script, printing each command before executing it:

```
dash -x {{path/to/script.sh}}
```

- Execute commands from a script, stopping at the first error:

```
dash -e {{path/to/script.sh}}
```

- Read and execute commands from stdin:

```
dash -s
```

date

Set or display the system date.

More information: https://www.gnu.org/software/coreutils/date.

- Display the current date using the default locale's format:

```
date +"%c"
```

- Display the current date in UTC and ISO 8601 format:

```
date -u +"%Y-%m-%dT%H:%M:%SZ"
```

- Display the current date as a Unix timestamp (seconds since the Unix epoch):

```
date +%s
```

- Display a specific date (represented as a Unix timestamp) using the default format:

```
date -d @1473305798
```

- Convert a specific date to the Unix timestamp format:

```
date -d "{{2018-09-01 00:00}}" +%s --utc
```

- Display the current date using the RFC-3339 format (`YYYY-MM-DD hh:mm:ss TZ`):

```
date --rfc-3339=s
```

dc

An arbitrary precision calculator. Uses reverse polish notation (RPN).

More information: https://www.gnu.org/software/bc/manual/dc-1.05/html_mono/dc.html.

- Run calculator in interactive mode:

```
dc
```

- Execute dc script in file:

```
dc -f {{file}}
```

- Calculate 4 times 5 [4 5 *], subtract 17 [17 -], and [p]rint the output (using ec

```
echo "4 5 * 17 - p" | dc
```

- Set number of decimal places to 7 [7 k], calculate 5 divided by -3 [5 _3 /] and [p] (using dc -e):

```
dc -e "7 k 5 _3 / p"
```

- Calculate the golden ratio, phi: Set number of decimal places to 100 [100 k], square root of 5 [5 v] plus 1 [1 +], divided by 2 [2 /], and [p]rint result:

```
dc -e "100 k 5 v 1 + 2 / p"
```

dcfl dd

Enhanced version of dd for forensics and security.

More information: http://dcfldd.sourceforge.net/

- Copy a disk to a raw image file and hash the image using SHA256:

```
dcfldd if=/dev/{{disk_device}} of={{file.img}} hash=sha256
hashlog={{file.hash}}
```

- Copy a disk to a raw image file, hashing each 1GB chunk:

```
dcfldd if=/dev/{{disk_device}} of={{file.img}} hash={{sha512|
sha384|sha256|sha1|md5}} hashlog={{file.hash}}
hashwindow={{1G}}
```

dcg

Drupal code generator.

More information: https://github.com/Chi-teck/drupal-code-generator.

- S tart a wiz ard to choose what kind of code (e.g. module, service, form, etc.) to generate:

  ```
  dcg
  ```

- Directly specify the kind of code to generate:

  ```
  dcg {{serv ice| plu g in| them e| m odu le| form }}
  ```

- G enerate the code in a specific directory:

  ```
  dcg - - directory {{path/to/directory}}
  ```

dcode

Recursively detect and decode strings, supporting hex, decimal, binary, base64, URL, FromChar encodings, Caesar ciphers, and MD5, SHA1, and SHA2 hashes.

Warning: uses 3rd-party web services for MD5, SHA1 and SHA2 hash lookups. For sensitive data, use `-s` to avoid these services.

More information: https://github.com/s0md3v/Decodify

- Recursively detect and decode a string:

  ```
  dcode "{{NjM3YTQyNzQ1YTQ0NGUzMg==}}"
  ```

- Rotate a string by the specified offset:

  ```
  dcode -rot {{11}} "{{spw wz hzcw o}}"
  ```

- Rotate a string by all 26 possible offsets:

  ```
  dcode -rot {{all}} "{{b pg k ta xh q tiitg iw pc sr}}"
  ```

- Reverse a string:

  ```
  dcode -rev "{{hello world}}"
  ```

dd

Convert and copy a file.

More information: https://www.gnu.org/software/coreutils/dd.

- Make a bootable usb drive from an isohybrid file (such like `archlinux-xxx.iso`) and show the progress:

```
dd if={{file.iso}} of=/dev/{{usb_drive}} status=progress
```

- Clone a drive to another drive with 4 MB block, ignore error and show progress:

```
dd if=/dev/{{source_drive}} of=/dev/{{dest_drive}} bs=4M
conv=noerror status=progress
```

- Generate a file of 100 random bytes by using kernel random driver:

```
dd if=/dev/urandom of={{random_file}} bs=100 count=1
```

- Benchmark the write performance of a disk:

```
dd if=/dev/zero of={{file_1GB}} bs=1024 count=1000000
```

- Generate a system backup into an IMG file and show the progress:

```
dd if=/dev/{{drive_device}} of={{path/to/file.img}}
status=progress
```

- Restore a drive from an IMG file and show the progress:

```
dd if={{path/to/file.img}} of=/dev/{{drive_device}}
status=progress
```

- Check progress of an ongoing dd operation (Run this command from another shell):

```
kill -USR1 $(pgrep ^dd)
```

decaffeinate

Move your CoffeeScript source to modern JavaScript.

More information: https://decaffeinate-project.org

- Convert a CoffeeScript file to JavaScript:

  ```
  decaffeinate {{path/to/file.coffee}}
  ```

- Convert a CoffeeScript v2 file to JavaScript:

  ```
  decaffeinate --use-cs2 {{path/to/file.coffee}}
  ```

- Convert require and module.exports to import and export:

  ```
  decaffeinate --use-js-modules {{path/to/file.coffee}}
  ```

- Convert a CoffeeScript, allowing named exports:

  ```
  decaffeinate --loose-js-modules {{path/to/file.coffee}}
  ```

deemix

A barebone deezer downloader library built from the ashes of Deezloader Remix.

It can be used as a standalone CLI app or implemented in a UI using the API.

More information: https://deemix.app.

- Download a track or playlist:

  ```
  deemix {{https://www.deezer.com/us/track/000000000}}
  ```

- Download track / playlist at a specific bitrate:

  ```
  deemix --bitrate {{FLAC|MP3}} {{url}}
  ```

- Download to a specific path:

  ```
  deemix --bitrate {{bitrate}} --path {{path}} {{url}}
  ```

- Create a portable deemix config in the current directory:

  ```
  deemix --portable --bitrate {{bitrate}} --path {{path}}
  {{url}}
  ```

delta

A viewer for G it and diff output.

More information: https://github.com/dandavison/delta.

- Compare files or directories:

  ```
  delta {{path/to/old_file_or_directory}} {{path/to/new _file_or_directory}}
  ```

- Compare files or directories, showing the line numbers:

  ```
  delta - - line- nu m b ers {{path/to/old_file_or_directory}} {{path/to/new _file_or_directory}}
  ```

- Compare files or directories, showing the differences side by side:

  ```
  delta - - side- b y- side {{path/to/old_file_or_directory}} {{path/to/new _file_or_directory}}
  ```

- Compare files or directories, ignoring any G it configuration settings:

  ```
  delta - - no- g itconfig  {{path/to/old_file_or_directory}} {{path/to/new _file_or_directory}}
  ```

- Compare, rendering commit hashes, file names, and line numbers as hyperlinks, according to the hyperlink spec for terminal emulators:

  ```
  delta - - hyperlink s {{path/to/old_file_or_directory}} {{path/to/new _file_or_directory}}
  ```

- Display the current settings:

  ```
  delta - - show - config
  ```

- Display supported languages and associated file extensions:

  ```
  delta - - list- lang u ag es
  ```

deluge-console

An interactive interface for the Deluge BitTorrent client.

More information: https://deluge-torrent.org.

- Start the interactive console interface:

  `deluge-console`

- Connect to a Deluge daemon instance:

  `connect {{hostname}}:{{port}}`

- Add a torrent to the daemon:

  `add {{url|magnet|path/to/file}}`

- Display information about all torrents:

  `info`

- Display information about a specific torrent:

  `info {{torrent_id}}`

- Pause a torrent:

  `pause {{torrent_id}}`

- Resume a torrent:

  `resume {{torrent_id}}`

- Remove a torrent from the daemon:

  `rm {{torrent_id}}`

deluge

A command-line BitTorrent client.

More information: https://deluge-torrent.org.

- Download a torrent:

```
deluge {{url|magnet|path/to/file}}
```

- Download a torrent using a specific configuration file:

```
deluge -c {{path/to/configuration_file}} {{url|magnet|path/to/file}}
```

- Download a torrent and launch the specified user interface:

```
deluge -u {{gtk|web|console}} {{url|magnet|path/to/file}}
```

- Download a torrent and output the log to a file:

```
deluge -l {{path/to/log_file}} {{url|magnet|path/to/file}}
```

deluged

A daemon process for the Deluge BitTorrent client.

More information: https://deluge-torrent.org.

- Start the Deluge daemon:

  ```
  deluged
  ```

- Start the Deluge daemon on a specific port:

  ```
  deluged -p {{port}}
  ```

- Start the Deluge daemon using a specific configuration file:

  ```
  deluged -c {{path/to/configuration_file}}
  ```

- Start the Deluge daemon and output the log to a file:

  ```
  deluged -l {{path/to/log_file}}
  ```

deno

A secure runtime for JavaScript and TypeScript.

More information: https://deno.land/.

- Run a JavaScript or TypeScript file:

  ```
  deno run {{path/to/file.ts}}
  ```

- Start a REPL (interactive shell):

  ```
  deno
  ```

- Run a file with network access enabled:

  ```
  deno run --allow-net {{path/to/file.ts}}
  ```

- Run a file from a URL:

  ```
  deno run {{https://deno.land/std/examples/welcome.ts}}
  ```

- Install an executable script from a URL:

  ```
  deno install {{https://deno.land/std/examples/colors.ts}}
  ```

dep

A CLI tool for deployment of PHP applications.

Note: The Go command `dep` with the same name is deprecated and archived.

More information: https://deployer.org.

- Interactively initialize deployer in the local path (use a framework template with `--template={{template}}`):

  ```
  dep init
  ```

- Deploy an application to a remote host:

  ```
  dep deploy {{hostname}}
  ```

- Rollback to the previous working release:

  ```
  dep rollback
  ```

- Connect to a remote host via ssh:

  ```
  dep ssh {{hostname}}
  ```

- List commands:

  ```
  dep list
  ```

- Run any arbitrary command on the remote hosts:

  ```
  dep run "{{command}}"
  ```

- Display help for a command:

  ```
  dep help {{command}}
  ```

detox

Renames files to make them easier to work with.

It removes spaces and other such annoyances like duplicate underline characters.

More information: https://github.com/dharple/detox.

- Remove spaces and other undesirable characters from a file's name:

  ```
  detox {{file}}
  ```

- Show how detox would rename all of the files in a directory tree:

  ```
  detox --dry-run -r {{directory}}
  ```

- Remove spaces and other undesirable characters from all files in a directory tree:

  ```
  detox -r {{directory}}
  ```

dexdump

Display information about Android DEX files.

More information: https://manned.org/dexdump.

- Extract classes and methods from an APK file:

  ```
  dexdump {{path/to/file.apk}}
  ```

- Display header information of DEX files contained in an APK file:

  ```
  dexdump -f {{path/to/file.apk}}
  ```

- Display the dis-assembled output of executable sections:

  ```
  dexdump -d {{path/to/file.apk}}
  ```

- Output results to a file:

  ```
  dexdump -o {{path/to/file}} {{path/to/file.apk}}
  ```

dexter

T ool for authenticating the kubectl users with O penId Connect.

More information: https://github.com/gini/dexter.

- Create and authenticate a user with G oogle O IDC:

  ```
  dexter au th - i {{client_id}} - s {{client_secret}}
  ```

- O verride the default kube config location:

  ```
  dexter au th - i {{client_id}} - s {{client_secret}} - - k u b e-
  config  {{sam ple/config }}
  ```

df

G ives an overview of the filesystem disk space usage.

More information: https://www.gnu.org/software/coreutils/df.

- Display all filesystems and their disk usage:

  ```
  df
  ```

- Display all filesystems and their disk usage in human readable form:

  ```
  df - h
  ```

- Display the filesystem and its disk usage containing the given file or directory:

  ```
  df {{path/to/file_or_directory}}
  ```

- Display statistics on the number of free inodes:

  ```
  df - i
  ```

- Display filesystems but exclude the specified types:

  ```
  df - x {{sq u ashfs}} - x {{tm pfs}}
  ```

dfc

Gives an overview of the filesystem disk space usage with colours and graphs.

More information: https://projects.gw-computing.net/projects/dfc/wiki

- Display filesystems and their disk usage in human readable form with colours and graphs:

  ```
  dfc
  ```

- Display all filesystems including pseudo, duplicate and inaccessible filesystems:

  ```
  dfc -a
  ```

- Display filesystems without colour:

  ```
  dfc -c never
  ```

- Display filesystems containing "ext" in the filesystem type:

  ```
  dfc -t ext
  ```

dhclient

DHCP client.

More information: https://manned.org/dhclient.

- Get an IP address for the `eth0` interface:

  ```
  sudo dhclient {{eth0}}
  ```

- Release an IP address for the `eth0` interface:

  ```
  sudo dhclient -r {{eth0}}
  ```

dhcpwn

Test DHCP IP exhaustion attacks and sniff local DHCP traffic.

More information: https://github.com/mschwager/dhcpwn.

- Flood the network with IP requests:

```
dhcpwn --interface {{network_interface}} flood --count {{number_of_requests}}
```

- Sniff local DHCP traffic:

```
dhcpwn --interface {{network_interface}} sniff
```

dict

Command line dictionary using the DICT protocol.

More information: https://github.com/cheusov/dictd.

- List available databases:

  ```
  dict -D
  ```

- Get information about a database:

  ```
  dict -i {{database_name}}
  ```

- Look up a word in a specific database:

  ```
  dict -d {{database_name}} {{word}}
  ```

- Look up a word in all available databases:

  ```
  dict {{word}}
  ```

- Show information about the DICT server:

  ```
  dict -I
  ```

diff

Compare files and directories.

More information: https://man7.org/linux/man-pages/man1/diff.1.html

- Compare files (lists changes to turn `old_file` into `new_file`):

  `diff {{old_file}} {{new_file}}`

- Compare files, ignoring white spaces:

  `diff -w {{old_file}} {{new_file}}`

- Compare files, showing the differences side by side:

  `diff -y {{old_file}} {{new_file}}`

- Compare files, showing the differences in unified format (as used by `git diff`):

  `diff -u {{old_file}} {{new_file}}`

- Compare directories recursively (shows names for differing files/directories as well as changes made to files):

  `diff -r {{old_directory}} {{new_directory}}`

- Compare directories, only showing the names of files that differ:

  `diff -rq {{old_directory}} {{new_directory}}`

diffoscope

Compare files, archives, and directories.

More information: https://diffoscope.org.

- Compare two files:

```
diffoscope {{path/to/file1 }} {{path/to/file2 }}
```

- Compare two files without displaying a progress bar:

```
diffoscope - - no- prog ress {{path/to/file1 }} {{path/to/file2 }}
```

- Compare two files and write a H T ML-report to a file (use - for stdout):

```
diffoscope - - htm l {{path/to/ou tfile| - }} {{path/to/file1 }}
{{path/to/file2 }}
```

- Compare two directories excluding files with a name matching a specified pattern:

```
diffoscope - - exclu de {{pattern}} {{path/to/directory1 }}
{{path/to/directory2 }}
```

- Compare two directories and control whether directory metadata is considered:

```
diffoscope - - exclu de- directory- m etadata {{au to| yes| no|
recu rsiv e}} {{path/to/directory1 }} {{path/to/directory2 }}
```

diffstat

Create a histogram from the output of the `diff` command.

More information: https://manned.org/diffstat.

- Display changes in a histogram:

```
diff {{file1}} {{file2}} | diffstat
```

- Display inserted, deleted and modified changes as a table:

```
diff {{file1}} {{file2}} | diffstat -t
```

dig

DNS Lookup utility.

More information: https://manpages.debian.org/dnsutils/dig.1.html

- Lookup the IP(s) associated with a hostname (A records):

```
dig +short {{example.com}}
```

- Get a detailed answer for a given domain (A records):

```
dig +noall +answer {{example.com}}
```

- Query a specific DNS record type associated with a given domain name:

```
dig +short {{example.com}} {{A|MX|TXT|CNAME|NS}}
```

- Get all types of records for a given domain name:

```
dig {{example.com}} ANY
```

- Specify an alternate DNS server to query:

```
dig @{{8.8.8.8}} {{example.com}}
```

- Perform a reverse DNS lookup on an IP address (PTR record):

```
dig -x {{8.8.8.8}}
```

- Find authoritative name servers for the zone and display SOA records:

```
dig +nssearch {{example.com}}
```

- Perform iterative queries and display the entire trace path to resolve a domain name:

```
dig +trace {{example.com}}
```

dillo

A lightweight web browser intended for slow computers.

More information: https://www.dillo.org/.

- Launch Dillo:

```
dillo
```

- Launch Dillo with a specific window size and screen location:

```
dillo --geometry {{width}}x{{height}}+{{x_position}}+{{y_position}}
```

- Launch Dillo and open a specific URL:

```
dillo {{duckduckgo.com}}
```

- Launch Dillo and open a file or directory:

```
dillo {{path/to/file_or_directory}}
```

- Launch Dillo in full-screen mode:

```
dillo --fullwindow
```

- Display version:

```
dillo --version
```

- Display help:

```
dillo --help
```

dircolors

Output commands to set the LS_COLOR environment variable and style `ls`, `dir`, etc.

More information: https://www.gnu.org/software/coreutils/dircolors.

- Output commands to set LS_COLOR using default colors:

  `dircolors`

- Output commands to set LS_COLOR using colors from a file:

  `dircolors {{file}}`

- Output commands for Bourne shell:

  `dircolors --bourne-shell`

- Output commands for C shell:

  `dircolors --c-shell`

- View the default colors for file types and extensions:

  `dircolors --print-data`

direnv

Shell extension to load and unload environment variables depending on the current directory.

More information: https://github.com/direnv/direnv.

- Grant direnv permission to load the specified `.envrc`:

  `direnv allow`

- Revoke the authorization of a given `.envrc`:

  `direnv deny`

- Edit the `.envrc` file in the default text editor and reload the environment on exit:

  `direnv edit .`

- Trigger a reload of the environment:

  `direnv reload`

- Print some debug status information:

  `direnv status`

dirname

Calculates the parent directory of a given file or directory path.

More information: https://www.gnu.org/software/coreutils/dirname.

- Calculate the parent directory of a given path:

  `dirnam e {{path/to/file_or_directory}}`

- Calculate the parent directory of multiple paths:

  `dirnam e {{path/to/file_a}} {{path/to/directory_b }}`

- Delimit output with a N U L character instead of a newline (useful when combining with `xarg s`):

  `dirnam e - - zero {{path/to/directory_a}} {{path/to/file_b }}`

dirs

Displays or manipulates the directory stack.

The directory stack is a list of recently visited directories that can be manipulated with the **`pushd`** and **`popd`** commands.

More information: https://www.gnu.org/software/bash/manual/bash.html#Directory-Stack-Builtins

- Display the directory stack with a space between each entry:

```
dirs
```

- Display the directory stack with one entry per line:

```
dirs -p
```

- Display only the nth entry in the directory stack, starting at 0:

```
dirs +{{N}}
```

- Clear the directory stack:

```
dirs -c
```

dirsearch

Web path scanner.

More information: https://github.com/maurosoria/dirsearch.

- Scan a web server for common paths with common extensions:

```
dirsearch --url {{url}} --extensions-list
```

- Scan a list of web servers for common paths with the `.php` extension:

```
dirsearch --urls-list {{path/to/urls-list.txt}} --extensions {{php}}
```

- Scan a web server for user-defined paths with common extensions:

```
dirsearch --url {{url}} --extensions-list --wordlist {{path/to/urls-paths.txt}}
```

- Scan a web server using a cookie:

```
dirsearch --url {{url}} --extensions {{php}} --cookie {{cookie}}
```

- Scan a web server using the `HEAD` HTTP method:

```
dirsearch --url {{url}} --extensions {{php}} --http-method {{HEAD}}
```

- Scan a web server, saving the results to a `.json` file:

```
dirsearch --url {{url}} --extensions {{php}} --json-report {{path/to/report.json}}
```

diskonaut

Terminal disk space navigator, written in Rust.

More information: https://github.com/imsnif/diskonaut.

- Start diskonaut in the current directory:

  ```
  diskonaut
  ```

- Start diskonaut in a specific directory:

  ```
  diskonaut {{path/to/directory}}
  ```

- Show file sizes rather than their block usage on the disk:

  ```
  diskonaut --apparent-size {{path/to/directory}}
  ```

- Disable deletion confirmation:

  ```
  diskonaut --disable-delete-confirmation
  ```

distccd

Server daemon for the distcc distributed compiler.

More information: https://distcc.github.io.

- Start a daemon with the default settings:

```
distccd --daemon
```

- Start a daemon, accepting connections from IPv4 private network ranges:

```
distccd --daemon --allow-private
```

- Start a daemon, accepting connections from a specific network address or address range:

```
distccd --daemon --allow {{ip_address|network_prefix}}
```

- Start a daemon with a lowered priority that can run a maximum of 4 tasks at a time:

```
distccd --daemon --jobs {{4}} --nice {{5}}
```

- Start a daemon and register it via mDNS/DNS-SD (Zeroconf):

```
distccd --daemon --zeroconf
```

dive

A tool for exploring a Docker image, layer contents, and discovering ways to shrink it.

More information: https://github.com/wagoodman/dive.

- Analyz e a Docker image:

  ```
  div e {{you r_im ag e_tag }}
  ```

- B uild an image and start analyz ing it:

  ```
  div e b u ild - t {{som e_tag }}
  ```

dj ango-admin

Dj ango' s utility for administrative tasks.

More information: https://docs.dj angoproj ect.com/en/3 .0 /ref/dj ango-admin/

- Create a new dj ango proj ect:

  ```
  dj ang o- adm in startproj ect {{proj ect_nam e}}
  ```

- Create a new app for the current proj ect:

  ```
  dj ang o- adm in startapp {{app_nam e}}
  ```

- Check the current version of Dj ango:

  ```
  dj ang o- adm in - - v ersion
  ```

- Display more information for the given command:

  ```
  dj ang o- adm in help {{com m and}}
  ```

doas

Executes a command as another user.

More information: https://man.openbsd.org/doas.

- Run a command as root:

  ```
  doas {{command}}
  ```

- Run a command as another user:

  ```
  doas -u {{user}} {{command}}
  ```

- Launch the default shell as root:

  ```
  doas -s
  ```

- Parse a config file and check if the execution of a command as another user is allowed:

  ```
  doas -C {{config_file}} {{command}}
  ```

- Make `doas` request a password even after it was supplied earlier:

  ```
  doas -L
  ```

docker build

Build an image from a Dockerfile.

More information: https://docs.docker.com/engine/reference/commandline/build/.

- Build a docker image using the Dockerfile in the current directory:

```
docker build .
```

- Build a docker image from a Dockerfile at a specified URL:

```
docker build {{github.com/creack/docker-firefox}}
```

- Build a docker image and tag it:

```
docker build --tag {{name:tag}} .
```

- Build a docker image with no build context:

```
docker build --tag {{name:tag}} - < {{Dockerfile}}
```

- Do not use the cache when building the image:

```
docker build --no-cache --tag {{name:tag}} .
```

- Build a docker image using a specific Dockerfile:

```
docker build --file {{Dockerfile}} .
```

- Build with custom build-time variables:

```
docker build --build-arg {{HTTP_PROXY=http://10.20.30.2:1234}} --build-arg {{FTP_PROXY=http://40.50.60.5:4567}} .
```

docker-compose

Run and manage multi container docker applications.

More information: https://docs.docker.com/compose/reference/overview/.

- List all running containers:

  ```
  docker-compose ps
  ```

- Create and start all containers in the background using a `docker-compose.yml` file from the current directory:

  ```
  docker-compose up -d
  ```

- Start all containers, rebuild if necessary:

  ```
  docker-compose up --build
  ```

- Start all containers using an alternate compose file:

  ```
  docker-compose --file {{path/to/file}} up
  ```

- Stop all running containers:

  ```
  docker-compose stop
  ```

- Stop and remove all containers, networks, images, and volumes:

  ```
  docker-compose down --rmi all --volumes
  ```

- Follow logs for all containers:

  ```
  docker-compose logs --follow
  ```

- Follow logs for a specific container:

  ```
  docker-compose logs --follow {{container_name}}
  ```

docker container

Manage Docker containers.

More information: https://docs.docker.com/engine/reference/commandline/container/.

- List currently running Docker containers:

  ```
  docker container ls
  ```

- Start one or more stopped containers:

  ```
  docker container start {{container1_name}} {{container2_name}}
  ```

- Kill one or more running containers:

  ```
  docker container kill {{container_name}}
  ```

- Stop one or more running containers:

  ```
  docker container stop {{container_name}}
  ```

- Pause all processes within one or more containers:

  ```
  docker container pause {{container_name}}
  ```

- Display detailed information on one or more containers:

  ```
  docker container inspect {{container_name}}
  ```

- Export a container's filesystem as a tar archive:

  ```
  docker container export {{container_name}}
  ```

- Create a new image from a container's changes:

  ```
  docker container commit {{container_name}}
  ```

docker cp

Copy files or directories between host and container filesystems.

More information: https://docs.docker.com/engine/reference/commandline/cp.

- Copy a file or directory from the host to a container:

```
dock er cp {{path/to/file_or_directory_on_host}}
{{container_nam e}}: {{path/to/file_or_directory_in_container}}
```

- Copy a file or directory from a container to the host:

```
dock er cp {{container_nam e}}: {{path/to/
file_or_directory_in_container}} {{path/to/
file_or_directory_on_host}}
```

- Copy a file or directory from the host to a container, following symlinks (copies the symlinked files directly, not the symlinks themselves):

```
dock er cp - - follow - link  {{path/to/sym link _on_host}}
{{container_nam e}}: {{path/to/file_or_directory_in_container}}
```

docker exec

Execute a command on an already running Docker container.

More information: https://docs.docker.com/engine/reference/commandline/exec/.

- Enter an interactive shell session on an already-running container:

```
docker exec --interactive --tty {{container_name}} {{/bin/bash}}
```

- Run a command in the background (detached) on a running container:

```
docker exec --detach {{container_name}} {{command}}
```

- Select the working directory for a given command to execute into:

```
docker exec --interactive -tty --workdir {{path/to/directory}} {{container_name}} {{command}}
```

- Run a command in background on existing container but keep stdin open:

```
docker exec --interactive --detach {{container_name}} {{command}}
```

- Set an environment variable in a running bash session:

```
docker exec --interactive --tty --env {{variable_name}}={{value}} {{container_name}} {{/bin/bash}}
```

- Run a command as a specific user:

```
docker exec --user {{user}} {{container_name}} {{command}}
```

docker image

Manage Docker images.

See also `docker build`, `docker import`, and `docker pull`.

More information: https://docs.docker.com/engine/reference/commandline/image/.

- List local Docker images:

  ```
  docker image ls
  ```

- Delete unused local Docker images:

  ```
  docker image prune
  ```

- Delete all unused images (not just those without a tag):

  ```
  docker image prune --all
  ```

- Show the history of a local Docker image:

  ```
  docker image history {{image}}
  ```

docker images

Manage Docker images.

More information: https://docs.docker.com/engine/reference/commandline/images/.

- List all Docker images:

```
dock er im ag es
```

- List all Docker images including intermediates:

```
dock er im ag es - - all
```

- List the output in q uiet mode (only numeric IDs):

```
dock er im ag es - - q u iet
```

- List all Docker images not used by any container:

```
dock er im ag es - - filter dang ling = tru e
```

- List images that contain a substring in their name:

```
dock er im ag es " {{* nam e* }}"
```

docker inspect

Return low-level information on Docker objects.

More information: https://docs.docker.com/engine/reference/commandline/inspect/.

- Show help:

```
docker inspect
```

- Display information about a container, image, or volume using a name or ID:

```
docker inspect {{container|image|ID}}
```

- Display a container's IP address:

```
docker inspect --format='{{range .NetworkSettings.Networks}
{{.IPAddress}}{{end}}' {{container}}
```

- Display the path to the container's log file:

```
docker inspect --format='{{.LogPath}}' {{container}}
```

- Display the image name of the container:

```
docker inspect --format='{{.Config.Image}}' {{container}}
```

- Display the configuration information as JSON:

```
docker inspect --format='{{json .Config}}' {{container}}
```

- Display all port bindings:

```
docker inspect --format='{{range $p, $conf
:= .NetworkSettings.Ports}} {{$p}} -> {{(index $conf
0).HostPort}} {{end}}' {{container}}
```

docker login

Log into a docker registry.

More information: https://docs.docker.com/engine/reference/commandline/login/.

- Interactively log into a registry:

```
docker login
```

- Log into a registry with a specific username (user will be prompted for a password):

```
docker login --username {{username}}
```

- Log into a registry with username and password:

```
docker login --username {{username}} --password {{password}} {{server}}
```

- Log into a registry with password from stdin:

```
echo "{{password}}" | docker login --username {{username}} --password-stdin
```

docker logs

Print container logs.

More information: https://docs.docker.com/engine/reference/commandline/logs.

- Print logs from a container:

```
docker logs {{container_name}}
```

- Print logs and follow them:

```
docker logs -f {{container_name}}
```

- Print last 5 lines:

```
docker logs {{container_name}} --tail {{5}}
```

- Print logs and append them with timestamps:

```
docker logs -t {{container_name}}
```

- Print logs from a certain point in time of container execution (i.e. 23m, 10s, 2013-01-02T13:23:37):

```
docker logs {{container_name}} --until {{time}}
```

docker-machine

Create and manage machines running Docker.

More information: https://docs.docker.com/machine/reference/.

- List currently running docker machines:

  ```
  dock er- m achine ls
  ```

- Create a new docker machine with specific name:

  ```
  dock er- m achine create {{nam e}}
  ```

- G et the status of a machine:

  ```
  dock er- m achine statu s {{nam e}}
  ```

- S tart a machine:

  ```
  dock er- m achine start {{nam e}}
  ```

- S top a machine:

  ```
  dock er- m achine stop {{nam e}}
  ```

- Inspect information about a machine:

  ```
  dock er- m achine inspect {{nam e}}
  ```

docker network

Create and manage docker networks.

More information: https://docs.docker.com/engine/reference/commandline/network/.

- List all available and configured networks on docker daemon:

  ```
  docker network ls
  ```

- Create a user defined network:

  ```
  docker network create --driver {{driver_name}} {{network_name}}
  ```

- Display detailed information of a space-separated list of networks:

  ```
  docker network inspect {{network_name}}
  ```

- Connect a container to a network using a name or ID:

  ```
  docker network connect {{network_name}} {{container_name|ID}}
  ```

- Disconnect a container from a network:

  ```
  docker network disconnect {{network_name}} {{container_name|ID}}
  ```

- Remove all unused (not referenced by any container) networks:

  ```
  docker network prune
  ```

- Remove a space-separated list of unused networks:

  ```
  docker network rm {{network_name}}
  ```

docker ps

List Docker containers.

More information: https://docs.docker.com/engine/reference/commandline/ps/.

- List currently running docker containers:

  ```
  docker ps
  ```

- List all docker containers (running and stopped):

  ```
  docker ps --all
  ```

- Show the latest created container (includes all states):

  ```
  docker ps --latest
  ```

- Filter containers that contain a substring in their name:

  ```
  docker ps --filter="name={{name}}"
  ```

- Filter containers that share a given image as an ancestor:

  ```
  docker ps --filter "ancestor={{image}}:{{tag}}"
  ```

- Filter containers by exit status code:

  ```
  docker ps --all --filter="exited={{code}}"
  ```

- Filter containers by status (created, running, removing, paused, exited and dead):

  ```
  docker ps --filter="status={{status}}"
  ```

- Filter containers that mount a specific volume or have a volume mounted in a specific path:

  ```
  docker ps --filter="volume={{path/to/directory}}" --format "table {{.ID}}\t{{.Image}}\t{{.Names}}\t{{.Mounts}}"
  ```

docker rmi

Remove one or more Docker images.

More information: https://docs.docker.com/engine/reference/commandline/rmi/.

- Show help:

  ```
  docker rmi
  ```

- Remove one or more images given their names:

  ```
  docker rmi {{image1 image2 ...}}
  ```

- Force remove an image:

  ```
  docker rmi --force {{image}}
  ```

- Remove an image without deleting untagged parents:

  ```
  docker rmi --no-prune {{image}}
  ```

docker run

Run a command in a new Docker container.

More information: https://docs.docker.com/engine/reference/commandline/run/.

- Run command in a new container from a tagged image:

```
docker run {{image:tag}} {{command}}
```

- Run command in a new container in background and display its ID:

```
docker run -d {{image}} {{command}}
```

- Run command in a one-off container in interactive mode and pseudo-TTY:

```
docker run --rm -it {{image}} {{command}}
```

- Run command in a new container with passed environment variables:

```
docker run -e '{{variable}}={{value}}' -e {{variable}} {{image}} {{command}}
```

- Run command in a new container with bind mounted volumes:

```
docker run -v {{path/to/host_path}}:{{path/to/container_path}} {{image}} {{command}}
```

- Run command in a new container with published ports:

```
docker run -p {{host_port}}:{{container_port}} {{image}} {{command}}
```

docker save

Export one or more docker images to archive.

More information: https://docs.docker.com/engine/reference/commandline/save/.

- Save an image by redirecting stdout to a tar archive:

```
docker save {{image}}:{{tag}} > {{path/to/file.tar}}
```

- Save an image to a tar archive:

```
docker save --output {{path/to/file.tar}} {{image}}:{{tag}}
```

- Save all tags of the image:

```
docker save --output {{path/to/file.tar}} {{image_name}}
```

- Cherry-pick particular tags of an image to save:

```
docker save --output {{path/to/file.tar}} {{image_name:tag1 image_name:tag2 ...}}
```

docker secret

Manage Docker swarm secrets.

More information: https://docs.docker.com/engine/reference/commandline/secret/.

- Create a new secret from stdin:

```
{{com m and}} |  dock er secret create {{secret_nam e}} -
```

- Create a new secret from a file:

```
dock er secret create {{secret_nam e}} {{path/to/file}}
```

- List all secrets:

```
dock er secret ls
```

- Display detailed information on one or multiple secrets in a human friendly format:

```
dock er secret inspect - - pretty {{secret_nam e1  secret_nam e2
. . . }}
```

- R emove one or more secrets:

```
dock er secret rm  {{secret_nam e1  secret_nam e2  . . . }}
```

docker service

Manage the services on a docker daemon.

More information: https://docs.docker.com/engine/reference/commandline/service/.

- List the services on a docker daemon:

  ```
  dock er serv ice ls
  ```

- Create a new service:

  ```
  dock er serv ice create - - nam e {{serv ice_nam e}} {{im ag e}}:
  {{tag }}
  ```

- Display detailed information of a space-separated list of services:

  ```
  dock er serv ice inspect {{serv ice_nam e| I D }}
  ```

- List the tasks of a space-separated list of services:

  ```
  dock er serv ice ps {{serv ice_nam e| I D }}
  ```

- S cale to a specific number of replicas for a space-separated list of services:

  ```
  dock er serv ice scale {{serv ice_nam e}}= {{cou nt_of_replicas}}
  ```

- R emove a space-separated list of services:

  ```
  dock er serv ice rm  {{serv ice_nam e| I D }}
  ```

docker start

Start one or more stopped containers.

More information: https://docs.docker.com/engine/reference/commandline/start/.

- Show help:

  ```
  docker start
  ```

- Start a docker container:

  ```
  docker start {{container}}
  ```

- Start a container, attaching stdout and stderr and forwarding signals:

  ```
  docker start --attach {{container}}
  ```

- Start one or more space-separated containers:

  ```
  docker start {{container(s)}}
  ```

docker stats

Display a live stream of resource usage statistics for containers.

More information: https://docs.docker.com/engine/reference/commandline/stats/.

- Display a live stream for the statistics of all running containers:

  ```
  dock er stats
  ```

- Display a live stream of statistics for a space-separated list of containers:

  ```
  dock er stats {{container_nam e}}
  ```

- Change the columns format to display container' s cpu usage percentage:

  ```
  dock er stats - - form at " {{. N am e}}: \ t{{. C P U P erc}}"
  ```

- Display statistics for all containers (both running and stopped):

  ```
  dock er stats - - all
  ```

- Disable streaming stats and only pull the current stats:

  ```
  dock er stats - - no- stream
  ```

docker swarm

A container orchestration tool.

More information: https://docs.docker.com/engine/swarm/.

- Initialize a swarm cluster:

  ```
  docker swarm init
  ```

- Display the token to join a manager or a worker:

  ```
  docker swarm join-token {{worker|manager}}
  ```

- Join a new node to the cluster:

  ```
  docker swarm join --token {{token}} {{manager_node_url:2377
  ```

- Remove a worker from the swarm (run inside the worker node):

  ```
  docker swarm leave
  ```

- Display the current CA certificate in PEM format:

  ```
  docker swarm ca
  ```

- Rotate the current CA certificate and display the new certificate:

  ```
  docker swarm ca --rotate
  ```

- Change the valid period for node certificates:

  ```
  docker swarm update --cert-expiry {{hours}}h{{minutes}}
  m{{seconds}}s
  ```

docker system

Manage Docker data and display system-wide information.

More information: https://docs.docker.com/engine/reference/commandline/system/.

- Show help:

```
docker system
```

- Show docker disk usage:

```
docker system df
```

- Show detailed information on disk usage:

```
docker system df --verbose
```

- Remove unused data:

```
docker system prune
```

- Remove unused data created more than a specified amount of time in the past:

```
docker system prune --filter="until={{hours}}h{{minutes}}m"
```

- Display real-time events from the Docker daemon:

```
docker system events
```

- Display real-time events from containers streamed as valid JSON Lines:

```
docker system events --filter 'type=container' --format
'{{json .}}'
```

- Display system-wide information:

```
docker system info
```

docker

Manage Docker containers and images.

More information: https://docs.docker.com/engine/reference/commandline/cli/.

- List currently running docker containers:

  ```
  docker ps
  ```

- List all docker containers (running and stopped):

  ```
  docker ps -a
  ```

- Start a container from an image, with a custom name:

  ```
  docker run --name {{container_name}} {{image}}
  ```

- Start or stop an existing container:

  ```
  docker {{start|stop}} {{container_name}}
  ```

- Pull an image from a docker registry:

  ```
  docker pull {{image}}
  ```

- Open a shell inside of an already running container:

  ```
  docker exec -it {{container_name}} {{sh}}
  ```

- Remove a stopped container:

  ```
  docker rm {{container_name}}
  ```

- Fetch and follow the logs of a container:

  ```
  docker logs -f {{container_name}}
  ```

doctum

A PHP API documentation generator.

More information: https://github.com/code-lts/doctum.

- Parse a project:

  ```
  doctum parse
  ```

- Render a project:

  ```
  doctum render
  ```

- Parse then render a project:

  ```
  doctum update
  ```

- Parse and render only a specific version of a project:

  ```
  doctum update --only-version={{version}}
  ```

- Parse and render a project using a specific configuration:

  ```
  doctum update {{path/to/config.php}}
  ```

dokku

Docker powered mini-Heroku (PaaS).

Easily deploy multiple apps to your server in different languages using a single `git push` command.

More information: https://github.com/dokku/dokku.

- List running apps:

  `dokku apps`

- Create an app:

  `dokku apps:create {{app_name}}`

- Remove an app:

  `dokku apps:destroy {{app_name}}`

- Install plugin:

  `dokku plugin:install {{full_repo_url}}`

- Link database to an app:

  `dokku {{db}}:link {{db_name}} {{app_name}}`

dolt add

Add the contents of a table to the list of Dolt staged tables.

More information: https://github.com/dolthub/dolt.

- Add a table to the list of staged tables (stage a table):

  ```
  dolt add {{tab le}}
  ```

- S tage all tables:

  ```
  dolt add - - all
  ```

dolt blame

Displays commit information for each row of a Dolt table.

More information: http://github.com/dolthub/dolt.

- Display the latest commit for each row of a table:

  ```
  dolt b lam e {{tab le}}
  ```

- Display the latest commits for each row of a table when the specified commit was made:

  ```
  dolt b lam e {{com m it}} {{tab le}}
  ```

- V iew help:

  ```
  dolt b lam e - - help
  ```

dolt branch

Manage Dolt branches.

More information: https://github.com/dolthub/dolt.

- List local branches (current branch is highlighted by *):

  ```
  dolt b ranch
  ```

- List all local and remote branches:

  ```
  dolt b ranch - - all
  ```

- Create a new branch based on the current branch:

  ```
  dolt b ranch {{b ranch_nam e}}
  ```

- Create a new branch with the specified commit as the latest:

  ```
  dolt b ranch {{b ranch_nam e}} {{com m it}}
  ```

- R ename a branch:

  ```
  dolt b ranch - - m ov e {{b ranch_nam e1 }} {{b ranch_nam e2 }}
  ```

- Duplicate a branch:

  ```
  dolt b ranch - - copy {{b ranch_nam e1 }} {{b ranch_nam e2 }}
  ```

- Delete a branch:

  ```
  dolt b ranch - - delete {{b ranch_nam e}}
  ```

- Display the name of the current branch:

  ```
  dolt b ranch - - show - cu rrent
  ```

dolt checkout

Checkout the work tree or tables to a specific branch or commit.

More information: https://github.com/dolthub/dolt.

- Switch to a branch:

  ```
  dolt checkout {{branch_name}}
  ```

- Revert unstaged changes to a table:

  ```
  dolt checkout {{table}}
  ```

- Create new branch and switch to it:

  ```
  dolt checkout -b {{branch_name}}
  ```

- Create new branch based on a specified commit and switch to it:

  ```
  dolt checkout -b {{branch_name}} {{commit}}
  ```

dolt commit

Commit staged changes to tables.

More information: https://docs.dolthub.com/interfaces/cli#_dolt-commit

- Commit all staged changes, opening the editor specified by `$EDITOR` to enter the commit message:

  ```
  dolt commit
  ```

- Commit all staged changes with the specified message:

  ```
  dolt commit --message "{{commit_message}}"
  ```

- Stage all unstaged changes to tables before committing:

  ```
  dolt commit --all
  ```

- Use the specified ISO 8601 commit date (defaults to current date and time):

  ```
  dolt commit --date "{{2021-12-31T00:00:00}}"
  ```

- Use the specified author for the commit:

  ```
  dolt commit --author "{{author_name}} <{{author_email}}>"
  ```

- Allow creating an empty commit, with no changes:

  ```
  dolt commit --allow-empty
  ```

- Ignore foreign key warnings:

  ```
  dolt commit --force
  ```

dolt config

Read and write local (per repository) and global (per user) Dolt configuration variables.

More information: https://docs.dolthub.com/interfaces/cli#dolt-config

- List all local and global configuration options and their values:

  ```
  dolt config --list
  ```

- Display the value of a local or global configuration variable:

  ```
  dolt config --get {{name}}
  ```

- Modify the value of a local configuration variable, creating it if does not exist:

  ```
  dolt config --add {{name}} {{value}}
  ```

- Modify the value of a global configuration variable, creating it if does not exist:

  ```
  dolt config --global --add {{name}} {{value}}
  ```

- Delete a local configuration variable:

  ```
  dolt config --unset {{name}}
  ```

- Delete a global configuration variable:

  ```
  dolt config --global --unset {{name}}
  ```

dolt

Dolt is a SQL database that you can fork, clone, branch, merge, push and pull just like a Git repository.

More information: https://github.com/dolthub/dolt.

- Execute a dolt subcommand:

  ```
  dolt {{subcommand}}
  ```

- List available subcommands:

  ```
  dolt help
  ```

dot

Render an image of a **linear directed** network graph from a **graphviz** file.

Layouts: **dot**, **neato**, **twopi**, **circo**, **fdp**, **sfdp**, **osage** & **patchwork**

More information: https://graphviz.org/doc/info/command.html

- Render a `png` image with a filename based on the input filename and output format (uppercase -O):

  ```
  dot -T {{png}} -O {{path/to/input.gv}}
  ```

- Render a `svg` image with the specified output filename (lowercase -o):

  ```
  dot -T {{svg}} -o {{path/to/image.svg}} {{path/to/input.gv}}
  ```

- Render the output in `ps`, `pdf`, `svgz`, `fig`, `png`, `gif`, `jpg`, `json`, or `dot` format:

  ```
  dot -T {{format}} -O {{path/to/input.gv}}
  ```

- Render a `gif` image using stdin and stdout:

  ```
  echo "{{digraph {this -> that}}}" | dot -T {{gif}} > {{path/to/image.gif}}
  ```

- Display help:

  ```
  dot -?
  ```

dotnet build

Builds a .NET application and its dependencies.

More information: https://docs.microsoft.com/dotnet/core/tools/dotnet-build.

- Compile the project or solution in the current directory:

  ```
  dotnet build
  ```

- Compile a .NET project or solution in debug mode:

  ```
  dotnet build {{path/to/project_or_solution}}
  ```

- Compile in release mode:

  ```
  dotnet build --configuration {{Release}}
  ```

- Compile without restoring dependencies:

  ```
  dotnet build --no-restore
  ```

- Compile with a specific verbosity level:

  ```
  dotnet build --verbosity {{quiet|minimal|normal|detailed|diagnostic}}
  ```

- Compile for a specific runtime:

  ```
  dotnet build --runtime {{runtime_identifier}}
  ```

- Specify the output directory:

  ```
  dotnet build --output {{path/to/directory}}
  ```

dotnet publish

Publish a .NET application and its dependencies to a directory for deployment to a hosting system.

More information: https://docs.microsoft.com/dotnet/core/tools/dotnet-publish.

- Compile a .NET project in release mode:

  ```
  dotnet publish --configuration Release {{path/to/project_file}}
  ```

- Publish the .NET Core runtime with your application for the specified runtime:

  ```
  dotnet publish --self-contained true --runtime {{runtime_identifier}} {{path/to/project_file}}
  ```

- Package the application into a platform-specific single-file executable:

  ```
  dotnet publish --runtime {{runtime_identifier}} -p:PublishSingleFile=true {{path/to/project_file}}
  ```

- Trim unused libraries to reduce the deployment size of an application:

  ```
  dotnet publish --self-contained true --runtime {{runtime_identifier}} -p:PublishTrimmed=true {{path/to/project_file}}
  ```

- Compile a .NET project without restoring dependencies:

  ```
  dotnet publish --no-restore {{path/to/project_file}}
  ```

- Specify the output directory:

  ```
  dotnet publish --output {{path/to/directory}} {{path/to/project_file}}
  ```

dotnet restore

Restores the dependencies and tools of a .NET project.

More information: https://docs.microsoft.com/dotnet/core/tools/dotnet-restore.

- Restore dependencies for a .NET project or solution in the current directory:

  ```
  dotnet restore
  ```

- Restore dependencies for a .NET project or solution in a specific location:

  ```
  dotnet restore {{path/to/project_or_solution}}
  ```

- Restore dependencies without caching the HTTP requests:

  ```
  dotnet restore --no-cache
  ```

- Force all dependencies to be resolved even if the last restore was successful:

  ```
  dotnet restore --force
  ```

- Restore dependencies using package source failures as warnings:

  ```
  dotnet restore --ignore-failed-sources
  ```

- Restore dependencies with a specific verbosity level:

  ```
  dotnet restore --verbosity {{quiet|minimal|normal|detailed|diagnostic}}
  ```

dotnet

Cross platform .NET command-line tools for .NET Core.

More information: https://docs.microsoft.com/dotnet/core/tools.

- Initialize a new .NET project:

  ```
  dotnet new {{template_short_name}}
  ```

- Restore nuget packages:

  ```
  dotnet restore
  ```

- Build and execute the .NET project in the current directory:

  ```
  dotnet run
  ```

- Run a packaged dotnet application (only needs the runtime, the rest of the commands require the .NET Core SDK installed):

  ```
  dotnet {{path/to/application.dll}}
  ```

doxygen

A documentation system for various programming languages.

More information: http://www.doxygen.nl.

- G enerate a default template configuration file`D oxyfile`

  `doxyg en - g`

- G enerate a template configuration file:

  `doxyg en - g  {{path/to/config _file}}`

- G enerate documentation using an existing configuration file:

  `doxyg en {{path/to/config _file}}`

drill

Perform various DNS queries.

More information: https://manned.org/drill.

- Lookup the IP(s) associated with a hostname (A records):

  ```
  drill {{example.com}}
  ```

- Lookup the mail server(s) associated with a given domain name (MX record):

  ```
  drill mx {{example.com}}
  ```

- Get all types of records for a given domain name:

  ```
  drill any {{example.com}}
  ```

- Specify an alternate DNS server to query:

  ```
  drill {{example.com}} @{{8.8.8.8}}
  ```

- Perform a reverse DNS lookup on an IP address (PTR record):

  ```
  drill -x {{8.8.8.8}}
  ```

- Perform DNSSEC trace from root servers down to a domain name:

  ```
  drill -TD {{example.com}}
  ```

- Show DNSKEY record(s) for a domain name:

  ```
  drill -s dnskey {{example.com}}
  ```

drupal-check

Check Drupal PHP code for deprecations.

More information: https://github.com/mglaman/drupal-check.

- Check the code in a specific directory for deprecations:

  ```
  drupal-check {{path/to/directory}}
  ```

- Check the code excluding a comma-separated list of directories:

  ```
  drupal-check --exclude-dir {{path/to/excluded_directory}},
  {{path/to/excluded_files/*.php}} {{path/to/directory}}
  ```

- Don't show a progress bar:

  ```
  drupal-check --no-progress {{path/to/directory}}
  ```

- Perform static analysis to detect bad coding practices:

  ```
  drupal-check --analysis {{path/to/directory}}
  ```

drupal

CLI for Drupal.

G enerates boilerplate code, interacts with and debugs Drupal proj ects.

More information: https://drupalconsole.com/.

- Install a module:

  `dru pal m odu le: install {{m odu le_nam e}}`

- U ninstall a module:

  `dru pal m odu le: u ninstall {{m odu le_nam e}}`

- Clear all caches:

  `dru pal cache: reb u ild`

- V iew current Drupal installation status:

  `dru pal site: statu s`

drush

A command-line shell and scripting interface for Drupal.

More information: https://www.drush.org.

- Enable module "foo":

  ```
  drush en {{foo}}
  ```

- Uninstall module "foo":

  ```
  drush pm-uninstall {{foo}}
  ```

- Clear all caches:

  ```
  drush cr
  ```

- Clear CSS and JavaScript caches:

  ```
  drush cc css-js
  ```

du

Disk usage: estimate and summarize file and directory space usage.

More information: https://www.gnu.org/software/coreutils/du.

- List the sizes of a directory and any subdirectories, in the given unit (B/KB/MB):

```
du -{{b|k|m}} {{path/to/directory}}
```

- List the sizes of a directory and any subdirectories, in human-readable form (i.e. auto-selecting the appropriate unit for each size):

```
du -h {{path/to/directory}}
```

- Show the size of a single directory, in human readable units:

```
du -sh {{path/to/directory}}
```

- List the human-readable sizes of a directory and of all the files and directories within it:

```
du -ah {{path/to/directory}}
```

- List the human-readable sizes of a directory and any subdirectories, up to N levels deep:

```
du -h --max-depth={{N}} {{path/to/directory}}
```

- List the human-readable size of all `.jpg` files in subdirectories of the current directory, and show a cumulative total at the end:

```
du -ch */*.jpg
```

dua

Dua (Disk Usage Analyzer) is a tool to conveniently learn about the usage of disk space of a given directory.

More information: https://github.com/Byron/dua-cli

- Analyze specific directory:

  ```
  dua {{path/to/directory}}
  ```

- Display apparent size instead of disk usage:

  ```
  dua --apparent-size
  ```

- Count hard-linked files each time they are seen:

  ```
  dua --count-hard-links
  ```

- Aggregate the consumed space of one or more directories or files:

  ```
  dua aggregate
  ```

- Launch the terminal user interface:

  ```
  dua interactive
  ```

- Format printing byte counts:

  ```
  dua --format {{metric|binary|bytes|GB|GiB|MB|MiB}}
  ```

- Set the number of threads to be used:

  ```
  dua --threads {{count}}
  ```

duplicity

Creates incremental, compressed, encrypted and versioned backups.

Can also upload the backups to a variety of backend services.

More information: http://duplicity.nongnu.org.

- Backup a directory via FTPS to a remote machine, encrypting it with a password:

```
FTP_PASSWORD={{ftp_login_password}}
PASSPHRASE={{encryption_password}} duplicity {{path/to/
source/directory}} {{ftps://user@hostname/target/directory/
path/}}
```

- Backup a directory to Amazon S3, doing a full backup every month:

```
duplicity --full-if-older-than {{1M}} --use-new-style s3://
{{bucket_name[/prefix]}}
```

- Delete versions older than 1 year from a backup stored on a WebDAV share:

```
FTP_PASSWORD={{webdav_login_password}} duplicity remove-
older-than {{1Y}} --force {{webdav[s]://user@hostname[:port
some_dir}}
```

- List the available backups:

```
duplicity collection-status "file://{{absolute/path/to/
backup/directory}}"
```

- List the files in a backup stored on a remote machine, via ssh:

```
duplicity list-current-files --time {{YYYY-MM-DD}} scp://
{{user@hostname}}/path/to/backup/dir
```

- Restore a subdirectory from a GnuPG-encrypted local backup to a given location:

```
PASSPHRASE={{gpg_key_password}} duplicity restore --encryp
key {{gpg_key_id}} --file-to-restore {{relative/path/
restoredirectory}} file://{{absolute/path/to/backup/
directory}} {{path/to/directory/to/restore/to}}
```

dust

Dust gives an instant overview of which directories are using disk space.

More information: https://github.com/bootandy/dust.

- Display information for the current directory:

```
du st
```

- Display information for a space-separated list of directories:

```
du st {{path/to/directory1 }} {{path/to/directory2 }}
```

- Display 3 0 directories (defaults to 2 1):

```
du st - - nu m b er- of- lines {{3 0 }}
```

- Display information for the current directory, up to 3 levels deep:

```
du st - - depth {{3 }}
```

- Display the biggest directories at the top in descending order:

```
du st - - rev erse
```

- Ignore all files and directories with a specific name:

```
du st - - ig nore- directory {{file_or_directory_nam e}}
```

- Do not display percent bars and percentages:

```
du st - - no- percent- b ars
```

dvc add

Add changed files to the index.

More information: https://dvc.org/doc/command-reference/add.

- Add a single target file to the index:

  ```
  dvc add {{path/to/file}}
  ```

- Add a target directory to the index:

  ```
  dvc add {{path/to/directory}}
  ```

- Recursively add all the files in a given target directory:

  ```
  dvc add --recursive {{path/to/directory}}
  ```

- Add a target file with a custom `.dvc` filename:

  ```
  dvc add --file {{custom_name.dvc}} {{path/to/file}}
  ```

dvc checkout

Checkout data files and directories from cache.

More information: https://dvc.org/doc/command-reference/checkout.

- Checkout the latest version of all target files and directories:

  ```
  dv c check ou t
  ```

- Checkout to latest version of a specified target:

  ```
  dv c check ou t {{targ et}}
  ```

- Checkout a specific version of a target from a different G it commit/tag/branch:

  ```
  g it check ou t {{com m it_hash| tag | b ranch}} {{targ et}} & &  dv c
  check ou t {{targ et}}
  ```

dvc commit

Record changes to DVC-tracked files in the project.

More information: https://dvc.org/doc/command-reference/commit.

- Commit changes to all DVC-tracked files and directories:

  ```
  dvc commit
  ```

- Commit changes to a specified DVC-tracked target:

  ```
  dvc commit {{target}}
  ```

- Recursively commit all DVC-tracked files in a directory:

  ```
  dvc commit --recursive {{path/to/directory}}
  ```

dvc config

Low level command to manage custom configuration options for dvc repositories.

T hese configurations can be on proj ect, local, global, or system level.

More information: https://dvc.org/doc/command-reference/config.

- G et the name of the default remote:

  ```
  dv c config core. rem ote
  ```

- S et the proj ect' s default remote:

  ```
  dv c config core. rem ote {{rem ote_nam e}}
  ```

- U nset the proj ect' s default remote:

  ```
  dv c config - - u nset core. rem ote
  ```

- G et the config value for a specified key for the current proj ect:

  ```
  dv c config {{k ey}}
  ```

- S et the config value for a key on a proj ect level:

  ```
  dv c config {{k ey}} {{v alu e}}
  ```

- U nset a proj ect level config value for a given key:

  ```
  dv c config - - u nset {{k ey}}
  ```

- S et a local, global, or system level config value:

  ```
  dv c config - - local/g lob al/system {{k ey}} {{v alu e}}
  ```

dvc dag

Visualize the pipeline(s) defined in `dvc.yaml`

More information: https://dvc.org/doc/command-reference/dag.

- Visualize the entire pipeline:

  ```
  dvc dag
  ```

- Visualize the pipeline stages up to a specified target stage:

  ```
  dvc dag {{target}}
  ```

- Export the pipeline in the dot format:

  ```
  dvc dag --dot > {{path/to/pipeline.dot}}
  ```

dvc destroy

Remove all DVC files and directories from a DVC project.

More information: https://dvc.org/doc/command-reference/destroy.

- Destroy the current project:

  ```
  dvc destroy
  ```

- Force destroy the current project:

  ```
  dvc destroy --force
  ```

dvc diff

S how changes in DV C tracked file and directories.

More information: https://dvc.org/doc/command-reference/diff.

- Compare DV C tracked files from different G it commits, tags, and branches w.r.t the current workspace:

  ```
  dv c diff {{com m it_hash/tag /b ranch}}
  ```

- Compare the changes in DV C tracked files from 1 G it commit to another:

  ```
  dv c diff {{rev ision_b }} {{rev ision_a}}
  ```

- Compare DV C tracked files, along with their latest hash:

  ```
  dv c diff - - show - hash {{com m it}}
  ```

- Compare DV C tracked files, displaying the output as J S O N :

  ```
  dv c diff - - show - j son - - show - hash {{com m it}}
  ```

- Compare DV C tracked files, displaying the output as Markdown:

  ```
  dv c diff - - show - m d - - show - hash {{com m it}}
  ```

dvc fetch

Download DV C tracked files and directories from a remote repository.

More information: https://dvc.org/doc/command-reference/fetch.

- F etch the latest changes from the default remote upstream repository (if set):

  ```
  dv c fetch
  ```

- F etch changes from a specific remote upstream repository:

  ```
  dv c fetch - - rem ote {{rem ote_nam e}}
  ```

- F etch the latest changes for a specific target/s:

  ```
  dv c fetch {{targ et/s}}
  ```

- F etch changes for all branch and tags:

  ```
  dv c fetch - - all- b ranches - - all- tag s
  ```

- F etch changes for all commits:

  ```
  dv c fetch - - all- com m its
  ```

dvc freeze

Freeze stages in the DVC pipeline.

This prevents DVC from tracking changes in stage dependencies and re-execution until unfreeze.

See also `dvc unfreeze`.

More information: https://dvc.org/doc/command-reference/freeze

- Freeze 1 or more specified stages:

```
dvc freeze {{stage_name_a}} [{{stage_name_b}} ...]
```

dvc gc

Remove unused files and directories from the cache or remote storage.

More information: https://dvc.org/doc/command-reference/gc.

- Garbage collect from the cache, keeping only versions referenced by the current workspace:

  ```
  dvc gc --workspace
  ```

- Garbage collect from the cache, keeping only versions referenced by branch, tags, and commits:

  ```
  dvc gc --all-branches --all-tags --all-commits
  ```

- Garbage collect from the cache, including the default cloud remote storage (if set):

  ```
  dvc gc --all-commits --cloud
  ```

- Garbage collect from the cache, including a specific cloud remote storage:

  ```
  dvc gc --all-commits --cloud --remote {{remote_name}}
  ```

dvc init

Initializ e a new local DV C repository.

More information: https://dvc.org/doc/command-reference/init.

- Initializ e a new local repository:

  ```
  dv c init
  ```

- Initializ e DV C without G it:

  ```
  dv c init - - no- scm
  ```

- Initializ e DV C in a subdirectory:

  ```
  cd {{path/to/su b dir}} & &  dv c init - - su dir
  ```

dvc unfreeze

Unfreeze stages in the DVC pipeline.

This allows DVC to start tracking changes in stage dependencies again after they were frozen.

See also **`dvc freeze`**.

More information: https://dvc.org/doc/command-reference/unfreeze

- Unfreeze 1 or more specified stages:

```
dvc unfreeze {{stage_name_a}} [{{stage_name_b}} ...]
```

dvc

Data V ersion Control: like `g i t` for data.

More information: https://dvc.org/.

- Check the DV C version:

  ```
  dv c - - v ersion
  ```

- Display general help:

  ```
  dv c - - help
  ```

- Display help about a specific subcommand:

  ```
  dv c {{su b com m and}} - - help
  ```

- E xecute a DV C subcommand:

  ```
  dv c {{su b com m and}}
  ```

ebook-convert

Can be used to convert e-books between common formats, e.g., pdf, epub and mobi.

P art of the Calibre e-book library tool.

More information: https://manual.calibre-ebook.com/generated/en/ebook-convert.html.

- Convert an e-book into another format:

  ```
  eb ook - conv ert {{sou rce}} {{destination}}
  ```

- Convert Markdown or H T ML to e-book with T O C, title and author:

  ```
  eb ook - conv ert {{sou rce}} {{destination}} - - lev el1 - toc= " //
  h: h1 "  - - lev el2 - toc= " //h: h2 "  - - lev el3 - toc= " //h: h3 "  - -
  title= {{title}} - - au thors= {{au thor}}
  ```

echo

Print given arguments.

More information: https://www.gnu.org/software/coreutils/echo.

- Print a text message. Note: quotes are optional:

```
echo "{{Hello World}}"
```

- Print a message with environment variables:

```
echo "{{My path is $PATH}}"
```

- Print a message without the trailing newline:

```
echo -n "{{Hello World}}"
```

- Append a message to the file:

```
echo "{{Hello World}}" >> {{file.txt}}
```

- Enable interpretation of backslash escapes (special characters):

```
echo -e "{{Column 1\tColumn 2}}"
```

ect

Efficient Compression Tool.

File optimizer written in C++. It supports `.png`, `.jpg`, `.gzip` and `.zip` files.

More information: https://github.com/fhanau/Efficient-Compression-Tool

- Compress a file:

  ```
  ect {{path/to/file.png}}
  ```

- Compress a file with specified compression level and multithreading (1 = Fastest (Worst), 9 = Slowest (Best), default is 3):

  ```
  ect -{{9}} --mt-deflate {{path/to/file.zip}}
  ```

- Compress all files in a directory recursively:

  ```
  ect -recurse {{path/to/directory}}
  ```

- Compress a file, keeping the original modification time:

  ```
  ect -keep {{path/to/file.png}}
  ```

- Compress a file, stripping metadata:

  ```
  ect -strip {{path/to/file.png}}
  ```

ed

The original Unix text editor.

More information: https://man.archlinux.org/man/ed.1.

- Start ed, editing an empty document (which can be saved as a new file in the current directory):

```
ed
```

- Start ed, editing an empty document, with `:` as a command prompt indicator:

```
ed -p :
```

- Start ed editing an existing file (this shows the byte count of the loaded file):

```
ed -p : {{path/to/file}}
```

- Toggle the printing of error explanations. (By default, explanations are not printed and only a `?` appears):

```
H
```

- Add text to the current document. Mark completion by entering a period by itself in a new line:

```
a<Enter> {{text_to_insert}}<Enter> .
```

- Print the entire document (`,` is a shortcut to the range `1,$` which covers the start to the end of the document):

```
,p
```

- Write the current document to a new file (the filename can be omitted if `ed` was called with an existing file):

```
w {{filename}}
```

- Quit ed:

```
q
```

electron-packager

A tool used to build Electron app executables for Windows, Linux and macOS.

Requires a valid package.json in the application directory.

More information: https://github.com/electron/electron-packager.

- Package an application for the current architecture and platform:

  ```
  electron-packager "{{path/to/app}}" "{{app_name}}"
  ```

- Package an application for all architectures and platforms:

  ```
  electron-packager "{{path/to/app}}" "{{app_name}}" --all
  ```

- Package an application for 64-bit Linux:

  ```
  electron-packager "{{path/to/app}}" "{{app_name}}" --
  platform="{{linux}}" --arch="{{x64}}"
  ```

- Package an application for ARM macOS:

  ```
  electron-packager "{{path/to/app}}" "{{app_name}}" --
  platform="{{darwin}}" --arch="{{arm64}}"
  ```

electrum

Ergonomic Bitcoin wallet and private key management.

More information: https://electrum.org.

- Create a new wallet:

```
electrum -w {{new_wallet.dat}} create
```

- Restore an existing wallet from seed offline:

```
electrum -w {{recovery_wallet.dat}} restore -o
```

- Create a signed transaction offline:

```
electrum mktx {{recipient}} {{amount}} -f 0.00000001 -F
{{from}} -o
```

- Display all wallet receiving addresses:

```
electrum listaddresses -a
```

- Sign a message:

```
electrum signmessage {{address}} {{message}}
```

- Verify a message:

```
electrum verifymessage {{address}} {{signature}} {{message}}
```

- Connect only to a specific electrum-server instance:

```
electrum -p socks5:{{127.0.0.1}}:9050 -s
{{56ckl5obj37gypcu.onion}}:50001:t -1
```

elinks

A text based browser similar to lynx.

More information: http://elinks.or.cz.

- S tart elinks:

  ```
  elink s
  ```

- Q uit elinks:

  ```
  C trl + C
  ```

- Dump output of webpage to console, coloriz ing the text with AN S I control codes:

  ```
  elink s - du m p - du m p- color- m ode {{1 }} {{u rl}}
  ```

elixir

Elixir programming language interpreter.

More information: https://elixir-lang.org.

- Run an Elixir file:

  ```
  elixir {{path/to/file}}
  ```

- Evaluate Elixir code by passing it as an argument:

  ```
  elixir -e "{{code}}"
  ```

elm

Compile and run Elm source files.

More information: https://elm-lang.org.

- Initialize an Elm project, generates an elm.json file:

  `elm init`

- Start interactive Elm shell:

  `elm repl`

- Compile an Elm file, output the result to an `index.html` file:

  `elm make {{source}}`

- Compile an Elm file, output the result to a JavaScript file:

  `elm make {{source}} --output={{destination}}.js`

- Start local web server that compiles Elm files on page load:

  `elm reactor`

- Install Elm package from https://package.elm-lang.org:

  `elm install {{author}}/{{package}}`

emacs

The extensible, customizable, self-documenting, real-time display editor.

See also `emacsclient`.

More information: https://www.gnu.org/software/emacs.

- Start Emacs and open a file:

  ```
  emacs {{path/to/file}}
  ```

- Open a file at a specified line number:

  ```
  emacs +{{line_number}} {{path/to/file}}
  ```

- Start Emacs in console mode (without an X window):

  ```
  emacs --no-window-system
  ```

- Start an Emacs server in the background (accessible via `emacsclient`):

  ```
  emacs --daemon
  ```

- Stop a running Emacs server and all its instances, asking for confirmation on unsaved files:

  ```
  emacsclient --eval '(save-buffers-kill-emacs)'
  ```

- Save a file in Emacs:

  ```
  Ctrl+X, Ctrl+S
  ```

- Quit Emacs:

  ```
  Ctrl+X, Ctrl+C
  ```

emacsclient

Open files in an existing Emacs server.

See also `emacs`

More information: https://www.emacswiki.org/emacs/EmacsClient

- Open a file in an existing Emacs server (using GUI if available):

  ```
  emacsclient {{path/to/file}}
  ```

- Open a file in console mode (without an X window):

  ```
  emacsclient --no-window-system {{path/to/file}}
  ```

- Open a file in a new Emacs window:

  ```
  emacsclient --create-frame {{path/to/file}}
  ```

- Evaluate a command, printing the output to stdout, and then quit:

  ```
  emacsclient --eval '({{command}})'
  ```

- Specify an alternative editor in case no Emacs server is running:

  ```
  emacsclient --alternate-editor {{editor}} {{path/to/file}}
  ```

- Stop a running Emacs server and all its instances, asking for confirmation on unsaved files:

  ```
  emacsclient --eval '(save-buffers-kill-emacs)'
  ```

ember

The Ember.js command-line utility.

Used for creating and maintaining Ember.js applications.

More information: https://cli.emberjs.com

- Create a new Ember application:

  `ember new {{my-new-app}}`

- Create a new Ember addon:

  `ember addon {{my-new-addon}}`

- Build the project:

  `ember build`

- Build the project in production mode:

  `ember build --prod`

- Run the development server:

  `ember serve`

- Run the test suite:

  `ember test`

- Run a blueprint to generate something like a route or component:

  `ember generate {{type}} {{name}}`

- Install an ember-cli addon:

  `ember install {{name-of-addon}}`

emulator

Manager Android emulators from the command-line.

More information: https://developer.android.com/studio/run/emulator-commandline.

- Display the help:

  ```
  em u lator - help
  ```

- S tart an Android emulator device:

  ```
  em u lator - av d {{nam e}}
  ```

- Display the webcams on your development computer that are available for emulation:

  ```
  em u lator - av d {{nam e}} - w eb cam - list
  ```

- S tart an emulator overriding the facing back camera setting (use `- cam era- front` for front camera):

  ```
  em u lator - av d {{nam e}} - cam era- b ack  {{none| em u lated| w eb cam N
  ```

- S tart an emulator, with a maximum network speed:

  ```
  em u lator - av d {{nam e}} - netspeed {{g sm | hscsd| g prs| edg e| hsdpa
  lte| ev do| fu ll}}
  ```

- S tart an emulator with network latency:

  ```
  em u lator - av d {{nam e}} - netdelay {{g sm | hscsd| g prs| edg e| hsdpa
  lte| ev do| none}}
  ```

- S tart an emulator, making all T CP connections through a specified H T T P /H T T P S proxy (port number is req uired):

  ```
  em u lator - av d {{nam e}} - http- proxy {{http: //exam ple. com : 8 0 }}
  ```

- S tart an emulator with a given S D card partition image file:

  ```
  em u lator - av d {{nam e}} - sdcard {{path/to/sdcard. im g }}
  ```

enca

Detect and convert the encoding of text files.

More information: https://github.com/nij_el/enca

- Detect file(s) encoding according to the system's locale:

```
enca {{file1 file2 ...}}
```

- Detect file(s) encoding specifying a language in the POSIX/C locale format (e.g. zh_CN, en_US):

```
enca -L {{language}} {{file1 file2 ...}}
```

- Convert file(s) to a specific encoding:

```
enca -L {{language}} -x {{to_encoding}} {{file1 file2 ...}}
```

- Create a copy of an existing file using a different encoding:

```
enca -L {{language}} -x {{to_encoding}} < {{original_file}} > {{new_file}}
```

encfs

Mounts or creates encrypted virtual filesystems.

S ee also `f u s ermou`, `m t`, which can unmount filesystems mounted by this command.

More information: https://github.com/vgough/encfs.

- Initiaiz e or mount an encrypted filesystem:

  ```
  encfs {{/path/to/cipher_dir}} {{/path/to/m ou nt_point}}
  ```

- Initiaiz e an encrypted filesystem with standard settings:

  ```
  encfs - - standard {{/path/to/cipher_dir}} {{/path/to/
  m ou nt_point}}
  ```

- R un encfs in the foreground instead of spawning a daemon:

  ```
  encfs - f {{/path/to/cipher_dir}} {{/path/to/m ou nt_point}}
  ```

- Mount an encrypted snapshot of a plain directory:

  ```
  encfs - - rev erse {{path/to/plain_dir}} {{path/to/cipher_dir}}
  ```

enscript

Convert text files to PostScript, HTML, RTF, ANSI, and overstrikes.

More information: https://www.gnu.org/software/enscript.

- Generate a PostScript file from a text file:

```
enscript {{path/to/input_file}} --output={{path/to/output_file}}
```

- Generate a file in a different language than PostScript:

```
enscript {{path/to/input_file}} --language={{html|rtf|...}} --output={{path/to/output_file}}
```

- Generate a PostScript file with a landscape layout, splitting the page into columns (maximum 9):

```
enscript {{path/to/input_file}} --columns={{num}} --landscape --output={{path/to/output_file}}
```

- Display available syntax highlighting languages and file formats:

```
enscript --help-highlight
```

- Generate a PostScript file with syntax highlighting and color for a specified language:

```
enscript {{path/to/input_file}} --color=1 --highlight={{language}} --output={{path/to/output_file}}
```

entr

Run arbitrary commands when files change.

More information: https://manned.org/entr.

- Rebuild with `make` if any file in any subdirectory changes:

  ```
  {{ag -l}} | entr {{make}}
  ```

- Rebuild and test with `make` if any `.c` source files in the current directory change:

  ```
  {{ls *.c}} | entr {{'make && make test'}}
  ```

- Send a `SIGTERM` to any previously spawned ruby subprocesses before executing `ruby main.rb`:

  ```
  {{ls *.rb}} | entr -r {{ruby main.rb}}
  ```

- Run a command with the changed file (`/_`) as an argument:

  ```
  {{ls *.sql}} | entr {{psql -f}} /_
  ```

env

Show the environment or run a program in a modified environment.

More information: https://www.gnu.org/software/coreutils/env.

- Show the environment:

```
env
```

- Run a program. Often used in scripts after the shebang (`#!`) for looking up the path to the program:

```
env {{program}}
```

- Clear the environment and run a program:

```
env -i {{program}}
```

- Remove variable from the environment and run a program:

```
env -u {{variable}} {{program}}
```

- Set a variable and run a program:

```
env {{variable}={{value}} {{program}}
```

- Set multiple variables and run a program:

```
env {{variable1}={{value}} {{variable2}={{value}} {{variable3}={{value}} {{program}}
```

envoy

A P H P -based task manager for Laravel remote servers.

More information: https://laravel.com/docs/envoy.

- Initialise a configuration file:

  ```
  env oy init {{host_nam e}}
  ```

- R un a task:

  ```
  env oy ru n {{task _nam e}}
  ```

- R un a task from a specific proj ect:

  ```
  env oy ru n - - path {{path/to/directory}} {{task _nam e}}
  ```

- R un a task and continue on failure:

  ```
  env oy ru n - - continu e {{task _nam e}}
  ```

- Dump a task as a bash script for inspection:

  ```
  env oy ru n - - pretend {{task _nam e}}
  ```

- Connect to the specified server via S S H :

  ```
  env oy ssh {{serv er_nam e}}
  ```

envsubst

Substitutes environment variables with their value in shell format strings.

Variables to be replaced should be in either `${var}` or `$var` format.

More information: https://www.gnu.org/software/gettext/manual/html_node/envsubst-Invocation.html.

- Replace environment variables in stdin and output to stdout:

  ```
  echo '{{$HOME}}' | envsubst
  ```

- Replace environment variables in an input file and output to stdout:

  ```
  envsubst < {{path/to/input_file}}
  ```

- Replace environment variables in an input file and output to a file:

  ```
  envsubst < {{path/to/input_file}} > {{path/to/output_file}}
  ```

- Replace environment variables in an input file from a space-separated list:

  ```
  envsubst '{{$USER $SHELL $HOME}}' < {{path/to/input_fil
  ```

erl

Run and manage programs in the Erlang programming language.

More information: https://www.erlang.org.

- Compile and run sequential Erlang program as a common script and then exit:

```
erlc {{files}} && erl -noshell
'{{module:myfunction(arguments)}}, init:stop().'
```

- Connect to a running Erlang node:

```
erl -remsh {{nodename}}@{{hostname}} -sname
{{customshortname}} -hidden -setcookie
{{cookieofremotenode}}
```

- Tell the Erlang shell to load modules from a directory:

```
erl -pa {{directory_with_beam_files}}
```

esbuild

JavaScript bundler and minifier built for speed.

More information: https://esbuild.github.io/.

- Bundle a JavaScript application and print to stdout:

```
esbuild --bundle {{path/to/file.js}}
```

- Bundle a JSX application from stdin:

```
esbuild --bundle --outfile={{path/to/out.js}} < {{path/to/file.jsx}}
```

- Bundle and minify a JSX application with source maps in production mode:

```
esbuild --bundle --define:{{process.env.NODE_ENV=\"production\"}} --minify --sourcemap {{path/to/file.js}}
```

- Bundle a JSX application for a comma-separated list of browsers:

```
esbuild --bundle --minify --sourcemap --target={{chrome58,firefox57,safari11,edge16}} {{path/to/file.jsx}}
```

- Bundle a JavaScript application for a specific node version:

```
esbuild --bundle --platform={{node}} --target={{node12}} {{path/to/file.js}}
```

- Bundle a JavaScript application enabling JSX syntax in .js files:

```
esbuild --bundle app.js --loader:{{.js=jsx}} {{path/to/file.js}}
```

- Bundle and serve a JavaScript application on an HTTP server:

```
esbuild --bundle --serve={{port}} --outfile={{index.js}} {{path/to/file.js}}
```

- Bundle a list of files to an output directory:

```
esbuild --bundle --outdir={{path/to/output_directory}} {{path/to/file1}} {{path/to/file2}}
```

eslint

A pluggable linting utility for JavaScript and JSX.

More information: https://eslint.org.

- Create eslint config:

  ```
  eslint --init
  ```

- Lint on a given set of files:

  ```
  eslint {{filename}}.js {{filename1}}.js
  ```

- Fix lint issues:

  ```
  eslint --fix
  ```

- Lint with config:

  ```
  eslint -c {{path/to/config_file}} {{app/src}}
  ```

espanso

Cross-platform Text Expander written in Rust.

More information: https://espanso.org.

- Check status:

  ```
  espanso status
  ```

- Edit the configuration:

  ```
  espanso edit config
  ```

- Install a package from the hub store (https://hub.espanso.org/):

  ```
  espanso install {{package name}}
  ```

- Restart (required after installing a package, useful in case of failure):

  ```
  espanso restart
  ```

espeak

Uses text-to-speech to speak through the default sound device.

More information: http://espeak.sourceforge.net.

- Speak a phrase aloud:

```
espeak "I like to ride my bike."
```

- Speak a file aloud:

```
espeak -f {{filename}}
```

- Save output to a WAV audio file, rather than speaking it directly:

```
espeak -w {{filename.wav}} "It's GNU plus Linux"
```

- Use a different voice:

```
espeak -v {{voice}}
```

etcd

A distributed, reliable key-value store for the most critical data of a distributed system.

More information: https://etcd.io.

- Start a single-node etcd cluster:

  ```
  etcd
  ```

- Start a single-node etcd cluster, listening for client requests on a custom URL:

  ```
  etcd --advertise-client-urls {{http://127.0.0.1:1234}} \
  listen-client-urls {{http://127.0.0.1:1234}}
  ```

- Start a single-node etcd cluster with a custom name:

  ```
  etcd --name {{my_etcd_cluster}}
  ```

- Start a single-node etcd cluster with extensive metrics available at http://localhost:2379/debug/pprof/:

  ```
  etcd --enable-pprof --metrics extensive
  ```

etcdctl

CLI interface for interacting with etcd, a highly-available key-value pair store.

More information: https://etcd.io/docs/latest/dev-guide/interacting_v3/

- Display the value associated with a specified key:

  ```
  etcdctl get {{mykey}}
  ```

- Store a key-value pair:

  ```
  etcdctl put {{mykey}} {{myvalue}}
  ```

- Delete a key-value pair:

  ```
  etcdctl del {{mykey}}
  ```

- Store a key-value pair, reading the value from a file:

  ```
  etcdctl put {{mykey}} < {{path/to/file.txt}}
  ```

- Save a snapshot of the etcd keystore:

  ```
  etcdctl snapshot save {{path/to/snapshot.db}}
  ```

- Restore a snapshot of an etcd keystore (restart the etcd server afterwards):

  ```
  etcdctl snapshot restore {{path/to/snapshot.db}}
  ```

- Add a user:

  ```
  etcdctl user add {{myuser}}
  ```

- Watch a key for changes:

  ```
  etcdctl watch {{mykey}}
  ```

eva

Simple calculator REPL, similar to `bc`, with syntax highlighting and persistent history.

More information: https://github.com/NerdyPepper/eva

- Run the calculator in interactive mode:

```
eva
```

- Calculate the result of an expression:

```
eva "{{(1 + 2) * 2 ^ 2}}"
```

- Calculate an expression forcing the number of decimal places to 5:

```
eva --fix {{5}} "{{5 / 3}}"
```

- Calculate an expression with sine and cosine:

```
eva "{{sin(1) + cos(1)}}"
```

evil-winrm

Windows Remote Management (WinRM) shell for pentesting.

Once connected, we get a PowerShell prompt on the target host.

More information: https://github.com/Hackplayers/evil-winrm

- Connect to a host:

```
evil-winrm --ip {{ip}} --user {{user}} --password {{password}}
```

- Connect to a host, passing the password hash:

```
evil-winrm --ip {{ip}} --user {{user}} --hash {{nt_hash}}
```

- Connect to a host, specifying directories for scripts and executables:

```
evil-winrm --ip {{ip}} --user {{user}} --password {{password}} --scripts {{path/to/scripts}} --executables {{path/to/executables}}
```

- Connect to a host, using SSL:

```
evil-winrm --ip {{ip}} --user {{user}} --password {{password}} --ssl --pub-key {{path/to/pubkey}} --priv-key {{path/to/privkey}}
```

- Upload a file to the host:

```
PS > upload {{path/to/local/file}} {{path/to/remote/file}}
```

- Get a list of loaded PowerShell functions:

```
PS > menu
```

- Load a PowerShell script from the `-scripts` directory:

```
PS > {{script.ps1}}
```

- Invoke a binary on the host from the `--executables` directory:

```
PS > Invoke-Binary {{binary.exe}}
```

exa

A modern replacement for **`ls`** (List directory contents).

More information: https://the.exa.website.

- List files one per line:

  ```
  exa --oneline
  ```

- List all files, including hidden files:

  ```
  exa --all
  ```

- Long format list (permissions, ownership, size and modification date) of all files:

  ```
  exa --long --all
  ```

- List files with the largest at the top:

  ```
  exa --reverse --sort={{size}}
  ```

- Display a tree of files, three levels deep:

  ```
  exa --long --tree --level={{3}}
  ```

- List files sorted by modification date (oldest first):

  ```
  exa --long --sort={{modified}}
  ```

- List files with their headers, icons, and Git statuses:

  ```
  exa --long --header --icons --git
  ```

- Don't list files mentioned in `.gitignore`:

  ```
  exa --git-ignore
  ```

exec

Replace the current process with another process.

- Replace with the specified command using the current environment variables:

  ```
  exec {{command and-with-flags}}
  ```

- Replace with the specified command, clearing environment variables:

  ```
  exec -c {{command and-with-flags}}
  ```

- Replace with the specified command and login using the default shell:

  ```
  exec -l {{command and-with-flags}}
  ```

- Replace with the specified command and change the process name:

  ```
  exec -a {{process_name}} {{command and-with-flags}}
  ```

exenv

A tool to easily install Elixir versions and manage application environments.

More information: https://github.com/exenv/exenv.

- Display a list of installed versions:

  ```
  exenv versions
  ```

- Use a specific version of Elixir across the whole system:

  ```
  exenv global {{version}}
  ```

- Use a specific version of Elixir for the current application/project directory:

  ```
  exenv local {{version}}
  ```

- Show the currently selected Elixir version:

  ```
  exenv {{version}}
  ```

- Install a version of Elixir (requires `elixir-build` plugin https://github.com/mururu/elixir-build):

  ```
  exenv install {{version}}
  ```

exiftool

Read and write meta information in files.

More information: https://exiftool.org.

- Remove all EXIF metadata from the given files:

```
exiftool -All= {{file1 file2 ...}}
```

- Move the date at which all photos in a directory were taken 1 hour forward:

```
exiftool "-AllDates+=0:0:0 1:0:0" {{path/to/directory}}
```

- Move the date at which all JPEG photos in the current directory were taken 1 day and 2 hours backward:

```
exiftool "-AllDates-=0:0:1 2:0:0" -ext jpg
```

- Only change the `DateTimeOriginal` field subtracting 1.5 hours, without keeping backups:

```
exiftool -DateTimeOriginal-=1.5 -overwrite_original
```

- Recursively rename all JPEG photos in a directory based on the `DateTimeOriginal` field:

```
exiftool '-filename<DateTimeOriginal' -d %Y-%m-%d_%H-%M
%lc.%%e {{path/to/directory}} -r -ext jpg
```

exit

E xit the shell.

More information: https://manned.org/exit.

- E xit the shell with the exit code of the last command executed:

  ```
  exit
  ```

- E xit the shell with the specified exit code:

  ```
  exit {{exit_code}}
  ```

exiv2

Image metadata manipulation tool.

More information: https://www.exiv2.org/manpage.html

- Print a summary of the image Exif metadata:

```
exiv2 {{path/to/file}}
```

- Print all metadata (Exif, IPTC, XMP) with interpreted values:

```
exiv2 -Pkt {{path/to/file}}
```

- Print all metadata with raw values:

```
exiv2 -Pkv {{path/to/file}}
```

- Delete all metadata from an image:

```
exiv2 -da {{path/to/file}}
```

- Delete all metadata, preserving the file timestamp:

```
exiv2 -da -k {{path/to/file}}
```

- Rename the file, prepending the date and time from metadata (not from the file timestamp):

```
exiv2 -r {{'%Y%m%d_%H%M%S_:basename:'}} {{path/to/file}}
```

expand

Convert tabs to spaces.

More information: https://www.gnu.org/software/coreutils/expand.

- Convert tabs in each file to spaces, writing to standard output:

  ```
  expand {{file}}
  ```

- Convert tabs to spaces, reading from standard input:

  ```
  expand
  ```

- Do not convert tabs after non blanks:

  ```
  expand -i {{file}}
  ```

- Have tabs a certain number of characters apart, not 8:

  ```
  expand -t={{number}} {{file}}
  ```

- Use a comma separated list of explicit tab positions:

  ```
  expand -t={{1,4,6}}
  ```

expose

An open source tunnel application for sharing websites.

More information: https://beyondco.de/docs/expose.

- Register your authentication token:

  ```
  expose token {{token}}
  ```

- Share the current working directory:

  ```
  expose
  ```

- Share the current working directory with a specific subdomain:

  ```
  expose --subdomain={{subdomain}}
  ```

- Share a local URL:

  ```
  expose share {{url}}
  ```

- Run the Expose server:

  ```
  expose server
  ```

- Run the Expose server with a specific hostname:

  ```
  expose server {{hostname}}
  ```

expr

Evaluate expressions and manipulate strings.

More information: https://www.gnu.org/software/coreutils/expr.

- Get string length:

```
expr length {{string}}
```

- Evaluate logical or math expression with an operator ('+', '-', '*', '&', '|', etc symbols should be escaped:

```
expr {{first_arg}} {{operator}} {{second_arg}}
```

- Get position of the first character in 'string' that matches 'substring':

```
echo $(expr index {{string}} {{substring}})
```

- Extract part of the string:

```
echo $(expr substr {{string}} {{position_to_start}} {{number_of_characters}})
```

- Extract part of the string which matches a regular expression:

```
echo $(expr {{string}} : '\({{regular_expression}}\)')
```

exrex

Generate all/random matching strings for a regular expression.

It can also simplify regular expressions.

More information: https://github.com/asciimoo/exrex.

- Generate all possible strings that match a regular expression:

  ```
  exrex '{{regular_expression}}'
  ```

- Generate a random string that matches a regular expression:

  ```
  exrex --random '{{regular_expression}}'
  ```

- Generate at most 100 strings that match a regular expression:

  ```
  exrex --max-number {{100}} '{{regular_expression}}'
  ```

- Generate all possible strings that match a regular expression, joined together by a custom delimiter string:

  ```
  exrex --delimiter "{{, }}" '{{regular_expression}}'
  ```

- Print count of all possible strings that match a regular expression:

  ```
  exrex --count '{{regular_expression}}'
  ```

- Simplify a regular expression:

  ```
  exrex --simplify '{{ab|ac}}'
  ```

- Print eyes:

  ```
  exrex '{{[oO0](_)[oO0]}}'
  ```

- Print a boat:

  ```
  exrex '{{(  {20}(\|  *\\|-{22}|\|)|\.={50}|  (  ){0,5}
  ~{39})}}'
  ```

f3 fix

Edit the partition table of a fake flash drive.

See also `f3 probe`, `f3 write`, `f3 read`.

More information: http://oss.digirati.com.br/f3/

- Fill a fake flash drive with a single partition that matches its real capacity:

  ```
  sudo f3fix {{/dev/device_name}}
  ```

- Mark the partition as bootable:

  ```
  sudo f3fix --boot {{/dev/device_name}}
  ```

- Specify the filesystem:

  ```
  sudo f3fix --fs-type={{filesystem_type}} {{/dev/device_name}}
  ```

f3 probe

Probe a block device (e.g. a flash drive or a microSD card) for counterfeit flash memory.

See also: `f3read`, `f3write`, `f3fix`.

More information: https://github.com/AltraMayor/f3.

- Probe a block device:

  ```
  sudo f3probe {{path/to/blockdevice}}
  ```

- Use the minimum about of RAM possible:

  ```
  sudo f3probe --min-memory {{path/to/blockdevice}}
  ```

- Time disk operations:

  ```
  sudo f3probe --time-ops {{path/to/blockdevice}}
  ```

f3 read

Validate .h2w files to test the real capacity of the drive.

See also `f3 write`, `f3 probe`, `f3 fix`

More information: http://oss.digirati.com.br/f3/

- Validate a device by checking the files in a given directory:

  ```
  f3 read {{path/to/mount_point}}
  ```

f3 write

Fill a drive out with .h2w files to test its real capacity.

See also: `f3read`, `f3probe`, `f3fix`

More information: http://oss.digirati.com.br/f3w/

- Write test files to a given directory, filling the drive:

  ```
  f3write {{path/to/mountpoint}}
  ```

- Limit the write speed:

  ```
  f3write --max-write-rate={{kb_per_second}} {{path/to/mountpoint}}
  ```

factor

Prints the prime factorization of a number.

More information: https://www.gnu.org/software/coreutils/factor.

- Display the prime-factorization of a number:

```
factor {{number}}
```

- Take the input from stdin if no argument is specified:

```
echo {{number}} | factor
```

fakedata

Generate fake data using a large variety of generators.

More information: https://github.com/lucapette/fakedata.

- List all valid generators:

  ```
  fakedata --generators
  ```

- Generate data using one or more generators:

  ```
  fakedata {{generator1}} {{generator2}}
  ```

- Generate data with a specific output format:

  ```
  fakedata --format {{csv|tab|sql}} {{generator}}
  ```

- Generate a given number of data items (defaults to 10):

  ```
  fakedata --limit {{n}} {{generator}}
  ```

- Generate data using a custom output template (the first letter of generator names must be capitalized):

  ```
  echo "{{\{\{Generator\}\}}}" | fakedata
  ```

false

R eturns an exit code of 1 .

More information: https://www.gnu.org/software/coreutils/false.

- R eturn an exit code of 1 :

```
false
```

fast

T est your download and upload speed using fast.com.

More information: https://github.com/sindresorhus/fast-cli.

- Measure the current download speed:

  ```
  fast
  ```

- Measure the current upload speed in addition to download speed:

  ```
  fast - - u pload
  ```

- Display results on a single line to reduce spacing:

  ```
  fast - - sing le- line
  ```

fastboot

Communicate with connected Android devices when in bootloader mode (the one place `adb` doesn't work).

More information: https://android.googlesource.com/platform/system/core/+/master/fastboot/#fastboot

- Unlock the bootloader:

  ```
  fastboot oem unlock
  ```

- Relock the bootloader:

  ```
  fastboot oem lock
  ```

- Reboot the device from fastboot mode into fastboot mode again:

  ```
  fastboot reboot bootloader
  ```

- Flash a given image:

  ```
  fastboot flash {{file.zip}}
  ```

- Flash a custom recovery image:

  ```
  fastboot flash recovery {{file.img}}
  ```

- Display connected devices:

  ```
  fastboot devices
  ```

- Display all information of a device:

  ```
  fastboot getvar all
  ```

fastlane

Build and release mobile applications from the command-line.

More information: https://docs.fastlane.tools/actions/.

- Build and sign the iOS application in the current directory:

  ```
  fastlane run build_app
  ```

- Run `pod install` for the project in the current directory:

  ```
  fastlane run cocoapods
  ```

- Delete the derived data from Xcode:

  ```
  fastlane run clear_derived_data
  ```

- Remove the cache for pods:

  ```
  fastlane run clean_cocoapods_cache
  ```

fd

An alternative to `find`

Aims to be faster and easier to use than `find`

More information: https://github.com/sharkdp/fd.

- Recursively find files matching the given pattern in the current directory:

  `fd {{pattern}}`

- Find files that begin with "foo":

  `fd {{'^foo'}}`

- Find files with a specific extension:

  `fd --extension {{txt}}`

- Find files in a specific directory:

  `fd {{pattern}} {{path/to/directory}}`

- Include ignored and hidden files in the search:

  `fd --hidden --no-ignore {{pattern}}`

- Execute a command on each search result returned:

  `fd {{pattern}} --exec {{command}}`

fdp

Render an image of a **force-directed** network graph from a **graphviz** file.

Layouts: **dot**, **neato**, **twopi**, **circo**, **fdp**, **sfdp**, **osage** & **patchwork**.

More information: https://graphviz.org/doc/info/command.html

- Render a `png` image with a filename based on the input filename and output format (uppercase -O):

  ```
  fdp -T {{png}} -O {{path/to/input.gv}}
  ```

- Render a `svg` image with the specified output filename (lowercase -o):

  ```
  fdp -T {{svg}} -o {{path/to/image.svg}} {{path/to/input.gv}}
  ```

- Render the output in `ps`, `pdf`, `svgz`, `fig`, `png`, `gif`, `jpg`, `jpeg`, or `dot` format:

  ```
  fdp -T {{format}} -O {{path/to/input.gv}}
  ```

- Render a `gif` image using stdin and stdout:

  ```
  echo "{{digraph {this -> that}}}" | fdp -T {{gif}} > {{path/to/image.gif}}
  ```

- Display help:

  ```
  fdp -?
  ```

fdroid

F -Droid build tool.

F -Droid is an installable catalog of F O S S (F ree and O pen S ource S oftware) applications for the Android platform.

More information: https://f-droid.org/.

- B uild a specific app:

  ```
  fdroid b u ild {{app_id}}
  ```

- B uild a specific app in a build server V M:

  ```
  fdroid b u ild {{app_id}} - - serv er
  ```

- P ublish the app to the local repository:

  ```
  fdroid pu b lish {{app_id}}
  ```

- Install the app on every connected device:

  ```
  fdroid install {{app_id}}
  ```

- Check if the metadata is formatted correctly:

  ```
  fdroid lint - - form at {{app_id}}
  ```

- F ix the formatting automatically (if possible):

  ```
  fdroid rew ritem eta {{app_id}}
  ```

fdroidcl

F -Droid CLI client.

More information: https://github.com/mvdan/fdroidcl.

- F etch the F -Droid index:

  `fdroidcl u pdate`

- Display info about an app:

  `fdroidcl show {{app_id}}`

- Download an AP K file:

  `fdroidcl dow nload {{app_id}}`

- S earch for an app in the index:

  `fdroidcl search {{search_pattern}}`

- Install an app on a connected device:

  `fdroidcl install {{app_id}}`

fdupes

Finds duplicate files in a given set of directories.

More information: https://github.com/adrianlopez_roche/fdupes

- Search a single directory:

```
fdupes {{directory}}
```

- Search multiple directories:

```
fdupes {{directory1}} {{directory2}}
```

- Search a directory recursively:

```
fdupes -r {{directory}}
```

- Search multiple directories, one recursively:

```
fdupes {{directory1}} -R {{directory2}}
```

- Search recursively for duplicates and display interactive prompt to pick which ones to keep, deleting the others:

```
fdupes -rd {{directory}}
```

- Search recursively and delete duplicates without prompting:

```
fdupes -rdN {{directory}}
```

ffe

E xtract fields from a fl at database file and write to another format.

A configuration file is req uired to interpret the input and format the output.

More information: http://ff-extractor.sourceforge.net/ffe.html.

- Display all input data using the specified data configuration:

```
ffe - - config u ration= {{path/to/config . ffe}} {{path/to/inpu t}}
```

- Convert an input file to an output file in a new format:

```
ffe - - ou tpu t= {{path/to/ou tpu t}} - c {{path/to/config . ffe}}
{{path/to/inpu t}}
```

- S elect input structure and print format from definitions in `~ /. fferc` config file:

```
ffe - - stru ctu re= {{stru ctu re}} - - print= {{form at}} {{path/to/
inpu t}}
```

- Write only the selected fields:

```
ffe - - field- list= " {{F irstN am e, L astN am e, A g e}}"  - c {{path/to/
config . ffe}} {{path/to/inpu t}}
```

- Write only the records that match an expression:

```
ffe - e " {{L astN am e= S m ith}}"  - c {{path/to/config . ffe}} {{path/
to/inpu t}}
```

- Display help:

```
ffe - - help
```

ffmpeg

Video conversion tool.

More information: https://ffmpeg.org.

- Extract the sound from a video and save it as MP3:

```
ffmpeg -i {{video.mp4}} -vn {{sound}}.mp3
```

- Convert frames from a video or GIF into individual numbered images:

```
ffmpeg -i {{video.mpg|video.gif}} {{frame_%d.png}}
```

- Combine numbered images (`frame_1.jpg`, `frame_2.jpg`, etc) into a video or GIF:

```
ffmpeg -i {{frame_%d.jpg}} -f image2 {{video.mpg|video.g
```

- Quickly extract a single frame from a video at time mm:ss and save it as a 128x128 resolution image:

```
ffmpeg -ss {{mm:ss}} -i {{video.mp4}} -frames 1 -s
{{128x128}} -f image2 {{image.png}}
```

- Trim a video from a given start time mm:ss to an end time mm2:ss2 (omit the -to flag to trim till the end):

```
ffmpeg -ss {{mm:ss}} -to {{mm2:ss2}} -i {{video.mp4}} -cod
copy {{output.mp4}}
```

- Convert AVI video to MP4. AAC Audio @ 128kbit, h264 Video @ CRF 23:

```
ffmpeg -i {{input_video}}.avi -codec:audio aac -b:audio 128
-codec:video libx264 -crf 23 {{output_video}}.mp4
```

- Remux MKV video to MP4 without re-encoding audio or video streams:

```
ffmpeg -i {{input_video}}.mkv -codec copy
{{output_video}}.mp4
```

- Convert MP4 video to VP9 codec. For the best quality, use a CRF value (recommended range 15-35) and -b:video MUST be 0:

```
ffmpeg -i {{input_video}}.mp4 -codec:video libvpx-vp9 -crf
{{30}} -b:video 0 -codec:audio libopus -vbr on -threads
{{number_of_threads}} {{output_video}}.webm
```

ffprobe

Multimedia stream analyz er.

More information: https://ffmpeg.org/ffprobe.html.

- Display all available stream info for a media file:

```
ffprob e - v  error - show _entries {{inpu t. m p4 }}
```

- Display media duration:

```
ffprob e - v  error - show _entries form at= du ration - of
defau lt= noprint_w rappers= 1 : nok ey= 1  {{inpu t. m p4 }}
```

- Display the frame rate of a video:

```
ffprob e - v  error - select_stream s v : 0  - show _entries
stream = av g _fram e_rate - of defau lt= noprint_w rappers= 1 : nok ey= 1
{{inpu t. m p4 }}
```

- Display the width or height of a video:

```
ffprob e - v  error - select_stream s v : 0  - show _entries
stream = {{w idth| heig ht}} - of
defau lt= noprint_w rappers= 1 : nok ey= 1  {{inpu t. m p4 }}
```

- Display the average bit rate of a video:

```
ffprob e - v  error - select_stream s v : 0  - show _entries
stream = b it_rate - of defau lt= noprint_w rappers= 1 : nok ey= 1
{{inpu t. m p4 }}
```

ffsend

E asily and securely share files from command-line.

More information: https://gitlab.com/timvisee/ffsend.

- U pload a file:

  ```
  ffsend u pload {{file}}
  ```

- Download a file:

  ```
  ffsend dow nload {{u rl}}
  ```

- U pload a file with password:

  ```
  ffsend u pload {{file}} - p {{passw ord}}
  ```

- Download a file protected by password:

  ```
  ffsend dow nload {{file}} - p {{passw ord}}
  ```

- U pload a file and allow 4 downloads:

  ```
  ffsend u pload {{file}} - d {{4 }}
  ```

fg

Run jobs in foreground.

More information: https://manned.org/fg.

- Bring most recently suspended background job to foreground:

  ```
  fg
  ```

- Bring a specific job to foreground:

  ```
  fg %{{job_id}}
  ```

fgrep

Matches patterns in files.

S upports simple patterns and regular expressions.

More information: https://manned.org/fgrep.

- S earch for an exact string in a file:

```
fg rep {{search_string }} {{path/to/file}}
```

- S earch only lines that match entirely in files:

```
fg rep - x {{path/to/file1 }} {{path/to/file2 }}
```

- Count the number of lines that match the given string in a file:

```
fg rep - c {{search_string }} {{path/to/file}}
```

- S how the line number in the file along with the line matched:

```
fg rep - n {{search_string }} {{path/to/file}}
```

- Display all lines except those that contain the given regular expression:

```
fg rep - v  {{reg u lar_expression}} {{path/to/file}}
```

- Display filenames whose content matches the regular expression at least once:

```
fg rep - l {{reg u lar_expression}} {{path/to/file1 }} {{path/to/
file2 }}
```

figlet

Generate ASCII banners from user input.

Véase también `showfigfonts`

More information: http://www.figlet.org/figlet-man.html.

- Generate by directly inputting text:

  `figlet {{input_text}}`

- Use a custom font file:

  `figlet {{input_text}} -f {{path/to/font_file.flf}}`

- Use a font from the default font directory (the extension can be omitted):

  `figlet {{input_text}} -f {{font_filename}}`

- Pipe command output through figlet:

  `{{command}} | figlet`

fin

Docksal command-line utility.

More information: https://docs.docksal.io/fin/fin/.

- S tart the proj ect in the current directory:

  ```
  fin proj ect start
  ```

- S top the proj ect in the current directory:

  ```
  fin proj ect stop
  ```

- O pen a shell into a specific container:

  ```
  fin b ash {{container_nam e}}
  ```

- Display logs of a specific container:

  ```
  fin log s {{container_nam e}}
  ```

- Display logs of a specific container and follow the log:

  ```
  fin log s - f {{container_nam e}}
  ```

find

Find files or directories under the given directory tree, recursively.

More information: https://manned.org/find.

- Find files by extension:

```
find {{root_path}} -name '{{*.ext}}'
```

- Find files matching multiple path/name patterns:

```
find {{root_path}} -path '{{**/path/**/*.ext}} -or -name
'{{*pattern*}}'
```

- Find directories matching a given name, in case-insensitive mode:

```
find {{root_path}} -type d -iname '{{*lib*}}'
```

- Find files matching a given pattern, excluding specific paths:

```
find {{root_path}} -name '{{*.py}}' -not -path '{{*/site-
packages/*}}'
```

- Find files matching a given size range:

```
find {{root_path}} -size {{+500k}} -size {{-10M}}
```

- Run a command for each file (use `{}` within the command to access the filename):

```
find {{root_path}} -name '{{*.ext}}' -exec {{wc -l {}}}\;
```

- Find files modified in the last 7 days and delete them:

```
find {{root_path}} -mtime -{{7}} -delete
```

- Find empty (0 byte) files and delete them:

```
find {{root_path}} -type {{f}} -empty -delete
```

finger

User information lookup program.

More information: https://manned.org/finger.

- Display information about currently logged in users:

  ```
  finger
  ```

- Display information about a specific user:

  ```
  finger {{username}}
  ```

- Display the user's login name, real name, terminal name, and other information:

  ```
  finger -s
  ```

- Produce multiline output format displaying same information as `-s` as well as user's home directory, home phone number, login shell, mail status, etc.:

  ```
  finger -l
  ```

- Prevent matching against user's names and only use login names:

  ```
  finger -m
  ```

firebase

Test, manage, and deploy Firebase projects from the command-line.

More information: https://github.com/firebase/firebase-tools.

- Log in to https://console.firebase.google.com:

  ```
  firebase login
  ```

- List existing Firebase projects:

  ```
  firebase projects:list
  ```

- Start an interactive wizard to create a Firebase project in the current directory:

  ```
  firebase init
  ```

- Deploy code and assets to the current Firebase project:

  ```
  firebase deploy
  ```

- Start a local server to statically host the current Firebase project's assets:

  ```
  firebase serve
  ```

- Start an interactive wizard to open one of many links of the current Firebase project in the default web browser:

  ```
  firebase open
  ```

firefox

A free and open source web browser.

More information: https://developer.mozilla.org/en-US/docs/Mozilla/Command_Line_Options

- Launch Firefox and open a web page:

```
firefox {{https://www.duckduckgo.com}}
```

- Open a new window:

```
firefox --new-window {{https://www.duckduckgo.com}}
```

- Open a private (incognito) window:

```
firefox --private-window
```

- Search for "wikipedia" using the default search engine:

```
firefox --search "{{wikipedia}}"
```

- Launch Firefox in safe mode, with all extensions disabled:

```
firefox --safe-mode
```

- Take a screenshot of a web page in headless mode:

```
firefox --headless --screenshot {{path/to/output_file.png}} {{https://example.com/}}
```

- Use a specific profile to allow multiple separate instances of Firefox to run at once:

```
firefox --profile {{path/to/directory}} {{https://example.com/}}
```

- Set Firefox as the default browser:

```
firefox --setDefaultBrowser
```

fish

The Friendly Interactive Shell, a command-line interpreter designed to be user friendly.

More information: https://fishshell.com.

- Start an interactive shell session:

```
fish
```

- Execute a command and then exit:

```
fish -c "{{command}}"
```

- Execute a script:

```
fish {{path/to/script.fish}}
```

- Check a script for syntax errors:

```
fish --no-execute {{path/to/script.fish}}
```

- Start an interactive shell session in private mode, where the shell does not access old history or save new history:

```
fish --private
```

- Display version information and exit:

```
fish --version
```

- Set and export environmental variables that persist across shell restarts (from within the shell only):

```
set -Ux {{variable_name}} {{variable_value}}
```

fisher

F isher, a fish-shell plugin manager.

Install plugins by name or from a managed ' fishfile' for bundled installs.

More information: https://github.com/j orgebucaran/fisher

- Install one or more plugins:

  `fisher {{plu g in1 }} {{plu g in2 }}`

- Install a plugin from a G itH ub gist:

  `fisher {{g ist_u rl}}`

- E dit ' fishfile' by hand with your favorite editor and install multiple plugins:

  `{{editor}} ~ /. config /fish/fishfile; fisher`

- List installed plugins:

  `fisher ls`

- U pdate plugins:

  `fisher u pdate`

- R emove one or more plugins:

  `fisher rem ov e {{plu g in1 }} {{plu g in2 }}`

fkill

F abulously kill processes. Cross-platform.

More information: https://github.com/sindresorhus/fkill.

- R un without arguments to use the interactive interface:

  ```
  fk ill
  ```

- Kill the process by pid, name or port:

  ```
  fk ill {{pid| nam e| : port}}
  ```

flac

Encodes, decodes and tests flac files.

More information: https://xiph.org/flac

- Encode a wav file to flac (this will create a flac file in the same location as the wav file):

```
flac {{path/to/file.wav}}
```

- Encode a wav file to flac, specifying the output file:

```
flac -o {{path/to/output.flac}} {{path/to/file.wav}}
```

- Decode a flac file to wav, specifying the output file:

```
flac -d -o {{path/to/output.wav}} {{path/to/file.flac}}
```

- Test a flac file for the correct encoding:

```
flac -t {{path/to/file.flac}}
```

flake8

Tool to check the style and quality of Python code.

More information: https://flake8.pycqa.org/

- Lint a file or directory recursively:

  ```
  flake8 {{path/to/file_or_directory}}
  ```

- Lint a file or directory recursively and show the line on which each error occurred:

  ```
  flake8 --show-source {{path/to/file_or_directory}}
  ```

- Lint a file or directory recursively and ignore a list of rules. (All available rules can be found at flake8rules.com):

  ```
  flake8 --ignore {{rule1,rule2}} {{path/to/file_or_directory}}
  ```

- Lint a file or directory recursively but exclude files matching the given globs or substrings:

  ```
  flake8 --exclude {{substring1,glob2}} {{path/to/
  file_or_directory}}
  ```

flask

A general utility script for Flask applications. Loads the application defined in the `FLASK_APP` environment variable.

More information: https://flask.palletsprojects.com/en/1.1.x/cli/

- Run a development server:

  ```
  flask run
  ```

- Show the routes for the app:

  ```
  flask routes
  ```

- Run a Python interactive shell in the app's context:

  ```
  flask shell
  ```

flex

Lexical analyser generator. Based on **`lex`**.

Given the specification for a lexical analyser, generates C code implementing it.

More information: https://manned.org/flex

- Generate an analyser from a flex file:

  ```
  flex {{analyser.l}}
  ```

- Specify the output file:

  ```
  flex --outfile {{analyser.c}} {{analyser.l}}
  ```

- Compile a C file generated by flex:

  ```
  cc {{path/to/lex.yy.c}} --output {{executable}}
  ```

flow

A static type checker for JavaScript.

More information: https://flow.org

- Run a flow check:

  ```
  flow
  ```

- Check which files are being checked by flow:

  ```
  flow ls
  ```

- Run a type coverage check on all files in a directory:

  ```
  flow batch-coverage --show-all --strip-root {{path/to/directory}}
  ```

- Display line-by-line type coverage stats:

  ```
  flow coverage --color {{path/to/file.jsx}}
  ```

fls

List files and directories in an image file or device.

More information: https://wiki.sleuthkit.org/index.php?title=Fls

- Build a recursive fls list over a device, output paths will start with C:

```
fls -r -m {{C:}} {{/dev/loop1p1}}
```

- Analyse a single partition, providing the sector offset at which the filesystem starts in the image:

```
fls -r -m {{C:}} -o {{sector}} {{path/to/image_file}}
```

- Analyse a single partition, providing the timezone of the original system:

```
fls -r -m {{C:}} -z {{timezone}} {{/dev/loop1p1}}
```

flutter

Google's free, open source, and cross-platform mobile app SDK.

More information: https://github.com/flutter/flutter/wiki/The-flutter-tool

- Display help about a specific command:

  ```
  flutter help {{command}}
  ```

- Check if all external tools are correctly installed:

  ```
  flutter doctor
  ```

- List or change Flutter channel:

  ```
  flutter channel {{stable|beta|dev|master}}
  ```

- Run Flutter on all started emulators and connected devices:

  ```
  flutter run -d all
  ```

- Download all packages specified in `pubspec.yaml`

  ```
  flutter pub get
  ```

- Run tests in a terminal from the root of the project:

  ```
  flutter test {{test/example_test.dart}}
  ```

- Build a release APK targeting most modern smartphones:

  ```
  flutter build apk --target-platform {{android-arm}},
  {{android-arm64}}
  ```

fly

Command-line tool for concourse-ci.

More information: https://concourse-ci.org/fly.html

- Authenticate with and save concourse target:

```
fly --target {{target_name}} login --team-name {{team_name}}
-c {{https://ci.example.com}}
```

- List targets:

```
fly targets
```

- List pipelines:

```
fly -t {{target_name}} pipelines
```

- Upload or update a pipeline:

```
fly -t {{target_name}} set-pipeline --config {{pipeline.yml}}
--pipeline {{pipeline_name}}
```

- Unpause pipeline:

```
fly -t {{target_name}} unpause-pipeline --pipeline
{{pipeline_name}}
```

- Show pipeline configuration:

```
fly -t {{target_name}} get-pipeline --pipeline
{{pipeline_name}}
```

- Update local copy of fly:

```
fly -t {{target_name}} sync
```

- Destroy pipeline:

```
fly -t {{target_name}} destroy-pipeline --pipeline
{{pipeline_name}}
```

fmt

Reformat a text file by joining its paragraphs and limiting the line width to given number of characters (75 by default).

More information: https://www.gnu.org/software/coreutils/fmt.

- Reformat a file:

```
fmt {{path/to/file}}
```

- Reformat a file producing output lines of (at most) n characters:

```
fmt -w {{n}} {{path/to/file}}
```

- Reformat a file without joining lines shorter than the given width together:

```
fmt -s {{path/to/file}}
```

- Reformat a file with uniform spacing (1 space between words and 2 spaces between paragraphs):

```
fmt -u {{path/to/file}}
```

fnm

Fast Node.js version manager.

Install, uninstall or switch between Node.js versions.

More information: https://github.com/Schniz/fnm

- Install a specific version of Node.js:

  ```
  fnm install {{node_version}}
  ```

- List all available Node.js versions and highlight the default one:

  ```
  fnm ls
  ```

- Use a specific version of Node.js in the current shell:

  ```
  fnm use {{node_version}}
  ```

- Set the default Node.js version:

  ```
  fnm default {{node_version}}
  ```

- Uninstall a given Node.js version:

  ```
  fnm uninstall {{node_version}}
  ```

fold

Wraps each line in an input file to fit a specified width and prints it to the standard output.

More information: https://www.gnu.org/software/coreutils/fold.

- Wrap each line to default width (80 characters):

```
fold {{file}}
```

- Wrap each line to width "30":

```
fold -w 30 {{file}}
```

- Wrap each line to width "5" and break the line at spaces (puts each space separated word in a new line, words with length > 5 are wrapped):

```
fold -w 5 -s {{file}}
```

for

Shell loop over parameters.

More information: https://man.archlinux.org/man/for.n.

- Perform a command with different arguments:

```
for argument in 1 2 3; do {{command $argument}}; done
```

- Perform a command in every directory:

```
for d in *; do (cd $d; {{command}}); done
```

forever

Server-side JavaScript application that makes sure Node.js applications run indefinitely (restarts after exit).

More information: https://github.com/foreversd/forever.

- Start running a file forever (as a daemon):

  ```
  forever {{script}}
  ```

- List running "forever" processes (along with IDs and other details of "forever" processes):

  ```
  forever list
  ```

- Stop a running "forever" process:

  ```
  forever stop {{ID|pid|script}}
  ```

fortune

Print a random quotation (fortune-cookie style).

More information: https://man.archlinux.org/man/fortune.6.

- Print a quotation:

  ```
  fortune
  ```

- Print an offensive quotation:

  ```
  fortune -o
  ```

- Print a long quotation:

  ```
  fortune -l
  ```

- Print a short quotation:

  ```
  fortune -s
  ```

- List the available quotation database files:

  ```
  fortune -f
  ```

- Print a quotation from one of the database files listed by `fortune -f`:

  ```
  fortune {{filename}}
  ```

fossa

CLI for the Fossa service - Generate realtime license audits, vulnerability scans and reports about dependencies licenses.

More information: https://github.com/fossas/fossa-cli.

- Initialize a `fossa.yml` configuration file:

  ```
  fossa init
  ```

- Run a default project build:

  ```
  fossa build
  ```

- Analyze built dependencies:

  ```
  fossa analyze
  ```

- Generate reports:

  ```
  fossa report
  ```

- Test current revision against the FOSSA scan status and exit with errors if issues are found:

  ```
  fossa test
  ```

fping

A more powerful ping which can ping multiple hosts.

More information: https://fping.org.

- List alive hosts within a subnet generated from a netmask:

```
fping -a -g 192.168.1.0/24
```

- List alive hosts within a subnet generated from an IP range:

```
fping -a -g 192.168.1.1 192.168.1.254
```

- List unreachable hosts within a subnet generated from a netmask:

```
fping -u -g 192.168.1.0/24
```

from

Prints mail header lines from the current user's mailbox.

More information: https://mailutils.org/manual/html_chapter/Programs.html#frm-and-from.

- List mail:

  ```
  from
  ```

- Display the number of messages stored:

  ```
  from --count
  ```

- List mail in the specified mailbox directory:

  ```
  MAIL={{path/to/mailbox}} from
  ```

- Print the mail from the specified address:

  ```
  from --sender={{me@example.com}}
  ```

fselect

Find files with SQL-like queries.

More information: https://github.com/jhspetersson/fselect

- Select full path and size from temporary or config files in a given directory:

```
fselect size, path from {{path/to/directory}} where name = {{'*.cfg'}} or name = {{'*.tmp'}}
```

- Find square images:

```
fselect path from {{path/to/directory}} where width = height
```

- Find old-school rap 320 kbps MP3 files:

```
fselect path from {{path/to/directory}} where genre = {{Rap}} and bitrate = {{320}} and mp3_year lt {{2000}}
```

- Select only the first 5 results and output as JSON:

```
fselect size, path from {{path/to/directory}} limit {{5}} into json
```

- Use SQL aggregate functions to calculate minimum, maximum and average size of files in a directory:

```
fselect "{{MIN(size), MAX(size), AVG(size), SUM(size), COUNT(*)}} from {{path/to/directory}}"
```

fswatch

A cross-platform file change monitor.

More information: https://emcrisostomo.github.io/fswatch.

- Run a bash command on file creation, update or deletion:

```
fswatch {{path/to/file}} | xargs -n 1 {{bash_command}}
```

- Watch one or more files and/or directories:

```
fswatch {{path/to/file}} {{path/to/directory}} {{path/to/another_directory/**/*.js}} | xargs -n 1 {{bash_command}}
```

- Print the absolute paths of the changed files:

```
fswatch {{path/to/directory}} | xargs -n 1 -I {} echo {}
```

- Filter by event type, eg. Updated, Deleted or Created:

```
fswatch --event {{Updated}} {{path/to/directory}} | xargs -n 1 {{bash_command}}
```

fswebcam

Small and simple webcam for *nix.

More information: https://www.sanslogic.co.uk/fswebcam.

- Take a picture:

  ```
  fswebcam {{filename}}
  ```

- Take a picture with custom resolution:

  ```
  fswebcam -r {{width}}x{{height}} {{filename}}
  ```

- Take a picture from selected device(Default is `/dev/video0`):

  ```
  fswebcam -d {{device}} {{filename}}
  ```

- Take a picture with timestamp(timestamp string is formatted by strftime):

  ```
  fswebcam --timestamp {{timestamp}} {{filename}}
  ```

ftp

Tools to interact with a server via File Transfer Protocol.

- Connect to an FTP server:

  ```
  ftp {{ftp.example.com}}
  ```

- Switch to binary transfer mode (graphics, compressed files, etc):

  ```
  binary
  ```

- Transfer multiple files without prompting for confirmation on every file:

  ```
  prompt off
  ```

- Download multiple files (glob expression):

  ```
  mget {{*.png}}
  ```

- Upload multiple files (glob expression):

  ```
  mput {{*.zip}}
  ```

- Delete multiple files on the remote server:

  ```
  mdelete {{*.txt}}
  ```

- Rename a file on the remote server:

  ```
  rename {{original_filename}} {{new_filename}}
  ```

fuck

Corrects your previous console command.

More information: https://github.com/nvbn/thefuck.

- Set the `fuck` alias to `thefuck` tool:

  ```
  eval "$(thefuck --alias)"
  ```

- Try to match a rule for the previous command:

  ```
  fuck
  ```

- Confirm the first choice immediately (correct argument depends on level of annoyance):

  ```
  fuck --{{yes|yeah|hard}}
  ```

func

Az ure F unctions Core T ools: Develop and test Az ure F unctions locally.

Local functions can connect to live Az ure services, and can deploy a function app to an Az ure subscription.

More information: https://docs.microsoft.com/az ure/az ure-functions/functions-run-local.

- Create a new functions proj ect:

  ```
  fu nc init {{proj ect}}
  ```

- Create a new function:

  ```
  fu nc new
  ```

- R un functions locally:

  ```
  fu nc start
  ```

- P ublish your code to a function app in Az ure:

  ```
  fu nc azu re fu nctionapp pu b lish {{fu nction}}
  ```

- Download all settings from an existing function app:

  ```
  fu nc azu re fu nctionapp fetch- app- setting s {{fu nction}}
  ```

- G et the connection string for a specific storage account:

  ```
  fu nc azu re storag e fetch- connection- string
  {{storag e_accou nt}}
  ```

fusermount

Mount and unmount F U S E filesystems.

More information: https://man.archlinux.org/man/fusermount.1.

- U nmount a F U S E filesystem:

  ```
  fu serm ou nt - u {{path/to/m ou nt_point}}
  ```

- U nmount a F U S E filesystem as soon as it becomes unused:

  ```
  fu serm ou nt - z {{path/to/m ou nt_point}}
  ```

- Display version:

  ```
  fu serm ou nt - - v ersion
  ```

fzf

Command-line fuzzy finder.

Similar to `sk`.

More information: https://github.com/junegunn/fzf

- Start fzf on all files in the specified directory:

  ```
  find {{path/to/directory}} -type f | fzf
  ```

- Start fzf for running processes:

  ```
  ps aux | fzf
  ```

- Select multiple files with `Shift + Tab` and write to a file:

  ```
  find {{path/to/directory}} -type f | fzf --multi > {{filename}}
  ```

- Start fzf with a specified query:

  ```
  fzf --query "{{query}}"
  ```

- Start fzf on entries that start with core and end with either go, rb, or py:

  ```
  fzf --query "^core go$ | rb$ | py$"
  ```

- Start fzf on entries that not match pyc and match exactly travis:

  ```
  fzf --query "!pyc 'travis"
  ```

g++

Compiles C++ source files.

Part of GCC (GNU Compiler Collection).

More information: https://gcc.gnu.org.

- Compile a source code file into an executable binary:

```
g++ {{source.cpp}} -o {{output_executable}}
```

- Display (almost) all errors and warnings:

```
g++ {{source.cpp}} -Wall -o {{output_executable}}
```

- Choose a language standard to compile for(C++98/C++11/C++14/C++17):

```
g++ {{source.cpp}} -std={{language_standard}} -o
{{output_executable}}
```

- Include libraries located at a different path than the source file:

```
g++ {{source.cpp}} -o {{output_executable}} -I {{header_path}}
-L {{library_path}} -l{{library_name}}
```

gatsby

Static site generator for React.

More information: https://gatsbyjs.org

- Create a new site:

  ```
  gatsby new {{site_name}}
  ```

- Create a new site with a Gatsby 'starter':

  ```
  gatsby new {{site_name}} {{url_of_starter_github_repo}}
  ```

- Start a live-reloading local development server:

  ```
  gatsby develop
  ```

- Perform a production build and generate static HTML:

  ```
  gatsby build
  ```

- Start a local server which serves the production build:

  ```
  gatsby serve
  ```

gcal

Displays calendar.

More information: https://www.gnu.org/software/gcal.

- Display calendar for the current month:

```
gcal
```

- Display calendar for the month of February of the year 2010:

```
gcal {{2}} {{2010}}
```

- Provide calendar sheet with week numbers:

```
gcal --with-weeknumber
```

- Change starting day of week to 1st day of the week (Monday):

```
gcal --starting-day={{1}}
```

- Display the previous, current and next month surrounding today:

```
gcal .
```

gcalcli

Command-line tool to interact with G oogle Calendar.

R eq uests G oogle AP I authoriz ation upon first launch.

More information: https://github.com/insanum/gcalcli.

- List your events for all your calendars over the next 7 days:

  ```
  g calcli ag enda
  ```

- S how events starting from or between specific dates (also takes relative dates e.g. " tomorrow"):

  ```
  g calcli ag enda {{m m /dd}} [ {{m m /dd}}]
  ```

- List events from a specific calendar:

  ```
  g calcli - - calendar {{calendar_nam e}} ag enda
  ```

- Display an AS CII calendar of events by week:

  ```
  g calcli calw
  ```

- Display an AS CII calendar of events for a month:

  ```
  g calcli calm
  ```

- Q uick-add an event to your calendar:

  ```
  g calcli - - calendar {{calendar_nam e}} q u ick " {{m m /dd}} {{H H : M M }} {{ev ent_nam e}}"
  ```

- Add an event to calendar. T riggers interactive prompt:

  ```
  g calcli - - calendar " {{calendar_nam e}}" add
  ```

gcc

Preprocess and compile C and C++ source files, then assemble and link them together.

More information: https://gcc.gnu.org.

- Compile multiple source files into executable:

```
gcc {{source1.c}} {{source2.c}} --output {{executable}}
```

- Allow warnings, debug symbols in output:

```
gcc {{source.c}} -Wall -Og --output {{executable}}
```

- Include libraries from a different path:

```
gcc {{source.c}} --output {{executable}} -I {{header_path}} -L {{library_path}} -l{{library_name}}
```

- Compile source code into Assembler instructions:

```
gcc -S {{source.c}}
```

- Compile source code without linking:

```
gcc -c {{source.c}}
```

gcloud

The official CLI tool for Google Cloud Platform.

More information: https://cloud.google.com/sdk/gcloud.

- List all properties in one's active configuration:

  ```
  gcloud config list
  ```

- Log in to Google account:

  ```
  gcloud auth login
  ```

- Set the active project:

  ```
  gcloud config set project {{project_name}}
  ```

- SSH into a virtual machine instance:

  ```
  gcloud compute ssh {{user}}@{{instance}}
  ```

- Display all Google Compute Engine instances in a project. Instances from all zones are listed by default:

  ```
  gcloud compute instances list
  ```

- Update a kubeconfig file with the appropriate credentials to point kubectl to a specific cluster in Google Kubernetes Engine:

  ```
  gcloud container clusters get-credentials {{cluster_name}}
  ```

- Update all gcloud CLI components:

  ```
  gcloud components update
  ```

- Show help for a given command:

  ```
  gcloud help {{command}}
  ```

gdalbuildvrt

Build Virtual Datasets from a list of existing datasets.

More information: https://gdal.org/programs/gdalbuildvrt.html.

- Make a virtual mosaic from all TIFF files contained in a directory:

```
gdalbuildvrt {{path/to/output.vrt}} {{path/to/input_directory/*.tif}}
```

- Make a virtual mosaic from files whose name is specified in a text file:

```
gdalbuildvrt -input_file_list {{path/to/list.txt}} {{path/to/output.vrt}}
```

- Make a RGB virtual mosaic from 3 single-band input files:

```
gdalbuildvrt -separate {{path/to/rgb.vrt}} {{path/to/red.tif}} {{path/to/green.tif}} {{path/to/blue.tif}}
```

- Make a virtual mosaic with blue background colour (RGB: 0 0 255):

```
gdalbuildvrt -hidenodata -vrtnodata "{{0 0 255}}" {{path/to/output.vrt}} {{path/to/input_directory/*.tif}}
```

gdb

The GNU Debugger.

More information: https://www.gnu.org/software/gdb.

- Debug an executable:

```
gdb {{executable}}
```

- Attach a process to gdb:

```
gdb -p {{processID}}
```

- Debug with a core file:

```
gdb -c {{core}} {{executable}}
```

- Execute given GDB commands upon start:

```
gdb -ex "{{commands}}" {{executable}}
```

- Start gdb and pass arguments to the executable:

```
gdb --args {{executable}} {{argument1}} {{argument2}}
```

gdrive

Command-line tool to interact with G oogle Drive.

F older/file id can be obtained from the G oogle Drive folder or id url.

More information: https://github.com/gdrive-org/gdrive.

- U pload a local path to the parent folder with the specified id:

  ```
  g driv e u pload - p {{id}} {{path/to/file_or_folder}}
  ```

- Download file or directory by id to current directory:

  ```
  g driv e dow nload {{id}}
  ```

- Download to a given local path by its id:

  ```
  g driv e dow nload - - path {{path/to/folder}} {{id}}
  ```

- Create a new revision of an id using a given file or folder:

  ```
  g driv e u pdate {{id}} {{path/to/file_or_folder}}
  ```

gem

Interact with the package manager for the Ruby programming language.

More information: https://rubygems.org.

- Search for remote gem(s) and show all available versions:

  ```
  gem search {{regular_expression}} --all
  ```

- Install latest version of a gem:

  ```
  gem install {{gem name}}
  ```

- Install specific version of a gem:

  ```
  gem install {{gem name}} --version {{1.0.0}}
  ```

- Install the latest matching (SemVer) version of a gem:

  ```
  gem install {{gem name}} --version '~> {{1.0}}'
  ```

- Update a gem:

  ```
  gem update {{gem name}}
  ```

- List all local gems:

  ```
  gem list
  ```

- Uninstall a gem:

  ```
  gem uninstall {{gem name}}
  ```

- Uninstall specific version of a gem:

  ```
  gem uninstall {{gem name}} --version {{1.0.0}}
  ```

geth

The go-ethereum command-line interface.

More information: https://geth.ethereum.org.

- Connect to the main Ethereum network and automatically download the full node:

  ```
  geth
  ```

- Connect to the Ropsten test network:

  ```
  geth --testnet
  ```

- Create a new account:

  ```
  geth account new
  ```

- Enable mining:

  ```
  geth --mine
  ```

gh alias

Manage GitHub CLI command aliases from the command-line.

More information: https://cli.github.com/manual/gh_alias

- Display the subcommand help:

  ```
  gh alias
  ```

- List all of the aliases gh is configured to use:

  ```
  gh alias list
  ```

- Create a gh subcommand alias:

  ```
  gh alias set {{pv}} '{{pr view}}'
  ```

- Set a shell command as a gh subcommand:

  ```
  gh alias set --shell {{alias_name}} {{command}}
  ```

- Delete a command shortcut:

  ```
  gh alias delete {{alias_name}}
  ```

gh api

Makes authenticated HTTP requests to the GitHub API and prints the response.

More information: https://cli.github.com/manual/gh_api

- Display the subcommand help:

```
gh api --help
```

- Display the releases for the current repository in JSON format:

```
gh api repos/:owner/:repo/releases
```

- Create a reaction for a specific issue:

```
gh api --header {{Accept: application/vnd.github.squirrel-girl-preview+json}} --raw-field '{{content=+1}}' {{repos/:owner/:repo/issues/123/reactions}}
```

- Display the result of a GraphQL query in JSON format:

```
gh api graphql --field {{name=':repo'}} --raw-field '{{query}}'
```

- Send a request using a custom HTTP method:

```
gh api --method {{POST}} {{endpoint}}
```

- Include the HTTP response headers in the output:

```
gh api --include {{endpoint}}
```

- Do not print the response body:

```
gh api --silent {{endpoint}}
```

- Send a request to a specific GitHub Enterprise Server:

```
gh api --hostname {{github.example.com}} {{endpoint}}
```

gh auth

Authenticate with a GitHub host from the command-line.

More information: https://cli.github.com/manual/gh_auth

- Log in with interactive prompt:

```
gh auth login
```

- Log in with a token from standard input (created in https://github.com/settings/tokens):

```
echo {{your_token}} | gh auth login --with-token
```

- Check if you are logged in:

```
gh auth status
```

- Log out:

```
gh auth logout
```

- Log in with a specific GitHub Enterprise Server:

```
gh auth login --hostname {{github.example.com}}
```

- Refresh the session to ensure authentication credentials have the correct minimum scopes (removes additional scopes requested previously):

```
gh auth refresh
```

- Expand the permission scopes:

```
gh auth refresh --scopes {{write:org,read:public_key}}
```

gh browse

Open a GitHub repository in the browser or print the URL.

More information: https://cli.github.com/manual/gh_browse

- Open the homepage of the current repository in the default web browser:

  ```
  gh browse
  ```

- Open the homepage of a specific repository in the default web browser:

  ```
  gh browse {{owner}}/{{repository}}
  ```

- Open the settings page of the current repository in the default web browser:

  ```
  gh browse --settings
  ```

- Open the wiki of the current repository in the default web browser:

  ```
  gh browse --wiki
  ```

- Open a specific issue or pull request in the web browser:

  ```
  gh browse {{issue_or_pull_request_number}}
  ```

- Open a specific branch in the web browser:

  ```
  gh browse --branch {{branch_name}}
  ```

- Open a specific file or directory of the current repository in the web browser:

  ```
  gh browse {{path_from_root_of_repository}}
  ```

- Print the destination URL without open the web browser:

  ```
  gh browse --no-browser
  ```

gh completion

Generate shell completion scripts for GitHub CLI commands.

More information: https://cli.github.com/manual/gh_completion

- Display the subcommand help:

  ```
  gh completion
  ```

- Print a completion script:

  ```
  gh completion --shell {{bash|zsh|fish|powershell}}
  ```

- Append the gh completion script to ~/.bashrc

  ```
  gh completion --shell {{bash}} >> {{~/.bashrc}}
  ```

- Append the gh completion script to ~/.zshrc:

  ```
  gh completion --shell {{zsh}} >> {{~/.zshrc}}
  ```

gh config

Change configuration for GitHub cli.

More information: https://cli.github.com/manual/gh_config

- Display what Git protocol is being used:

  `gh config get git_protocol`

- Set protocol to SSH:

  `gh config set git_protocol {{ssh}}`

- Use `delta` in side-by-side mode as the default pager for all `gh` commands:

  `gh config set pager '{{delta --side-by-side}}'`

- Set text editor to Vim:

  `gh config set editor {{vim}}`

- Reset to default text editor:

  `gh config set editor {{""}}`

- Disable interactive prompts:

  `gh config set prompt {{disabled}}`

- Set a specific configuration value:

  `gh config set {{key}} {{value}}`

gh environment

Display help about environment variables for the G itH ub CLI command.

More information: https://cli.github.com/manual/gh_help_environment

- Display help about environment variables that can be used with `gh`

  ```
  gh help environment
  ```

gh gist

Work with GitHub Gists on the command-line.

More information: https://cli.github.com/manual/gh_gist

- Create a new Gist from a space-separated list of files:

```
gh gist create {{path/to/files}}
```

- Create a new Gist with a description:

```
gh gist create {{filename}} --desc "{{description}}"
```

- Edit a Gist:

```
gh gist edit {{id_or_url}}
```

- List Gists owned by the currently logged in user:

```
gh gist list --limit {{int}}
```

- View a Gist in the default browser without rendering Markdown:

```
gh gist view {{id_or_url}} --web --raw
```

gh help

Display help about the GitHub CLI command.

More information: https://cli.github.com/manual/gh_help

- Display general help:

  `gh help`

- Display help for the `gh help` subcommand:

  `gh help --help`

- Display help about environment variables that can be used with `gh`:

  `gh help environment`

- Display a markdown reference of all `gh` commands:

  `gh help reference`

- Display help about formatting JSON output from `gh` using `jq`:

  `gh help formatting`

- Display help about using `gh` with MinTTY:

  `gh help mintty`

- Display help for a subcommand:

  `gh help {{subcommand}}`

- Display help for a subcommand action:

  `gh help {{pr}} {{create}}`

gh issue

Create GitHub issues on a repository from the command-line.

More information: https://cli.github.com/manual/gh_issue_create

- Create a new issue against the current repository interactively:

```
gh issue create
```

- Create a new issue with the `bug` label interactively:

```
gh issue create --label "{{bug}}"
```

- Create a new issue interactively and assign it to the specified users:

```
gh issue create --assignee {{user1,user2,...}}
```

- Create a new issue with a title, body and assign it to the current user:

```
gh issue create --title "{{title}}" --body "{{body}}" --assignee "{{@me}}"
```

- Create a new issue interactively, reading the body text from a file:

```
gh issue create --body-file {{path/to/file}}
```

- Create a new issue in the default web browser:

```
gh issue create --web
```

- Display the help:

```
gh issue create --help
```

gh issue

Manage GitHub issues from the command-line.

More information: https://cli.github.com/manual/gh_issue.

- Print out the issue:

```
gh issue view {{issue_number}}
```

- Create a new issue in the web browser:

```
gh issue create --web
```

- List the last 10 issues with the bug label:

```
gh issue list --limit {{10}} --label "{{bug}}"
```

- List closed issues made by a specific user:

```
gh issue list --state closed --author {{username}}
```

- Display the status of issues relevant to the user, in a specific repository:

```
gh issue status --repo {{owner}}/{{repository}}
```

- Reopen an issue:

```
gh issue reopen {{issue_number}}
```

gh mintty

Display help about MinTTY integration for the GitHub CLI command.

More information: https://cli.github.com/manual/gh_help_mintty

- Display help about using `gh` with MinTTY:

  ```
  gh mintty
  ```

gh pr create

Manage GitHub pull requests from the command-line.

More information: https://cli.github.com/manual/gh_pr_create

- Interactively create a pull request:

  ```
  gh pr create
  ```

- Create a pull request, determining the title and description from the commit messages of the current branch:

  ```
  gh pr create --fill
  ```

- Create a draft pull request:

  ```
  gh pr create --draft
  ```

- Create a pull request specifying the base branch, title, and description:

  ```
  gh pr create --base {{base_branch}} --title "{{title}}" --body "{{body}}"
  ```

- Start opening a pull request in the browser:

  ```
  gh pr create --web
  ```

gh pr merge

Merge GitHub pull requests.

More information: https://cli.github.com/manual/gh_pr_merge

- Merge the pull request associated with the current branch interactively:

```
gh pr merge
```

- Merge the specified pull request, interactively:

```
gh pr merge {{pr_number}}
```

- Merge the pull request, removing the branch on both the local and the remote:

```
gh pr merge --delete-branch
```

- Merge the current pull request with the specified merge strategy:

```
gh pr merge --{{merge|squash|rebase}}
```

- Squash the current pull request into one commit with the message body and merge:

```
gh pr merge --squash --body="{{commit_message_body}}"
```

- Display help:

```
gh pr merge --help
```

gh pr

Manage GitHub pull requests from the command-line.

More information: https://cli.github.com/manual/gh_pr

- Create a pull request:

  ```
  gh pr create
  ```

- Check out a pull request locally:

  ```
  gh pr checkout {{pr_number}}
  ```

- View the changes made in the PR:

  ```
  gh pr diff
  ```

- Approve the pull request of the current branch:

  ```
  gh pr review --approve
  ```

- Merge the pull request associated with the current branch interactively:

  ```
  gh pr merge
  ```

- Edit a pull request interactively:

  ```
  gh pr edit
  ```

- Edit the base branch of a pull request:

  ```
  gh pr edit --base {{branch_name}}
  ```

gh reference

Display a reference about the G itH ub CLI command.

More information: https://cli.github.com/manual/gh_help_reference

- Display a markdown reference of all `gh` commands:

  ```
  gh reference
  ```

gh release

Manage GitHub releases from the command-line.

More information: https://cli.github.com/manual/gh_release

- List releases in a GitHub repository, limited to 30 items:

  ```
  gh release list
  ```

- Display information about a specific release:

  ```
  gh release view {{tag}}
  ```

- Create a new release:

  ```
  gh release create {{tag}}
  ```

- Delete a specific release:

  ```
  gh release delete {{tag}}
  ```

- Download assets from a specific release:

  ```
  gh release download {{tag}}
  ```

- Upload assets to a specific release:

  ```
  gh release upload {{tag}} {{path/to/file1}} {{path/to/file2}}
  ```

gh repo

Work with G itH ub repositories on the command-line.

More information: https://cli.github.com/manual/gh_repo

- Create a new repository (if the repository name is not set, the default name will be the name of the current directory):

  ```
  g h repo create {{nam e}}
  ```

- Clone a repository:

  ```
  g h repo clone {{ow ner}}/{{repository}}
  ```

- F ork and clone a repository:

  ```
  g h repo fork {{ow ner}}/{{repository}} - - clone
  ```

- V iew a repository in the web browser:

  ```
  g h repo v iew {{repository}} - - w eb
  ```

- List repositories owned by a specific user or organiz ation (if the owner is not set, the default owner will be the currently logged in user):

  ```
  g h repo list {{ow ner}}
  ```

- List only non-forks repositories:

  ```
  g h repo list {{ow ner}} - - non- fork s
  ```

- List repositories with a specific primary coding language:

  ```
  g h repo list {{ow ner}} - - lang u ag e {{lang u ag e_nam e}}
  ```

gh run

View, run and watch recent GitHub Actions workflow runs.

More information: https://cli.github.com/manual/gh_run

- Interactively select a run to see information about the jobs:

```
gh run view
```

- Display information about a specific run:

```
gh run view {{workflow_run_number}}
```

- Display information about the steps of a job:

```
gh run view --job={{job_number}}
```

- Display the log of a job:

```
gh run view --job={{job_number}} --log
```

- Check a specific workflow and exit with a non-zero status if the run failed:

```
gh run view {{workflow_run_number}} --exit-status && {{echo
" run pending or passed" }}
```

- Interactively select an active run and wait until it's done:

```
gh run watch
```

- Display the jobs for a run and wait until it's done:

```
gh run watch {{workflow_run_number}}
```

- Re-run a specific workflow:

```
gh run rerun {{workflow_run_number}}
```

gh secret set

Create or update GitHub secrets from the command line.

More information: https://cli.github.com/manual/gh_secret_set

- Set a secret for the current repository (user will be prompted for the value):

```
gh secret set {{name}}
```

- Set a secret from a file for the current repository:

```
gh secret set {{name}} < {{path/to/file}}
```

- Set a secret for a specific repository:

```
gh secret set {{name}} --body {{value}} --repo {{owner}}/
{{repository}}
```

- Set an organization secret for specific repositories:

```
gh secret set {{name}} --org {{organization}} --repos
"{{repository1,repository2,...}}"
```

- Set an organization secret with a specific visibility:

```
gh secret set {{name}} --org {{organization}} --visibility
{{all|private|selected}}
```

gh secret

Manage GitHub secrets from the command-line.

More information: https://cli.github.com/manual/gh_secret

- List secret keys for the current repository:

  ```
  gh secret list
  ```

- List secret keys for a specific organization:

  ```
  gh secret list --org {{organization}}
  ```

- List secret keys for a specific repository:

  ```
  gh secret list --repo {{owner}}/{{repository}}
  ```

- Set a secret for the current repository (user will be prompted for the value):

  ```
  gh secret set {{name}}
  ```

- Set a secret from a file for the current repository:

  ```
  gh secret set {{name}} < {{path/to/file}}
  ```

- Set an organization secret for specific repositories:

  ```
  gh secret set {{name}} --org {{organization}} --repos
  {{repository1,repository2}}
  ```

- Remove a secret for the current repository:

  ```
  gh secret remove {{name}}
  ```

- Remove a secret for a specific organization:

  ```
  gh secret remove {{name}} --org {{organization}}
  ```

gh ssh-key

Manage GitHub SSH keys from the command-line.

More information: https://cli.github.com/manual/gh_ssh-key

- Display help:

  ```
  gh ssh-key
  ```

- List SSH keys for the currently authenticated user:

  ```
  gh ssh-key list
  ```

- Add an SSH key to the currently authenticated user's account:

  ```
  gh ssh-key add {{path/to/key.pub}}
  ```

- Add an SSH key to the currently authenticated user's account with a specific title:

  ```
  gh ssh-key add --title {{title}} {{path/to/key.pub}}
  ```

gh workflow

List, view, and run GitHub Actions workflows.

More information: https://cli.github.com/manual/gh_workflow

- Interactively select a workflow to view the latest jobs for:

  ```
  gh workflow view
  ```

- View a specific workflow in the default browser:

  ```
  gh workflow view {{id|workflow_name|filename.yml}} --web
  ```

- Display the YAML definition of a specific workflow:

  ```
  gh workflow view {{id|workflow_name|filename.yml}} --yaml
  ```

- Display the YAML definition for a specific Git branch or tag:

  ```
  gh workflow view {{id|workflow_name|filename.yml}} --ref
  {{branch_or_tag_name}} --yaml
  ```

- List workflow files (use `--all` to include disabled workflows):

  ```
  gh workflow list
  ```

- Run a manual workflow with parameters:

  ```
  gh workflow run {{id|workflow_name|filename.yml}} --raw-fi
  {{param1}}={{value1}} --raw-field {{param2}}={{value2}}
  ```

- Run a manual workflow using a specific branch or tag with JSON parameters from stdin:

  ```
  echo '{{{"param1":"value1","param2":"value2"}}}' | g
  workflow run {{id|workflow_name|filename.yml}} --ref
  {{branch_or_tag_name}}
  ```

- Enable or disable a specific workflow:

  ```
  gh workflow {{enable|disable}} {{id|workflow_name|
  filename.yml}}
  ```

gh

Work seamlessly with G itH ub from the command-line.

More information: https://cli.github.com/.

- Clone a G itH ub repository locally:

  ```
  g h  repo  clone  {{ow ner}}/{{repository}}
  ```

- Create a new issue:

  ```
  g h  issu e  create
  ```

- V iew and filter the open issues of the current repository:

  ```
  g h  issu e  list
  ```

- V iew an issue in the browser:

  ```
  g h  issu e  v iew  - - w eb  {{issu e_nu m b er}}
  ```

- Create a pull req uest:

  ```
  g h  pr  create
  ```

- V iew a pull req uest in the browser:

  ```
  g h  pr  v iew  - - w eb  {{pr_nu m b er}}
  ```

- Locally check out the branch of a pull req uest, given its number:

  ```
  g h  pr  check ou t  {{pr_nu m b er}}
  ```

- Check the status of a repository' s pull req uests:

  ```
  g h  pr  statu s
  ```

ghc

The Glasgow Haskell Compiler.

Compiles and links Haskell source files.

More information: https://www.haskell.org/ghc.

- Find and compile all modules in the current directory:

  ```
  ghc Main
  ```

- Compile a single file:

  ```
  ghc {{file.hs}}
  ```

- Compile using extra optimization:

  ```
  ghc -O {{file.hs}}
  ```

- Stop compilation after generating object files (.o):

  ```
  ghc -c {{file.hs}}
  ```

- Run Haskell interactive interpreter (REPL):

  ```
  ghci
  ```

- Evaluate a single expression:

  ```
  ghc -e {{expression}}
  ```

ghci

The Glasgow Haskell Compiler's interactive environment.

More information: https://downloads.haskell.org/ghc/latest/docs/html/users_guide/ghci.html.

- Start a REPL (interactive shell):

  ```
  ghci
  ```

- Start a REPL and load the specified Haskell source file:

  ```
  ghci {{source_file.hs}}
  ```

- Start a REPL and enable a language option:

  ```
  ghci -X {{language_option}}
  ```

- Start a REPL and enable some level of compiler warnings (e.g. `all` or `compact`):

  ```
  ghci -W {{warning_level}}
  ```

- Start a REPL with a colon-separated list of directories for finding source files:

  ```
  ghci -i{{path/to/directory1}}:{{path/to/directory2}}
  ```

ghcup

Haskell toolchain installer.

Install, manage, and update Haskell toolchains.

More information: https://gitlab.haskell.org/haskell/ghcup-hs.

- Start the interactive TUI:

  `ghcup tui`

- List available GHC/cabal versions:

  `ghcup list`

- Install the recommended GHC version:

  `ghcup install ghc`

- Install a specific GHC version:

  `ghcup install ghc {{version}}`

- Set the currently "active" GHC version:

  `ghcup set ghc {{version}}`

- Install cabal-install:

  `ghcup install cabal`

- Update ghcup itself:

  `ghcup upgrade`

ghdl

Open-source simulator for the VHDL language.

More information: http://ghdl.free.fr.

- Analyze a VHDL source file and produce an object file:

  ```
  ghdl -a {{filename.vhdl}}
  ```

- Elaborate a design (where `{{design}}` is the name of a configuration unit, entity unit or architecture unit):

  ```
  ghdl -e {{design}}
  ```

- Run an elaborated design:

  ```
  ghdl -r {{design}}
  ```

- Run an elaborated design and dump output to a waveform file:

  ```
  ghdl -r {{design}} --wave={{output.ghw}}
  ```

- Check the syntax of a VHDL source file:

  ```
  ghdl -s {{filename.vhdl}}
  ```

- Display the help page:

  ```
  ghdl --help
  ```

ghost

A blogging platform and headless CMS .

More information: https://ghost.org.

- Install G host in the current directory:

  ```
  g host install
  ```

- S tart an instance of G host:

  ```
  g host start
  ```

- R estart the G host instance:

  ```
  g host restart
  ```

- Check the system for any potential hiccups while installing or updating G host:

  ```
  g host doctor
  ```

- V iew the logs of a G host instance:

  ```
  g host log {{nam e}}
  ```

- R un a G host instance directly (used by process managers and for debugging):

  ```
  g host ru n
  ```

- V iew running G host processes:

  ```
  g host ls
  ```

- V iew or edit G host configuration:

  ```
  g host config {{k ey}} {{v alu e}}
  ```

gibo

F etch gitignore boilerplates.

More information: https://github.com/simonwhitaker/gibo.

- List available boilerplates:

  ```
  g ib o list
  ```

- Write a boilerplate to stdout:

  ```
  g ib o du m p {{b oilerplate}}
  ```

- Write a boilerplate to .gitignore:

  ```
  g ib o du m p {{b oilerplate}} > > {{. g itig nore}}
  ```

- S earch for boilerplates containing a given string:

  ```
  g ib o search {{string }}
  ```

- U pdate available local boilerplates:

  ```
  g ib o u pdate
  ```

gifsicle

Create GIFs.

More information: https://www.lcdf.org/gifsicle.

- Optimise a GIF:

```
gifsicle --batch --optimize=3 {{animin.gif}}
```

- Make a GIF animation with gifsicle:

```
gifsicle --delay={{10}} --loop *.gif > {{anim.gif}}
```

- Extract frames from an animation:

```
gifsicle {{anim.gif}} '#0' > {{firstframe.gif}}
```

- You can also edit animations by replacing, deleting, or inserting frames:

```
gifsicle -b {{anim.gif}} --replace '#0' {{new.gif}}
```

gimp

GNU image manipulation program.

More information: https://docs.gimp.org/en/gimp-fire-up.html#_gimp-concepts-running-command-line.

- Launch GIMP:

```
gimp
```

- Launch GIMP without showing the splash screen:

```
gimp --no-splash
```

- Start a new GIMP instance, even if there is already a running one:

```
gimp --new-instance
```

- Open the given file as a new image:

```
gimp --as-new {{path/to/image}}
```

- Print errors and warnings to the console instead of showing them in a dialog box:

```
gimp --console-messages
```

- Enable debugging signal handlers:

```
gimp --debug-handlers
```

gist

Upload code to https://gist.github.com.

More information: https://github.com/defunkt/gist.

- Log in in gist on this computer:

```
gist --login
```

- Create a gist from any number of text files:

```
gist {{file.txt}} {{file2.txt}}
```

- Create a private gist with a description:

```
gist --private --description "{{A meaningful description}}" {{file.txt}}
```

- Read contents from stdin and create a gist from it:

```
{{echo "hello world"}} | gist
```

- List your public and private gists:

```
gist --list
```

- List all public gists for any user:

```
gist --list {{username}}
```

- Update a gist using the id from URL:

```
gist --update {{GIST_ID}} {{file.txt}}
```

git abort

Abort an ongoing rebase, merge, or cherry-pick.

Part of `git-extras`.

More information: https://github.com/tj/git-extras/blob/master/Commands.md#git-abort.

- Abort a Git rebase, merge, or cherry-pick:

  ```
  git abort
  ```

git add

Adds changed files to the index.

More information: https://git-scm.com/docs/git-add.

- Add a file to the index:

  `g it add {{path/to/file}}`

- Add all files (tracked and untracked):

  `g it add - A`

- O nly add already tracked files:

  `g it add - u`

- Also add ignored files:

  `g it add - f`

- Interactively stage parts of files:

  `g it add - p`

- Interactively stage parts of a given file:

  `g it add - p {{path/to/file}}`

- Interactively stage a file:

  `g it add - i`

git alias

Create shortcuts for Git commands.

Part of `git-extras`

More information: https://github.com/tj/git-extras/blob/master/Commands.md#git-alias.

- List all aliases:

  ```
  git alias
  ```

- Create a new alias:

  ```
  git alias "{{name}}" "{{command}}"
  ```

- Search for an existing alias:

  ```
  git alias ^{{name}}
  ```

git am

Apply patch files. Useful when receiving commits via email.

See also `git format-patch`, which can generate patch files.

More information: https://git-scm.com/docs/git-am.

- Apply a patch file:

```
git am {{path/to/file.patch}}
```

- Abort the process of applying a patch file:

```
git am --abort
```

- Apply as much of a patch file as possible, saving failed hunks to reject files:

```
git am --reject {{path/to/file.patch}}
```

git annex

Manage files with G it, without checking their contents in.

When a file is annexed, its content is moved into a key-value store, and a symlink is made that points to the content.

More information: https://git-annex.branchable.com.

- H elp:

  ```
  g it annex help
  ```

- Initializ e a repo with G it annex:

  ```
  g it annex init
  ```

- Add a file:

  ```
  g it annex add {{path/to/file_or_directory}}
  ```

- S how the current status of a file or directory:

  ```
  g it annex statu s {{path/to/file_or_directory}}
  ```

- S ynchroniz e a local repository with a remote:

  ```
  g it annex {{rem ote}}
  ```

- G et a file or directory:

  ```
  g it annex g et {{path/to/file_or_directory}}
  ```

git annotate

Show commit hash and last author on each line of a file.

See `git blame`, which is preferred over `git annotate`.

`git annotate` is provided for those familiar with other version control systems.

More information: https://git-scm.com/docs/git-annotate.

- Print a file with the author name and commit hash prepended to each line:

  `git annotate {{path/to/file}}`

- Print a file with the author email and commit hash prepended to each line:

  `git annotate -e {{path/to/file}}`

git apply

Apply a patch to files and/or to the index.

More information: https://git-scm.com/docs/git-apply.

- Print messages about the patched files:

```
git apply --verbose {{path/to/file}}
```

- Apply and add the patched files to the index:

```
git apply --index {{path/to/file}}
```

- Apply a remote patch file:

```
curl {{https://example.com/file.patch}} | git apply
```

- Output diffstat for the input and apply the patch:

```
git apply --stat --apply {{path/to/file}}
```

- Apply the patch in reverse:

```
git apply --reverse {{path/to/file}}
```

- Store the patch result in the index without modifying the working tree:

```
git apply --cache {{path/to/file}}
```

git archive-file

Export all the files of the current Git branch into a zip archive.

Part of `git-extras`.

More information: https://github.com/tj/git-extras/blob/master/Commands.md#git-archive-file.

- Pack the currently checked out commit into a zip archive:

  ```
  git archive-file
  ```

git archive

Create an archive of files from a named tree.

More information: https://git-scm.com/docs/git-archive.

- Create a tar archive from the contents of the current HEAD and print it to standard output:

```
git archive --verbose HEAD
```

- Create a zip archive from the current HEAD and print it to standard output:

```
git archive --verbose --format=zip HEAD
```

- Same as above, but write the zip archive to file:

```
git archive --verbose --output={{path/to/file.zip}} HEAD
```

- Create a tar archive from the contents of the latest commit on a specific branch:

```
git archive --output={{path/to/file.tar}} {{branch_name}}
```

- Create a tar archive from the contents of a specific directory:

```
git archive --output={{path/to/file.tar}} HEAD:{{path/to/directory}}
```

- Prepend a path to each file to archive it inside a specific directory:

```
git archive --output={{path/to/file.tar}} --prefix={{path/to/prepend}}/ HEAD
```

git authors

Generate a list of committers of a Git repository.

Part of `git-extras`.

More information: https://github.com/tj/git-extras/blob/master/Commands.md#git-authors.

- Print a full list of committers to stdout instead of to the `AUTHORS` file:

  ```
  git authors --list
  ```

- Append the list of committers to the the `AUTHORS` file and open it in the default editor:

  ```
  git authors
  ```

- Append the list of committers, excluding emails, to the the `AUTHORS` file and open it in the default editor:

  ```
  git authors --no-email
  ```

git bisect

Use binary search to find the commit that introduced a bug.

Git automatically jumps back and forth in the commit graph to progressively narrow down the faulty commit.

More information: https://git-scm.com/docs/git-bisect.

- Start a bisect session on a commit range bounded by a known buggy commit, and a known clean (typically older) one:

  ```
  git bisect start {{bad_commit}} {{good_commit}}
  ```

- For each commit that `git bisect` selects, mark it as "bad" or "good" after testing it for the issue:

  ```
  git bisect {{good|bad}}
  ```

- After `git bisect` pinpoints the faulty commit, end the bisect session and return to the previous branch:

  ```
  git bisect reset
  ```

- Skip a commit during a bisect (e.g. one that fails the tests due to a different issue):

  ```
  git bisect skip
  ```

git blame

Show commit hash and last author on each line of a file.

More information: https://git-scm.com/docs/git-blame.

- Print file with author name and commit hash on each line:

  ```
  git blame {{file}}
  ```

- Print file with author email and commit hash on each line:

  ```
  git blame -e {{file}}
  ```

git branch

Main G it command for working with branches.

More information: https://git-scm.com/docs/git-branch.

- List local branches. T he current branch is highlighted by*:

  ```
  g it  b ranch
  ```

- List all branches (local and remote):

  ```
  g it  b ranch  - a
  ```

- S how the name of the current branch:

  ```
  g it  b ranch  - - show - cu rrent
  ```

- Create new branch based on the current commit:

  ```
  g it  b ranch  {{b ranch_nam e}}
  ```

- Create new branch based on a specific commit:

  ```
  g it  b ranch  {{b ranch_nam e}}  {{com m it_hash}}
  ```

- R ename a branch (must not have it checked out to do this):

  ```
  g it  b ranch  - m  {{old_b ranch_nam e}}  {{new _b ranch_nam e}}
  ```

- Delete a local branch (must not have it checked out to do this):

  ```
  g it  b ranch  - d  {{b ranch_nam e}}
  ```

- Delete a remote branch:

  ```
  g it  pu sh  {{rem ote_nam e}}  - - delete  {{rem ote_b ranch_nam e}}
  ```

git browse

View an upstream repository in the default browser.

Part of **`git-extras`**.

More information: https://github.com/tj/git-extras/blob/master/Commands.md#git-browse.

- Open the first upstream in the default browser:

  ```
  git browse
  ```

- Open a specific upstream in the default browser:

  ```
  git browse {{upstream}}
  ```

git brv

Print a list of branches, sorted by last commit date.

Part of `git-extras`.

More information: https://github.com/tj/git-extras/blob/master/Commands.md#git-brv

- List each branch showing date, latest commit hash and message:

  ```
  git brv
  ```

git bugreport

Captures debug information from the system and user, generating a text file to aid in the reporting of a bug in G it.

More information: https://git-scm.com/docs/git-bugreport.

- Create a new bugreport file in the current directory:

```
g it b u g report
```

- Create a new bugreport file in the specified directory, creating it if it does not exist:

```
g it b u g report - - ou tpu t- directory {{path/to/directory}}
```

- Create a new bugreport file with the specified filename suffix in `strftim e` format:

```
g it b u g report - - su ffix {{% m % d% y}}
```

git bundle

Package objects and references into an archive.

More information: https://git-scm.com/docs/git-bundle.

- Create a bundle file that contains all objects and references of a specific branch:

```
git bundle create {{path/to/file.bundle}} {{branch_name}}
```

- Create a bundle file of all branches:

```
git bundle create {{path/to/file.bundle}} --all
```

- Create a bundle file of the last 5 commits of the current branch:

```
git bundle create {{path/to/file.bundle}} -{{5}} {{HEAD}}
```

- Create a bundle file of the latest 7 days:

```
git bundle create {{path/to/file.bundle}} --since={{7.days}} {{HEAD}}
```

- Verify that a bundle file is valid and can be applied to the current repository:

```
git bundle verify {{path/to/file.bundle}}
```

- Print to the standard output the list of references contained in a bundle:

```
git bundle unbundle {{path/to/file.bundle}}
```

- Unbundle a specific branch from a bundle file into the current repository:

```
git pull {{path/to/file.bundle}} {{branch_name}}
```

git cat-file

Provide content or type and size information for Git repository objects.

More information: https://git-scm.com/docs/git-cat-file.

- Get the [s]ize of the HEAD commit in bytes:

```
git cat-file -s HEAD
```

- Get the [t]ype (blob, tree, commit, tag) of a given Git object:

```
git cat-file -t {{8c442dc3}}
```

- Pretty-[p]rint the contents of a given Git object based on its type:

```
git cat-file -p {{HEAD~2}}
```

git check-attr

F or every pathname, list if each attribute is unspecified, set, or unset as a gitattribute on that pathname.

More information: https://git-scm.com/docs/git-check-attr.

- Check the values of all attributes on a file:

```
g it check - attr - - all {{path/to/file}}
```

- Check the value of a specific attribute on a file:

```
g it check - attr {{attrib u te}} {{path/to/file}}
```

- Check the value of a specific attribute on files:

```
g it check - attr - - all {{path/to/file1 }} {{path/to/file2 }}
```

- Check the value of a specific attribute on one or more files:

```
g it check - attr {{attrib u te}} {{path/to/file1 }} {{path/to/
file2 }}
```

git check-ignore

Analyse and debug G it ignore / exclude (" .gitignore") files.

More information: https://git-scm.com/docs/git-check-ignore.

- Check whether a file or directory is ignored:

```
g it check - ig nore {{path/to/file_or_directory}}
```

- Check whether multiple files or directories are ignored:

```
g it check - ig nore {{path/to/file}} {{path/to/directory}}
```

- U se pathnames, one per line, from stdin:

```
g it check - ig nore - - stdin <  {{path/to/file_list}}
```

- Do not check the index (used to debug why paths were tracked and not ignored):

```
g it check - ig nore - - no- index {{path/to/files_or_directories}}
```

- Include details about the matching pattern for each path:

```
g it check - ig nore - - v erb ose {{path/to/files_or_directories}}
```

git check-mailmap

S how canonical names and email addresses of contacts.

More information: https://git-scm.com/docs/git-check-mailmap.

- Look up the canonical name associated with an email address:

```
g it check - m ailm ap " < {{em ail@ exam ple. com }}> "
```

git check-ref-format

Checks if a given refname is acceptable, and exits with a non-zero status if it is not.

More information: https://git-scm.com/docs/git-check-ref-format.

- Check the format of the specified refname:

```
git check-ref-format {{refs/head/refname}}
```

- Print the name of the last branch checked out:

```
git check-ref-format --branch @{-1}
```

- Normalize a refname:

```
git check-ref-format --normalize {{refs/head/refname}}
```

git checkout-index

Copy files from the index to the working tree.

More information: https://git-scm.com/docs/git-checkout-index.

- Restore any files deleted since the last commit:

```
git checkout-index --all
```

- Restore any files deleted or changed since the last commit:

```
git checkout-index --all --force
```

- Restore any files changed since the last commit, ignoring any files that were deleted:

```
git checkout-index --all --force --no-create
```

- Export a copy of the entire tree at the last commit to the specified directory (the trailing slash is important):

```
git checkout-index --all --force --prefix={{path/to/export_directory/}}
```

git checkout

Checkout a branch or paths to the working tree.

More information: https://git-scm.com/docs/git-checkout.

- Create and switch to a new branch:

```
g it check ou t - b {{b ranch_nam e}}
```

- Create and switch to a new branch based on a specific reference (branch, remote/branch, tag are examples of valid references):

```
g it check ou t - b {{b ranch_nam e}} {{reference}}
```

- S witch to an existing local branch:

```
g it check ou t {{b ranch_nam e}}
```

- S witch to the previously checked out branch:

```
g it check ou t -
```

- S witch to an existing remote branch:

```
g it check ou t - - track {{rem ote_nam e}}/{{b ranch_nam e}}
```

- Discard all unstaged changes in the current directory (see `g it reset` for more undo-like commands):

```
g it check ou t .
```

- Discard unstaged changes to a given file:

```
g it check ou t {{filenam e}}
```

- R eplace a file in the current directory with the version of it committed in a given branch:

```
g it check ou t {{b ranch_nam e}} - - {{filenam e}}
```

git cherry-pick

Apply the changes introduced by existing commits to the current branch.

To apply changes to another branch, first use `git checkout` to switch to the desired branch.

More information: https://git-scm.com/docs/git-cherry-pick.

- Apply a commit to the current branch:

  ```
  git cherry-pick {{commit}}
  ```

- Apply a range of commits to the current branch (see also `git rebase --onto`):

  ```
  git cherry-pick {{start_commit}}~..{{end_commit}}
  ```

- Apply multiple (non-sequential) commits to the current branch:

  ```
  git cherry-pick {{commit_1}} {{commit_2}}
  ```

- Add the changes of a commit to the working directory, without creating a commit:

  ```
  git cherry-pick -n {{commit}}
  ```

git cherry

F ind commits that have yet to be applied upstream.

More information: https://git-scm.com/docs/git-cherry.

- S how commits (and their messages) with eq uivalent commits upstream:

  ```
  g it cherry - v
  ```

- S pecify a different upstream and topic branch:

  ```
  g it cherry {{orig in}} {{topic}}
  ```

- Limit commits to those within a given limit:

  ```
  g it cherry {{orig in}} {{topic}} {{b ase}}
  ```

git clean

Remove untracked files from the working tree.

More information: https://git-scm.com/docs/git-clean.

- Delete files that are not tracked by Git:

  ```
  git clean
  ```

- Interactively delete files that are not tracked by Git:

  ```
  git clean -i
  ```

- Show what files would be deleted without actually deleting them:

  ```
  git clean --dry-run
  ```

- Forcefully delete files that are not tracked by Git:

  ```
  git clean -f
  ```

- Forcefully delete directories that are not tracked by Git:

  ```
  git clean -fd
  ```

- Delete untracked files, including ignored files in `.gitignore` and `.git/info/exclude`:

  ```
  git clean -x
  ```

git clone

Clone an existing repository.

More information: https://git-scm.com/docs/git-clone.

- Clone an existing repository:

```
g it clone {{rem ote_repository_location}}
```

- Clone an existing repository into a specific directory:

```
g it clone {{rem ote_repository_location}} {{path/to/
directory}}
```

- Clone an existing repository and its submodules:

```
g it clone - - recu rsiv e {{rem ote_repository_location}}
```

- Clone a local repository:

```
g it clone - l {{path/to/local/repository}}
```

- Clone q uietly:

```
g it clone - q  {{rem ote_repository_location}}
```

- Clone an existing repository only fetching the 1 0 most recent commits on the default branch (useful to save time):

```
g it clone - - depth {{1 0 }} {{rem ote_repository_location}}
```

- Clone an existing repository only fetching a specific branch:

```
g it clone - - b ranch {{nam e}} - - sing le- b ranch
{{rem ote_repository_location}}
```

- Clone an existing repository using a specific S S H command:

```
g it clone - - config  core. sshC om m and= " {{ssh - i path/to/
priv ate_ssh_k ey}}"  {{rem ote_repository_location}}
```

git column

Display data in columns.

More information: https://git-scm.com/docs/git-column.

- Format the standard input as multiple columns:

```
ls | git column --mode={{column}}
```

- Format the standard input as multiple columns with a maximum width of 100:

```
ls | git column --mode=column --width={{100}}
```

- Format the standard input as multiple columns with a maximum padding of 30:

```
ls | git column --mode=column --padding={{30}}
```

git commit-graph

Write and verify Git commit-graph files.

More information: https://git-scm.com/docs/git-commit-graph.

- Write a commit-graph file for the packed commits in the repository's local `.git` directory:

  ```
  git commit-graph write
  ```

- Write a commit-graph file containing all reachable commits:

  ```
  git show-ref --hash | git commit-graph write --stdin-commits
  ```

- Write a commit-graph file containing all commits in the current commit-graph file along with those reachable from `HEAD`:

  ```
  git rev-parse {{HEAD}} | git commit-graph write --stdin-commits --append
  ```

git commit-tree

Low level utility to create commit objects.

See also: `git commit`

More information: https://git-scm.com/docs/git-commit-tree.

- Create a commit object with the specified message:

  ```
  git commit-tree {{tree}} -m "{{message}}"
  ```

- Create a commit object reading the message from a file (use - for stdin):

  ```
  git commit-tree {{tree}} -F {{path/to/file}}
  ```

- Create a GPG-signed commit object:

  ```
  git commit-tree {{tree}} -m "{{message}}" --gpg-sign
  ```

- Create a commit object with the specified parent commit object:

  ```
  git commit-tree {{tree}} -m "{{message}}" -p
  {{parent_commit_sha}}
  ```

git commit

Commit files to the repository.

More information: https://git-scm.com/docs/git-commit.

- Commit staged files to the repository with a message:

```
g it com m it - m " {{m essag e}}"
```

- Commit staged files with a message read from a file:

```
g it com m it - - file {{path/to/com m it_m essag e_file}}
```

- Auto stage all modified files and commit with a message:

```
g it com m it - a - m " {{m essag e}}"
```

- Commit staged files and [S] ign them with the G P G key defined in g itconfig:

```
g it com m it - S - m " {{m essag e}}"
```

- U pdate the last commit by adding the currently staged changes, changing the commit' s hash:

```
g it com m it - - am end
```

- Commit only specific (already staged) files:

```
g it com m it {{path/to/file1 }} {{path/to/file2 }}
```

- Create a commit, even if there are no staged files:

```
g it com m it - m " {{m essag e}}" - - allow - em pty
```

git commits-since

Display commits since a specific time or date.

Part of `git-extras`

More information: https://github.com/tj/git-extras/blob/master/Commands.md#git-commits-since.

- Display commits since yesterday:

  ```
  git commits-since {{yesterday}}
  ```

- Display commits since last week:

  ```
  git commits-since {{last week}}
  ```

- Display commits since last month:

  ```
  git commits-since {{last month}}
  ```

- Display commits since yesterday 2 pm:

  ```
  git commits-since {{yesterday 2 pm}}
  ```

git config

Manage custom configuration options for Git repositories.

These configurations can be local (for the current repository) or global (for the current user).

More information: https://git-scm.com/docs/git-config.

- List only local configuration entries (stored in `.git/config` in the current repository):

```
git config --list --local
```

- List only global configuration entries (stored in `~/.gitconfig`):

```
git config --list --global
```

- List all configuration entries that have been defined either locally or globally:

```
git config --list
```

- Get the value of a given configuration entry:

```
git config alias.unstage
```

- Set the global value of a given configuration entry:

```
git config --global alias.unstage "reset HEAD --"
```

- Revert a global configuration entry to its default value:

```
git config --global --unset alias.unstage
```

- Edit the Git configuration for the current repository in the default editor:

```
git config --edit
```

- Edit the global Git configuration in the default editor:

```
git config --global --edit
```

git contrib

Display commits from a specific author.

Part of `git-extras`.

More information: https://github.com/tj/git-extras/blob/master/Commands.md#git-contrib.

- Display all commit hashes and their corresponding commit messages from a specific author:

  ```
  git contrib {{author}}
  ```

git count-objects

Count the number of unpacked objects and their disk consumption.

More information: https://git-scm.com/docs/git-count-objects

- Count all objects and display the total disk usage:

```
git count-objects
```

- Display a count of all objects and their total disk usage, displaying sizes in human readable units:

```
git count-objects --human-readable
```

- Display more verbose information:

```
git count-objects --verbose
```

- Display more verbose information, displaying sizes in human readable units:

```
git count-objects --human-readable --verbose
```

git count

Print the total number of commits.

Part of `git-extras`.

More information: https://github.com/tj/git-extras/blob/master/Commands.md#git-count.

- Print the total number of commits:

  ```
  git count
  ```

- Print the number of commits per contributor and the total number of commits:

  ```
  git count --all
  ```

git cp

Copy an existing file to a new location, preserving history.

Part of `git-extras`

More information: https://github.com/tj/git-extras/blob/master/Commands.md#git-cp

- Copy an existing file in a git repo, staying in the same directory:

  ```
  git cp {{file}} {{new_file}}
  ```

- Copy an existing file in a git repo and place it elsewhere:

  ```
  git cp {{path/to/file}} {{path/to/new_file}}
  ```

git create-branch

Create a G it branch in a repository.

P art of `g i t - ex t ra s`

More information: https://github.com/tj /git-extras/blob/master/Commands.md# git-create-branch.

- Create a local branch:

  ```
  g it create- b ranch {{b ranch_nam e}}
  ```

- Create a branch locally and on origin:

  ```
  g it create- b ranch - - rem ote {{b ranch_nam e}}
  ```

- Create a branch locally and on upstream (through forks):

  ```
  g it create- b ranch - - rem ote u pstream  {{b ranch_nam e}}
  ```

git credential

Retrieve and store user credentials.

More information: https://git-scm.com/docs/git-credential.

- Display credential information, retrieving the username and password from configuration files:

```
echo "{{url=http://example.com}}" | git credential fill
```

- Send credential information to all configured credential helpers to store for later use:

```
echo "{{url=http://example.com}}" | git credential approve
```

- Erase the specified credential information from all the configured credential helpers:

```
echo "{{url=http://example.com}}" | git credential reject
```

git daemon

A really simple server for G it repositories.

More information: https://git-scm.com/docs/git-daemon.

- Launch a G it daemon with a whitelisted set of directories:

```
g it daem on - - export- all {{path/to/directory1 }} {{path/to/
directory2 }}
```

- Launch a G it daemon with a specific base directory and allow pulling from all sub-directories that look like G it repositories:

```
g it daem on - - b ase- path= {{path/to/directory}} - - export- all - -
reu seaddr
```

- Launch a G it daemon for the specified directory, verbosely printing log messages and allowing G it clients to write to it:

```
g it daem on {{path/to/directory}} - - enab le= receiv e- pack  - -
inform ativ e- errors - - v erb ose
```

git delete-branch

Delete local and remote G it branches.

P art of `g i t - ex t ras`. If deleting the checked out branch, only the remote branch will be deleted.

More information: [https://github.com/tj /git-extras/blob/master/Commands.md# git-delete-branch](https://github.com/tj/git-extras/blob/master/Commands.md#git-delete-branch).

- Delete a local and remote G it branch:

  ```
  g it delete- b ranch {{b ranch_nam e}}
  ```

- Delete multiple local and remote G it branches:

  ```
  g it delete- b ranch {{b ranch_nam e1  b ranch_nam e2  . . . }}
  ```

git delete-tag

Delete existing local and remote tags.

P art of git-extras.

More information: [https://github.com/tj /git-extras/blob/master/Commands.md# git-delete-tag](https://github.com/tj/git-extras/blob/master/Commands.md#git-delete-tag).

- Delete a tag:

```
g it delete- tag  {{tag _v ersion}}
```

git delta

List files that differ from another branch.

P art of git-extras.

More information: [https://github.com/tj /git-extras/blob/master/Commands.md# git-delta](https://github.com/tj/git-extras/blob/master/Commands.md#git-delta).

- List files from the current checked out branch that differ from the `m ain` branch:

  ```
  g it delta {{m ain}}
  ```

- List files from specific branch that differ from another specific branch:

  ```
  g it delta {{b ranch_1 }} {{b ranch_2 }}
  ```

git describe

G ive an obj ect a human readable name based on an available ref.

More information: https://git-scm.com/docs/git-describe.

- Create a uniq ue name for the current commit (the name contains the most recent annotated tag, the number of additional commits, and the abbreviated commit hash):

  ```
  g it  describ e
  ```

- Create a name with 4 digits for the abbreviated commit hash:

  ```
  g it  describ e  - - ab b rev = {{4 }}
  ```

- G enerate a name with the tag reference path:

  ```
  g it  describ e  - - all
  ```

- Describe a G it tag:

  ```
  g it  describ e  {{v 1 . 0 . 0 }}
  ```

- Create a name for the last commit of a given branch:

  ```
  g it  describ e  {{b ranch_nam e}}
  ```

git diff

Show changes to tracked files.

More information: https://git-scm.com/docs/git-diff.

- Show unstaged, uncommitted changes:

```
git diff
```

- Show all uncommitted changes (including staged ones):

```
git diff HEAD
```

- Show only staged (added, but not yet committed) changes:

```
git diff --staged
```

- Show changes from all commits since a given date/time (a date expression, e.g. "1 week 2 days" or an ISO date):

```
git diff 'HEAD@{3 months|weeks|days|hours|seconds ago}'
```

- Show only names of changed files since a given commit:

```
git diff --name-only {{commit}}
```

- Output a summary of file creations, renames and mode changes since a given commit:

```
git diff --summary {{commit}}
```

- Compare a single file between two branches or commits:

```
git diff {{branch_1}}..{{branch_2}} [--] {{path/to/file}}
```

- Compare different files from the current branch to other branch:

```
git diff {{branch}}:{{path/to/file2}} {{path/to/file}}
```

git difftool

Show file changes using external diff tools. Accepts the same options and arguments as `git diff`.

See also: `git diff`.

More information: https://git-scm.com/docs/git-difftool.

- List available diff tools:

  ```
  git difftool --tool-help
  ```

- Set the default diff tool to meld:

  ```
  git config --global diff.tool "{{meld}}"
  ```

- Use the default diff tool to show staged changes:

  ```
  git difftool --staged
  ```

- Use a specific tool (opendiff) to show changes since a given commit:

  ```
  git difftool --tool={{opendiff}} {{commit}}
  ```

git effort

Display how much activity a file has had, showing commits per file and "active days" i.e. total number of days that contributed to the file.

Part of `git-extras`.

More information: https://github.com/tj/git-extras/blob/master/Commands.md#git-effort.

- Display each file in the repository, showing commits and active days:

```
git effort
```

- Display files modified by a specific number of commits or more, showing commits and active days:

```
git effort --above {{5}}
```

- Display files modified by a specific author, showing commits and active days:

```
git effort --author="{{username}}"
```

- Display files modified since a specific time/date, showing commits and active days:

```
git effort --since="{{last month}}"
```

- Display only the specified files or directories, showing commits and active days:

```
git effort {{path/to/file_or_directory1 path/to/file_or_directory2 ...}}
```

- Display all files in a specific directory, showing commits and active days:

```
git effort {{path/to/directory/*}}
```

git extras

Git extension pack.

More information: https://github.com/tj/git-extras.

- Install or upgrade `git-extras` commands:

  `git extras update`

- Display help and subcommands:

  `git extras --help`

- Print the version of installed `git-extras`:

  `git extras --version`

git fame

Pretty-print Git repository contributions.

More information: https://github.com/casperdcl/git-fame.

- Calculate contributions for the current Git repository:

```
git fame
```

- Exclude files/directories that match the specified regular expression:

```
git fame --excl "{{regular_expression}}"
```

- Calculate contributions made after the specified date:

```
git fame --since "{{3 weeks ago|2021-05-13}}"
```

- Display contributions in the specified format:

```
git fame --format {{pipe|yaml|json|csv|tsv}}
```

- Display contributions per file extension:

```
git fame --bytype
```

- Ignore whitespace changes:

```
git fame --ignore-whitespace
```

- Detect inter-file line moves and copies:

```
git fame -C
```

- Detect intra-file line moves and copies:

```
git fame -M
```

git fetch

Download objects and refs from a remote repository.

More information: https://git-scm.com/docs/git-fetch.

- Fetch the latest changes from the default remote upstream repository (if set):

  ```
  git fetch
  ```

- Fetch new branches from a specific remote upstream repository:

  ```
  git fetch {{remote_name}}
  ```

- Fetch the latest changes from all remote upstream repositories:

  ```
  git fetch --all
  ```

- Also fetch tags from the remote upstream repository:

  ```
  git fetch --tags
  ```

- Delete local references to remote branches that have been deleted upstream:

  ```
  git fetch --prune
  ```

git flow

A collection of Git extensions to provide high-level repository operations.

More information: https://github.com/nvie/gitflow

- Initialize it inside an existing Git repository:

  `git flow init`

- Start developing on a feature branch based on `develop`

  `git flow feature start {{feature}}`

- Finish development on a feature branch, merging it into the `develop` branch and deleting it:

  `git flow feature finish {{feature}}`

- Publish a feature to the remote server:

  `git flow feature publish {{feature}}`

- Get a feature published by another user:

  `git flow feature pull origin {{feature}}`

git for-each-repo

Run a Git command on a list of repositories.

Note: this command is experimental and may change.

More information: https://git-scm.com/docs/git-for-each-repo.

- Run maintenance on each of a list of repositories stored in the `maintenance.repo` user configuration variable:

  ```
  git for-each-repo --config={{maintenance.repo}} {{maintenance run}}
  ```

- Run `git pull` on each repository listed in a global configuration variable:

  ```
  git for-each-repo --config={{global_configuration_variable}} {{pull}}
  ```

git fork

Fork the given GitHub repo. Like `git clone` but forks first.

Part of `git-extras`.

More information: https://github.com/tj/git-extras/blob/master/Commands.md#git-fork.

- Fork and clone a GitHub repository by its URL:

  ```
  git fork {{https://github.com/tldr-pages/tldr}}
  ```

- Fork and clone a GitHub repository by its slug:

  ```
  git fork {{tldr-pages/tldr}}
  ```

git format-patch

Prepare .patch files. Useful when emailing commits elsewhere.

See also `git am`, which can apply generated .patch files.

More information: https://git-scm.com/docs/git-format-patch.

- Create an auto-named `.patch` file for all the unpushed commits:

  ```
  git format-patch {{origin}}
  ```

- Write a `.patch` file for all the commits between 2 revisions to stdout:

  ```
  git format-patch {{revision_1}}..{{revision_2}}
  ```

- Write a `.patch` file for the 3 latest commits:

  ```
  git format-patch -{{3}}
  ```

git fsck

Verify the validity and connectivity of nodes in a Git repository index.

Does not make any modifications. See `git gc` for cleaning up dangling blobs.

More information: https://git-scm.com/docs/git-fsck.

- Check the current repository:

```
git fsck
```

- List all tags found:

```
git fsck --tags
```

- List all root nodes found:

```
git fsck --root
```

git gc

Optimise the local repository by cleaning unnecessary files.

More information: https://git-scm.com/docs/git-gc.

- Optimise the repository:

  ```
  g it g c
  ```

- Aggressively optimise, takes more time:

  ```
  g it g c - - ag g ressiv e
  ```

- Do not prune loose obj ects (prunes by default):

  ```
  g it g c - - no- pru ne
  ```

- Suppress all output:

  ```
  g it g c - - q u iet
  ```

- View full usage:

  ```
  g it g c - - help
  ```

git graft

Merge commits from a specific branch into another branch and delete the source branch.

Part of `git-extras`.

More information: https://github.com/tj/git-extras/blob/master/Commands.md#git-graft.

- Merge all commits not present on the target branch from the source branch to target branch, and delete the source branch:

  ```
  git graft {{source_branch}} {{target_branch}}
  ```

git-grep

Find strings inside files anywhere in a repository's history.

Accepts a lot of the same flags as regular `grep`

More information: https://git-scm.com/docs/git-grep.

- Search for a string in tracked files:

  ```
  git grep {{search_string}}
  ```

- Search for a string in files matching a pattern in tracked files:

  ```
  git grep {{search_string}} -- {{file_glob_pattern}}
  ```

- Search for a string in tracked files, including submodules:

  ```
  git grep --recurse-submodules {{search_string}}
  ```

- Search for a string at a specific point in history:

  ```
  git grep {{search_string}} {{HEAD~2}}
  ```

- Search for a string across all branches:

  ```
  git grep {{search_string}} $(git rev-list --all)
  ```

git help

Display help information about G it.

More information: https://git-scm.com/docs/git-help.

- Display help about a specific G it subcommand:

```
g it help {{su b com m and}}
```

- Display help about a specific G it subcommand in a web browser:

```
g it help - - w eb  {{su b com m and}}
```

- Display a list of all available G it subcommands:

```
g it help - - all
```

- List the available guides:

```
g it help - - g u ide
```

- List all possible configuration variables:

```
g it help - - config
```

git ignore-io

G enerate .gitignore files from predefined templates.

More information: [https://github.com/tj /git-extras/blob/master/Commands.md# _git-ignore-io](https://github.com/tj/git-extras/blob/master/Commands.md#_git-ignore-io).

- List available templates:

  ```
  g it ig nore- io list
  ```

- G enerate a .gitignore template:

  ```
  g it ig nore- io {{item _a, item _b , item _n}}
  ```

git ignore

Show/update `.gitignore` files.

Part of `git-extras`. See also `git ignore-io`.

More information: https://github.com/tj/git-extras/blob/master/Commands.md#git-ignore.

- Show the content of all global and local `.gitignore` files:

  `git ignore`

- Ignore file(s) privately, updating `.git/info/exclude` file:

  `git ignore {{file_pattern}} --private`

- Ignore file(s) locally, updating local `.gitignore` file:

  `git ignore {{file_pattern}}`

- Ignore file(s) globally, updating global `.gitignore` file:

  `git ignore {{file_pattern}} --global`

git-imerge

Perform a merge or rebase between two Git branches incrementally.

Conflicts between branches are tracked down to pairs of individual commits, to simplify conflict resolution.

More information: https://github.com/mhagger/git-imerge.

- Start imerge-based rebase (checkout the branch to be rebased, first):

  `git imerge rebase {{branch_to_rebase_onto}}`

- Start imerge-based merge (checkout the branch to merge into, first):

  `git imerge merge {{branch_to_be_merged}}`

- Show ASCII diagram of in-progress merge or rebase:

  `git imerge diagram`

- Continue imerge operation after resolving conflicts (`git add` the conflicted files, first):

  `git imerge continue --no-edit`

- Wrap up imerge operation, after all conflicts are resolved:

  `git imerge finish`

- Abort imerge operation, and return to the previous branch:

  `git-imerge remove && git checkout {{previous_branch}}`

git info

Display Git repository information.

Part of `git-extras`.

More information: https://github.com/tj/git-extras/blob/master/Commands.md#git-info.

- Display remote locations, remote and local branches, most recent commit data and `.git/config` settings:

  ```
  git info
  ```

- Display remote locations, remote and local branches and most recent commit data:

  ```
  git info --no-config
  ```

git init

Initializes a new local Git repository.

More information: https://git-scm.com/docs/git-init.

- Initialize a new local repository:

  ```
  git init
  ```

- Initialize a repository with the specified name for the initial branch:

  ```
  git init --initial-branch={{branch_name}}
  ```

- Initialize a repository using SHA256 for object hashes (requires Git version 2.29+):

  ```
  git init --object-format={{sha256}}
  ```

- Initialize a barebones repository, suitable for use as a remote over ssh:

  ```
  git init --bare
  ```

git instaweb

H elper to launch a gitweb server.

More information: https://git-scm.com/docs/git-instaweb.

- Launch a gitweb server for the current G it repository:

```
g it instaw eb  - - start
```

- Listen only on localhost:

```
g it instaw eb  - - start - - local
```

- Listen on a specific port:

```
g it instaw eb  - - start - - port {{1 2 3 4 }}
```

- U se a specified http daemon:

```
g it instaw eb  - - start - - httpd {{lig httpd| apache2 | m ong oose|
plack u p| w eb rick }}
```

- Also auto-launch a web browser:

```
g it instaw eb  - - start - - b row ser
```

- S top the currently running gitweb server:

```
g it instaw eb  - - stop
```

- R estart the currently running gitweb server:

```
g it instaw eb  - - restart
```

git lfs

Work with large files in Git repositories.

More information: https://git-lfs.github.com.

- Initialise Git LFS:

```
git lfs install
```

- Track files that match a glob:

```
git lfs track '{{*.bin}}'
```

- Change the Git LFS endpoint URL (useful if the LFS server is separate from the Git server):

```
git config -f .lfsconfig lfs.url {{lfs_endpoint_url}}
```

- List tracked patterns:

```
git lfs track
```

- List tracked files that have been committed:

```
git lfs ls-files
```

- Push all Git LFS objects to the remote server (useful if errors are encountered):

```
git lfs push --all {{remote_name}} {{branch_name}}
```

- Fetch all Git LFS objects:

```
git lfs fetch
```

- Checkout all Git LFS objects:

```
git lfs checkout
```

git local-commits

Show local commits that haven't been pushed to origin. Any additional arguments will be passed directly to `git log`

Part of `git-extras`.

More information: https://github.com/tj/git-extras/blob/master/Commands.md#git-local-commits.

- Show commits that haven't been pushed:

```
git local-commits
```

git log

S how a history of commits.

More information: https://git-scm.com/docs/git-log.

- S how the seq uence of commits starting from the current one, in reverse chronological order of the G it repository in the current working directory:

  ```
  g it log
  ```

- S how the history of a particular file or directory, including differences:

  ```
  g it log - p {{path/to/file_or_directory}}
  ```

- S how an overview of which file(s) changed in each commit:

  ```
  g it log - - stat
  ```

- S how a graph of commits in the current branch using only the first line of each commit message:

  ```
  g it log - - oneline - - g raph
  ```

- S how a graph of all commits, tags and branches in the entire repo:

  ```
  g it log - - oneline - - decorate - - all - - g raph
  ```

- S how only commits whose messages include a given string (case-insensitively):

  ```
  g it log - i - - g rep {{search_string }}
  ```

- S how the last N commits from a certain author:

  ```
  g it log - n {{nu m b er}} - - au thor= {{au thor}}
  ```

- S how commits between two dates:

  ```
  g it log - - b efore= {{date}} - - after= {{date}}
  ```

git ls-files

Show information about files in the index and the working tree.

More information: https://git-scm.com/docs/git-ls-files.

- Show deleted files:

```
git ls-files --deleted
```

- Show modified and deleted files:

```
git ls-files --modified
```

- Show ignored and untracked files:

```
git ls-files --others
```

git ls-remote

Git command for listing references in a remote repository based on name or URL.

If no name or URL are given, then the configured upstream branch will be used, or remote origin if the former is not configured.

More information: https://git-scm.com/docs/git-ls-remote.

- Show all references in the default remote repository:

```
git ls-remote
```

- Show only heads references in the default remote repository:

```
git ls-remote --heads
```

- Show only tags references in the default remote repository:

```
git ls-remote --tags
```

- Show all references from a remote repository based on name or URL:

```
git ls-remote {{repository_url}}
```

- Show references from a remote repository filtered by a pattern:

```
git ls-remote {{repository_name}} "{{pattern}}"
```

git ls-tree

List the contents of a tree obj ect.

More information: https://git-scm.com/docs/git-ls-tree.

- List the contents of the tree on a branch:

  ```
  g it ls- tree {{b ranch_nam e}}
  ```

- List the contents of the tree on a commit, recursing into subtrees:

  ```
  g it ls- tree - r {{com m it_hash}}
  ```

- List only the filenames of the tree on a commit:

  ```
  g it ls- tree - - nam e- only {{com m it_hash}}
  ```

git mailinfo

Extract patch and authorship information from a single email message.

More information: https://git-scm.com/docs/git-mailinfo.

- Extract the patch and author data from an email message:

```
git mailinfo {{message| patch}}
```

- Extract but remove leading and trailing whitespace:

```
git mailinfo -k {{message| patch}}
```

- Remove everything from the body before a scissors line (e.g. "--> * --") and retrieve the message or patch:

```
git mailinfo --scissors {{message| patch}}
```

git-maintenance

Run tasks to optimize Git repository data.

More information: https://git-scm.com/docs/git-maintenance.

- Register the current repository in the user's list of repositories to daily have maintenance run:

  ```
  git maintenance register
  ```

- Start running maintenance on the current repository:

  ```
  git maintenance start
  ```

- Halt the background maintenance schedule for the current repository:

  ```
  git maintenance stop
  ```

- Remove the current repository from the user's maintenance repository list:

  ```
  git maintenance unregister
  ```

- Run a specific maintenance task on the current repository:

  ```
  git maintenance run --task={{commit-graph|gc|incremental-repack|loose-objects|pack-refs|prefetch}}
  ```

git merge

Merge branches.

More information: https://git-scm.com/docs/git-merge.

- Merge a branch into your current branch:

  ```
  g it  m erg e  {{b ranch_nam e}}
  ```

- E dit the merge message:

  ```
  g it  m erg e  - e  {{b ranch_nam e}}
  ```

- Merge a branch and create a merge commit:

  ```
  g it  m erg e  - - no- ff  {{b ranch_nam e}}
  ```

- Abort a merge in case of confl icts:

  ```
  g it  m erg e  - - ab ort
  ```

git mergetool

Run merge conflict resolution tools to resolve merge conflicts.

More information: https://git-scm.com/docs/git-mergetool.

- Launch the default merge tool to resolve conflicts:

  ```
  git mergetool
  ```

- List valid merge tools:

  ```
  git mergetool --tool-help
  ```

- Launch the merge tool identified by a name:

  ```
  git mergetool --tool {{tool_name}}
  ```

- Don't prompt before each invocation of the merge tool:

  ```
  git mergetool --no-prompt
  ```

- Explicitly use the GUI merge tool (see the `merge.guitool` config variable):

  ```
  git mergetool --gui
  ```

- Explicitly use the regular merge tool (see the `merge.tool` config variable):

  ```
  git mergetool --no-gui
  ```

git missing

Show commits which aren't shared between two branches.

Part of `git-extras`

More information: https://github.com/tj/git-extras/blob/master/Commands.md#git-missing.

- Show commits which aren't shared between the currently checked-out branch and another branch:

  ```
  git missing {{branch}}
  ```

- Show commits which aren't shared between two branches:

  ```
  git missing {{branch_1}} {{branch_2}}
  ```

git mv

Move or rename files and update the G it index.

More information: https://git-scm.com/docs/git-mv.

- Move file inside the repo and add the movement to the next commit:

```
g it m v  {{path/to/file}} {{new /path/to/file}}
```

- R ename file and add renaming to the next commit:

```
g it m v  {{filenam e}} {{new _filenam e}}
```

- O verwrite the file in the target path if it exists:

```
g it m v  - - force {{file}} {{targ et}}
```

git notes

Add or inspect obj ect notes.

More information: https://git-scm.com/docs/git-notes.

- List all notes and the obj ects they are attached to:

```
g it notes list
```

- List all notes attached to a given obj ect (defaults to H E AD):

```
g it notes list [ {{ob j ect}}]
```

- S how the notes attached to a given obj ect (defaults to H E AD):

```
g it notes show  [ {{ob j ect}}]
```

- Append a note to a specified obj ect (opens the default text editor):

```
g it notes append {{ob j ect}}
```

- Append a note to a specified obj ect, specifying the message:

```
g it notes append - - m essag e= " {{m essag e_text}}"
```

- E dit an existing note (defaults to H E AD):

```
g it notes edit [ {{ob j ect}}]
```

- Copy a note from one obj ect to another:

```
g it notes copy {{sou rce_ob j ect}} {{targ et_ob j ect}}
```

- R emove all the notes added to a specified obj ect:

```
g it notes rem ov e {{ob j ect}}
```

git obliterate

Delete specific files and erase their history from a Git repository.

Part of `git-extras`.

More information: https://github.com/tj/git-extras/blob/master/Commands.md#git-obliterate.

- Erase the existence of specific files:

  ```
  git obliterate {{file_1 file_2 ...}}
  ```

- Erase the existence of specific files between 2 commits:

  ```
  git obliterate {{file_1 file_2 ...}} --{{commit_hash_1}}..{{commit_hash_2}}
  ```

git pr

Check out GitHub pull requests locally.

More information: https://github.com/tj/git-extras/blob/master/Commands.md#git-pr

- Check out a specific pull request:

  `git pr {{pr_number}}`

- Check out a pull request for a specific remote:

  `git pr {{pr_number}} {{remote}}`

- Check out a pull request from its URL:

  `git pr {{url}}`

- Clean up old pull request branches:

  `git pr clean`

git prune

Git command for pruning all unreachable objects from the object database.

This command is often not used directly but as an internal command that is used by Git gc.

More information: https://git-scm.com/docs/git-prune.

- Report what would be removed by Git prune without removing it:

  ```
  git prune --dry-run
  ```

- Prune unreachable objects and display what has been pruned to stdout:

  ```
  git prune --verbose
  ```

- Prune unreachable objects while showing progress:

  ```
  git prune --progress
  ```

git pull

Fetch branch from a remote repository and merge it to local repository.

More information: https://git-scm.com/docs/git-pull.

- Download changes from default remote repository and merge it:

```
g it pu ll
```

- Download changes from default remote repository and use fast forward:

```
g it pu ll - - reb ase
```

- Download changes from given remote repository and branch, then merge them into HEAD:

```
g it pu ll {{rem ote_nam e}} {{b ranch}}
```

git push

Push commits to a remote repository.

More information: https://git-scm.com/docs/git-push.

- Send local changes in the current branch to its remote counterpart:

```
git push
```

- Send local changes in a given branch to its remote counterpart:

```
git push {{remote_name}} {{local_branch}}
```

- Publish the current branch to a remote repository, setting the remote branch name:

```
git push {{remote_name}} -u {{remote_branch}}
```

- Send changes on all local branches to their counterparts in a given remote repository:

```
git push --all {{remote_name}}
```

- Delete a branch in a remote repository:

```
git push {{remote_name}} --delete {{remote_branch}}
```

- Remove remote branches that don't have a local counterpart:

```
git push --prune {{remote_name}}
```

- Publish tags that aren't yet in the remote repository:

```
git push --tags
```

git range-diff

Compare two commit ranges (e.g. two versions of a branch).

More information: https://git-scm.com/docs/git-range-diff.

- Diff the changes of two individual commits:

```
g it  rang e- diff  {{com m it_1 }}^ !  {{com m it_2 }}^ !
```

- Diff the changes of ours and theirs from their common ancestor, e.g. after an interactive rebase:

```
g it  rang e- diff  {{theirs}}. . . {{ou rs}}
```

- Diff the changes of two commit ranges, e.g. to check whether confl icts have been resolved appropriately when rebasing commits from `b ase1` to `b ase2`:

```
g it  rang e- diff  {{b ase1 }}. . {{rev 1 }} {{b ase2 }}. . {{rev 2 }}
```

git rebase

Reapply commits from one branch on top of another branch.

Commonly used to "move" an entire branch to another base, creating copies of the commits in the new location.

More information: https://git-scm.com/docs/git-rebase.

- Rebase the current branch on top of another specified branch:

  ```
  git rebase {{new_base_branch}}
  ```

- Start an interactive rebase, which allows the commits to be reordered, omitted, combined or modified:

  ```
  git rebase -i {{target_branch_or_commit_hash}}
  ```

- Continue a rebase that was interrupted by a merge failure, after editing conflicting files:

  ```
  git rebase --continue
  ```

- Continue a rebase that was paused due to merge conflicts, by skipping the conflicted commit:

  ```
  git rebase --skip
  ```

- Abort a rebase in progress (e.g. if it is interrupted by a merge conflict):

  ```
  git rebase --abort
  ```

- Move part of the current branch onto a new base, providing the old base to start from:

  ```
  git rebase --onto {{new_base}} {{old_base}}
  ```

- Reapply the last 5 commits in-place, stopping to allow them to be reordered, omitted, combined or modified:

  ```
  git rebase -i {{HEAD~5}}
  ```

- Auto-resolve any conflicts by favoring the working branch version (`theirs` keyword has reversed meaning in this case):

  ```
  git rebase -X theirs {{branch_name}}
  ```

git refl og

S how a log of changes to local references like H E AD, branches or tags.

More information: https://git-scm.com/docs/git-refl og

- S how the refl og for H E AD:

```
g it reflog
```

- S how the refl og for a given branch:

```
g it reflog {{b ranch_nam e}}
```

- S how only the 5 latest entries in the refl og:

```
g it reflog - n {{5 }}
```

git release

Create a G it tag for a release.

P art of `g i t - ex t ra s`

More information: https://github.com/tj /git-extras/blob/master/Commands.md# git-release.

- Create and push a release:

  ```
  g it release {{tag _nam e}}
  ```

- Create and push a signed release:

  ```
  g it release {{tag _nam e}} - s
  ```

- Create and push a release with a message:

  ```
  g it release {{{tag _nam e}}} - m " {{m essag e}}"
  ```

git remote

Manage set of tracked repositories ("remotes").

More information: https://git-scm.com/docs/git-remote.

- Show a list of existing remotes, their names and URL:

  ```
  git remote -v
  ```

- Show information about a remote:

  ```
  git remote show {{remote_name}}
  ```

- Add a remote:

  ```
  git remote add {{remote_name}} {{remote_url}}
  ```

- Change the URL of a remote (use `-add` to keep the existing URL):

  ```
  git remote set-url {{remote_name}} {{new_url}}
  ```

- Remove a remote:

  ```
  git remote remove {{remote_name}}
  ```

- Rename a remote:

  ```
  git remote rename {{old_name}} {{new_name}}
  ```

git rename-branch

Rename a Git branch.

Part of `git-extras`.

More information: https://github.com/tj/git-extras/blob/master/Commands.md#git-rename-branch.

- Rename the branch you are currently on:

  ```
  git rename-branch {{new_branch_name}}
  ```

- Rename a specific branch:

  ```
  git rename-branch {{old_branch_name}} {{new_branch_name}}
  ```

git rename-remote

Change remote for pulling and pushing.

Part of `git-extras`.

More information: https://github.com/tj/git-extras/blob/master/Commands.md#git-rename-remote.

- Change the upstream remote to origin:

```
git rename-remote {{upstream}} {{origin}}
```

git rename-tag

Rename a Git tag.

Part of `git-extras`.

More information: https://github.com/tj/git-extras/blob/master/Commands.md#git-rename-tag.

- Rename an existing Git tag locally and remotely:

```
git rename-tag {{old_tag_name}} {{new_tag_name}}
```

git repack

Pack unpacked objects in a Git repository.

More information: https://git-scm.com/docs/git-repack.

- Pack unpacked objects in the current directory:

  ```
  git repack
  ```

- Also remove redundant objects after packing:

  ```
  git repack -d
  ```

git repl

Git REPL (read-evaluate-print-loop) - an interactive Git shell.

Part of `git-extras`.

More information: https://github.com/tj/git-extras/blob/master/Commands.md#git-repl.

- Start an interactive Git shell:

  ```
  git repl
  ```

- Run a Git command while in the interactive Git shell:

  ```
  {{git_subcommand}} {{command_arguments}}
  ```

- Run an external (non-Git) command while in the interactive Git shell:

  ```
  ! {{command}} {{command_arguments}}
  ```

- Exit the interactive Git shell (or press Ctrl + D):

  ```
  exit
  ```

git replace

Create, list, and delete refs to replace obj ects.

More information: https://git-scm.com/docs/git-replace.

- R eplace any commit with a different one, leaving other commits unchanged:

```
g it replace {{ob j ect}} {{replacem ent}}
```

- Delete existing replace refs for the given obj ects:

```
g it replace - - delete {{ob j ect}}
```

- E dit an obj ect' s content interactively:

```
g it replace - - edit {{ob j ect}}
```

git req uest-pull

G enerate a req uest asking the upstream proj ect to pull changes into its tree.

More information: https://git-scm.com/docs/git-req uest-pull.

- P roduce a req uest summariz ing the changes between the v1 .1 release and a specified branch:

  ```
  g it req u est- pu ll {{v 1 . 1 }} {{https: //exam ple. com /proj ect}}
  {{b ranch_nam e}}
  ```

- P roduce a req uest summariz ing the changes between the v0 .1 release on the `foo` branch and the local `b ar` branch:

  ```
  g it req u est- pu ll {{v 0 . 1 }} {{https: //exam ple. com /proj ect}}
  {{foo: b ar}}
  ```

git reset-file

Revert a file to HEAD or a specific commit.

Part of `git-extras`.

More information: https://github.com/tj/git-extras/blob/master/Commands.md#git-reset-file.

- Reset a file to HEAD:

```
git reset-file {{path/to/file}}
```

- Reset a file to a specific commit:

```
git reset-file {{path/to/file}} {{commit_hash}}
```

git reset

Undo commits or unstage changes, by resetting the current Git HEAD to the specified state.

If a path is passed, it works as "unstage"; if a commit hash or branch is passed, it works as "uncommit".

More information: https://git-scm.com/docs/git-reset.

- Unstage everything:

```
git reset
```

- Unstage specific file(s):

```
git reset {{path/to/file(s)}}
```

- Interactively unstage portions of a file:

```
git reset --patch {{path/to/file}}
```

- Undo the last commit, keeping its changes (and any further uncommitted changes) in the filesystem:

```
git reset HEAD~
```

- Undo the last two commits, adding their changes to the index, i.e. staged for commit:

```
git reset --soft HEAD~2
```

- Discard any uncommitted changes, staged or not (for only unstaged changes, use `git checkout`):

```
git reset --hard
```

- Reset the repository to a given commit, discarding committed, staged and uncommitted changes since then:

```
git reset --hard {{commit}}
```

git restore

Restore working tree files. Requires Git version 2.23+.

See also `git checkout` and `git reset`.

More information: https://git-scm.com/docs/git-restore.

- Restore an unstaged file to the version of the current commit (HEAD):

  `git restore {{path/to/file}}`

- Restore an unstaged file to the version of a specific commit:

  `git restore --source {{commit}} {{path/to/file}}`

- Discard all unstaged changes to tracked files:

  `git restore :/`

- Unstage a file:

  `git restore --staged {{path/to/file}}`

- Unstage all files:

  `git restore --staged :/`

- Discard all changes to files, both staged and unstaged:

  `git restore --worktree --staged :/`

- Interactively select sections of files to restore:

  `git restore --patch`

git rev-list

List revisions (commits) in reverse chronological order.

More information: https://git-scm.com/docs/git-rev-list.

- List all commits on the current branch:

```
g it  rev - list  {{H E A D }}
```

- List commits more recent than a specific date, on a specific branch:

```
g it  rev - list  - - since= {{' 2 0 1 9 - 1 2 - 0 1  0 0 : 0 0 : 0 0 ' }}
{{b ranch_nam e}}
```

- List all merge commits on a specific commit:

```
g it  rev - list  - - m erg es  {{com m it}}
```

git rev-parse

Display metadata related to specific revisions.

More information: https://git-scm.com/docs/git-rev-parse.

- G et the commit hash of a branch:

  ```
  g it rev - parse {{b ranch_nam e}}
  ```

- G et the current branch name:

  ```
  g it rev - parse - - ab b rev - ref {{H E A D }}
  ```

- G et the absolute path to the root directory:

  ```
  g it rev - parse - - show - toplev el
  ```

git revert

Create new commits which reverse the effect of earlier ones.

More information: https://git-scm.com/docs/git-revert.

- Revert the most recent commit:

```
git revert {{@}}
```

- Revert the 5th last commit:

```
git revert HEAD~{{4}}
```

- Revert multiple commits:

```
git revert {{branch_name~5..branch_name~2}}
```

- Don't create new commits, just change the working tree:

```
git revert -n {{0c01a9..9a1743}}
```

git rm

Remove files from repository index and local filesystem.

More information: https://git-scm.com/docs/git-rm.

- Remove file from repository index and filesystem:

```
git rm {{file}}
```

- Remove directory:

```
git rm -r {{directory}}
```

- Remove file from repository index but keep it untouched locally:

```
git rm --cached {{file}}
```

git root

Print the root directory of the current Git repository.

Part of `git-extras`.

More information: https://github.com/tj/git-extras/blob/master/Commands.md#git-root.

- Print the absolute path of the current Git repository:

  ```
  git root
  ```

- Print the current working directory relative to the root of the current Git repository:

  ```
  git root --relative
  ```

git send-email

Send a collection of patches as emails.

Patches can be specified as files, directions, or a revision list.

More information: https://git-scm.com/docs/git-send-email.

- Send the last commit in the current branch:

```
git send-email -1
```

- Send a given commit:

```
git send-email -1 {{commit}}
```

- Send multiple (e.g. 10) commits in the current branch:

```
git send-email {{-10}}
```

- Send an introductory email message for the patch series:

```
git send-email -{{number_of_commits}} --compose
```

- Review and edit the email message for each patch you're about to send:

```
git send-email -{{number_of_commits}} --annotate
```

git setup

Create a Git repository in a specific directory and commit all files.

Part of `git-extras`.

More information: https://github.com/tj/git-extras/blob/master/Commands.md#git-setup.

- Create a Git repository in the current directory and commit all files:

  ```
  git setup
  ```

- Create a Git repository in a specific directory and commit all files:

  ```
  git setup {{path/to/directory}}
  ```

git shortlog

Summarizes the `git log` output.

More information: https://git-scm.com/docs/git-shortlog.

- View a summary of all the commits made, grouped alphabetically by author name:

  ```
  git shortlog
  ```

- View a summary of all the commits made, sorted by the number of commits made:

  ```
  git shortlog -n
  ```

- View a summary of all the commits made, grouped by the committer identities (name and email):

  ```
  git shortlog -c
  ```

- View a summary of the last 5 commits (i.e. specify a revision range):

  ```
  git shortlog HEAD~{{5}}..HEAD
  ```

- View all users, emails and the number of commits in the current branch:

  ```
  git shortlog -sne
  ```

- View all users, emails and the number of commits in all branches:

  ```
  git shortlog -sne --all
  ```

git show-branch

S how branches and their commits.

More information: https://git-scm.com/docs/git-show-branch.

- S how a summary of the latest commit on a branch:

```
g it show - b ranch {{b ranch_nam e| ref| com m it}}
```

- Compare commits in the history of multiple commits or branches:

```
g it show - b ranch {{b ranch_nam e| ref| com m it}}
```

- Compare all remote tracking branches:

```
g it show - b ranch - - rem otes
```

- Compare both local and remote tracking branches:

```
g it show - b ranch - - all
```

- List the latest commits in all branches:

```
g it show - b ranch - - all - - list
```

- Compare a given branch with the current branch:

```
g it show - b ranch - - cu rrent {{com m it| b ranch_nam e| ref}}
```

- Display the commit name instead of the relative name:

```
g it show - b ranch - - sha1 - nam e - - cu rrent {{cu rrent| b ranch_nam e
ref}}
```

- Keep going a given number of commits past the common ancestor:

```
g it show - b ranch - - m ore {{5 }} {{com m it| b ranch_nam e| ref}}
{{com m it| b ranch_nam e| ref}} {{. . . }}
```

git show-index

Show the packed archive index of a Git repository.

More information: https://git-scm.com/docs/git-show-index.

- Read an IDX file for a Git packfile and dump its contents to stdout:

```
git show-index {{path/to/file.idx}}
```

- Specify the hash algorithm for the index file (experimental):

```
git show-index --object-format={{sha1|sha256}} {{path/to/file}}
```

git show-ref

G it command for listing references.

More information: https://git-scm.com/docs/git-show-ref.

- S how all refs in the repository:

```
g it show - ref
```

- S how only heads references:

```
g it show - ref - - heads
```

- S how only tags references:

```
g it show - ref - - tag s
```

- V erify that a given reference exists:

```
g it show - ref - - v erify {{path/to/ref}}
```

git show-tree

Show a decorated tree graph with all branches of a Git repository, showing annotations.

Part of `git-extras`.

More information: https://github.com/tj/git-extras/blob/master/Commands.md#git-show-tree.

- Show a decorated tree graph for all branches annotated with tags and branch names:

  ```
  git show-tree
  ```

git show

Show various types of Git objects (commits, tags, etc.).

More information: https://git-scm.com/docs/git-show.

- Show information about the latest commit (hash, message, changes, and other metadata):

```
git show
```

- Show information about a given commit:

```
git show {{commit}}
```

- Show information about the commit associated with a given tag:

```
git show {{tag}}
```

- Show information about the 3rd commit from the HEAD of a branch:

```
git show {{branch}}~{{3}}
```

- Show a commit's message in a single line, suppressing the diff output:

```
git show --oneline -s {{commit}}
```

- Show only statistics (added/removed characters) about the changed files:

```
git show --stat {{commit}}
```

- Show only the list of added, renamed or deleted files:

```
git show --summary {{commit}}
```

- Show the contents of a file as it was at a given revision (e.g. branch, tag or commit):

```
git show {{revision}}:{{path/to/file}}
```

git siz er

Computes various G it repository siz e metrics and alerts you to any that might cause problems or inconvenience.

More information: https://github.com/github/git-siz er

- R eport only statistics that have a level of concern greater than 0 :

  ```
  g it sizer
  ```

- R eport all statistics:

  ```
  g it sizer - v
  ```

- S ee additional options:

  ```
  g it sizer - h
  ```

git stage

Add file contents to the staging area.

Synonym of `git add`

More information: https://git-scm.com/docs/git-stage.

- Add a file to the index:

  ```
  git stage {{path/to/file}}
  ```

- Add all files (tracked and untracked):

  ```
  git stage -A
  ```

- Only add already tracked files:

  ```
  git stage -u
  ```

- Also add ignored files:

  ```
  git stage -f
  ```

- Interactively stage parts of files:

  ```
  git stage -p
  ```

- Interactively stage parts of a given file:

  ```
  git stage -p {{path/to/file}}
  ```

- Interactively stage a file:

  ```
  git stage -i
  ```

git stash

Stash local Git changes in a temporary area.

More information: https://git-scm.com/docs/git-stash.

- Stash current changes, except new (untracked) files:

```
git stash [push -m {{optional_stash_message}}]
```

- Stash current changes, including new (untracked) files:

```
git stash -u
```

- Interactively select parts of changed files for stashing:

```
git stash -p
```

- List all stashes (shows stash name, related branch and message):

```
git stash list
```

- Apply a stash (default is the latest, named stash@{0}):

```
git stash apply {{optional_stash_name_or_commit}}
```

- Apply a stash (default is stash@{0}), and remove it from the stash list if applying doesn't cause conflicts:

```
git stash pop {{optional_stash_name}}
```

- Drop a stash (default is stash@{0}):

```
git stash drop {{optional_stash_name}}
```

- Drop all stashes:

```
git stash clear
```

git status

Show the changes to files in a Git repository.

Lists changed, added and deleted files compared to the currently checked-out commit.

More information: https://git-scm.com/docs/git-status.

- Show changed files which are not yet added for commit:

  ```
  git status
  ```

- Give output in [s]hort format:

  ```
  git status -s
  ```

- Don't show untracked files in the output:

  ```
  git status --untracked-files=no
  ```

- Show output in [s]hort format along with [b]ranch info:

  ```
  git status -sb
  ```

git stripspace

Read text (e.g. commit messages, notes, tags, and branch descriptions) from the standard input and clean it into the manner used by Git.

More information: https://git-scm.com/docs/git-stripspace.

- Trim whitespace from a file:

```
cat {{path/to/file}} | git stripspace
```

- Trim whitespace and Git comments from a file:

```
cat {{path/to/file}} | git stripspace --strip-comments
```

- Convert all lines in a file into Git comments:

```
git stripspace --comment-lines < {{path/to/file}}
```

git submodule

Inspects, updates and manages submodules.

More information: https://git-scm.com/docs/git-submodule.

- Install a repository's specified submodules:

```
g it su b m odu le u pdate - - init - - recu rsiv e
```

- Add a G it repository as a submodule:

```
g it su b m odu le add {{repository_u rl}}
```

- Add a G it repository as a submodule at the specified directory:

```
g it su b m odu le add {{repository_u rl}} {{path/to/directory}}
```

- U pdate every submodule to its latest commit:

```
g it su b m odu le foreach g it pu ll
```

git subtree

Tool to manage project dependencies as subprojects.

More information: https://manpages.debian.org/testing/git-man/git-subtree.1.en.html

- Add a Git repository as a subtree:

```
git subtree add --prefix={{path/to/directory/}} --squash
{{repository_url}} {{branch_name}}
```

- Update subtree repository to its latest commit:

```
git subtree pull --prefix={{path/to/directory/}}
{{repository_url}} {{branch_name}}
```

- Merge recent changes up to the latest subtree commit into the subtree:

```
git subtree merge --prefix={{path/to/directory/}} --squash
{{repository_url}} {{branch_name}}
```

- Push commits to a subtree repository:

```
git subtree push --prefix={{path/to/directory/}}
{{repository_url}} {{branch_name}}
```

- Extract a new project history from the history of a subtree:

```
git subtree split --prefix={{path/to/directory/}}
{{repository_url}} -b {{branch_name}}
```

git summary

Display information about a G it repository.

P art of `g i t - ex t ra s`

More information: [https://github.com/tj /git-extras/blob/master/Commands.md# git-summary](https://github.com/tj/git-extras/blob/master/Commands.md#git-summary).

- Display data about a G it repository:

  ```
  g it su m m ary
  ```

- Display data about a G it repository since a commit-ish:

  ```
  g it su m m ary {{com m it| b ranch_nam e| tag _nam e}}
  ```

- Display data about a G it repository, merging committers using different emails into 1 statistic for each author:

  ```
  g it su m m ary - - dedu p- b y- em ail
  ```

- Display data about a G it repository, showing the number of lines modified by each contributer:

  ```
  g it su m m ary - - line
  ```

git svn

Bidirectional operation between a Subversion repository and Git.

More information: https://git-scm.com/docs/git-svn.

- Clone an SVN repository:

```
git svn clone {{https://example.com/subversion_repo}} {{local_dir}}
```

- Clone a SVN repository starting at a given revision number:

```
git svn clone -r{{1234}}:HEAD {{https://svn.example.net/subversion/repo}} {{local_dir}}
```

- Update local clone from the remote SVN repository:

```
git svn rebase
```

- Fetch updates from the remote SVN repository without changing the Git HEAD:

```
git svn fetch
```

- Commit back to the SVN repository:

```
git svn dcommit
```

git switch

Switch between Git branches. Requires Git version 2.23+.

See also: `git checkout`

More information: https://git-scm.com/docs/git-switch.

- Switch to an existing branch:

  ```
  git switch {{branch_name}}
  ```

- Create a new branch and switch to it:

  ```
  git switch --create {{branch_name}}
  ```

- Create a new branch based on an existing commit and switch to it:

  ```
  git switch --create {{branch_name}} {{commit}}
  ```

- Switch to the previous branch:

  ```
  git switch -
  ```

- Switch to a branch and update all submodules to match:

  ```
  git switch --recurse-submodules {{branch_name}}
  ```

- Switch to a branch and automatically merge the current branch and any uncommitted changes into it:

  ```
  git switch --merge {{branch_name}}
  ```

git tag

Create, list, delete or verify tags.

A tag is a static reference to a specific commit.

More information: https://git-scm.com/docs/git-tag.

- List all tags:

```
g it tag
```

- Create a tag with the given name pointing to the current commit:

```
g it tag  {{tag _nam e}}
```

- Create a tag with the given name pointing to a given commit:

```
g it tag  {{tag _nam e}} {{com m it}}
```

- Create an annotated tag with the given message:

```
g it tag  {{tag _nam e}} - m  {{tag _m essag e}}
```

- Delete the tag with the given name:

```
g it tag  - d {{tag _nam e}}
```

- G et updated tags from upstream:

```
g it fetch - - tag s
```

- List all tags whose ancestors include a given commit:

```
g it tag  - - contains {{com m it}}
```

git touch

Create new files and add them to the index.

Part of `git-extras`.

More information: https://github.com/tj/git-extras/blob/master/Commands.md#git-touch.

- Create new files and add them to the index:

```
git touch {{path/to/file1 path/to/file2 ...}}
```

git undo

Undo recent commits.

Part of `git-extras`.

More information: https://github.com/tj/git-extras/blob/master/Commands.md#git-undo.

- Remove the most recent commit:

  ```
  git undo
  ```

- Remove a specific number of the most recent commits:

  ```
  git undo {{3}}
  ```

git update-index

Git command for manipulating the index.

More information: https://git-scm.com/docs/git-update-index.

- Pretend that a modified file is unchanged (`git status` will not show this as changed):

  `git update-index --skip-worktree {{path/to/modified_file}}`

git update-ref

G it command for creating, updating, and deleting G it refs.

More information: https://git-scm.com/docs/git-update-ref.

- Delete a ref, useful for soft resetting the first commit:

  ```
  g it  u pdate- ref  - d  {{H E A D }}
  ```

- U pdate ref with a message:

  ```
  g it  u pdate- ref  - m  {{m essag e}} {{H E A D }} {{4 e9 5 e0 5 }}
  ```

git var

Prints a Git logical variable's value.

See `git config`, which is preferred over `git var`.

More information: https://git-scm.com/docs/git-var.

- Print the value of a Git logical variable:

```
git var {{GIT_AUTHOR_IDENT|GIT_COMMITTER_IDENT|GIT_PAGER}}
```

- List all Git logical variables:

```
git var -l
```

git verify-commit

Check for G P G verification of commits.

If no commits are verified, nothing will be printed, regardless of options specified.

More information: https://git-scm.com/docs/git-verify-commit.

- Check commits for a G P G signature:

```
g it v erify- com m it {{com m it_hash1 optional_com m it_hash2 . . . }}
```

- Check commits for a G P G signature and show details of each commit:

```
g it v erify- com m it {{com m it_hash1 optional_com m it_hash2 . . . }}
- - v erb ose
```

- Check commits for a G P G signature and print the raw details:

```
g it v erify- com m it {{com m it_hash1 optional_com m it_hash2 . . . }}
- - raw
```

git verify-tag

Check for GPG verification of tags.

If a tag wasn't signed, an error will occur.

More information: https://git-scm.com/docs/git-verify-tag.

- Check tags for a GPG signature:

```
git verify-tag {{tag1 optional_tag2 ...}}
```

- Check tags for a GPG signature and show details for each tag:

```
git verify-tag {{tag1 optional_tag2 ...}} --verbose
```

- Check tags for a GPG signature and print the raw details:

```
git verify-tag {{tag1 optional_tag2 ...}} --raw
```

git whatchanged

Show what has changed with recent commits or files.

See also: `git log`

More information: https://git-scm.com/docs/git-whatchanged.

- Display logs and changes for recent commits:

  `git whatchanged`

- Display logs and changes for recent commits within the specified time frame:

  `git whatchanged --since="{{2 hours ago}}"`

- Display logs and changes for recent commits for specific files or directories:

  `git whatchanged {{path/to/file_or_directory}}`

git worktree

Manage multiple working trees attached to the same repository.

More information: https://git-scm.com/docs/git-worktree.

- Create a new directory with the specified branch checked out into it:

  ```
  g it w ork tree add {{path/to/directory}} {{b ranch}}
  ```

- Create a new directory with a new branch checked out into it:

  ```
  g it w ork tree add {{path/to/directory}} - b  {{new _b ranch}}
  ```

- List all the working directories attached to this repository:

  ```
  g it w ork tree list
  ```

- R emove a worktree (after deleting worktree directory):

  ```
  g it w ork tree pru ne
  ```

git

Distributed version control system.

More information: https://git-scm.com/.

- Check the Git version:

```
git --version
```

- Show general help:

```
git --help
```

- Show help on a Git subcommand (like `commit`, `log`, etc.):

```
git help {{subcommand}}
```

- Execute a Git subcommand:

```
git {{subcommand}}
```

- Execute a Git subcommand on a custom repository root path:

```
git -C {{path/to/repo}} {{subcommand}}
```

- Execute a Git subcommand with a given configuration set:

```
git -c '{{config.key}}={{value}}' {{subcommand}}
```

github-label-sync

A command-line interface for synchronising GitHub labels.

More information: https://npmjs.com/package/github-label-sync

- Synchronise labels using a local `labels.json` file:

  ```
  github-label-sync --access-token {{token}} {{repository_name}}
  ```

- Synchronise labels using a specific labels JSON file:

  ```
  github-label-sync --access-token {{token}} --labels {{url|path/to/json_file}} {{repository_name}}
  ```

- Perform a dry run instead of actually synchronising labels:

  ```
  github-label-sync --access-token {{token}} --dry-run {{repository_name}}
  ```

- Keep labels that aren't in `labels.json`:

  ```
  github-label-sync --access-token {{token}} --allow-added-labels {{repository_name}}
  ```

- Synchronise using the `GITHUB_ACCESS_TOKEN` environment variable:

  ```
  github-label-sync {{repository_name}}
  ```

gitk

A graphical Git repository browser.

More information: https://git-scm.com/docs/gitk.

- Show the repository browser for the current Git repository:

```
gitk
```

- Show repository browser for a specific file or directory:

```
gitk {{path/to/file_or_directory}}
```

- Show commits made since 1 week ago:

```
gitk --since="{{1 week ago}}"
```

- Show commits older than 1/1/2016:

```
gitk --until="{{1/1/2015}}"
```

- Show at most 100 changes in all branches:

```
gitk --max-count={{100}} --all
```

gitlab-ctl

CLI tool for managing the GitLab omnibus.

More information: https://docs.gitlab.com/omnibus/maintenance/.

- Display the status of every service:

  ```
  sudo gitlab-ctl status
  ```

- Display the status of a specific service:

  ```
  sudo gitlab-ctl status {{nginx}}
  ```

- Restart every service:

  ```
  sudo gitlab-ctl restart
  ```

- Restart a specific service:

  ```
  sudo gitlab-ctl restart {{nginx}}
  ```

- Display the logs of every service and keep reading until `Ctrl + C` is pressed:

  ```
  sudo gitlab-ctl tail
  ```

- Display the logs of a specific service:

  ```
  sudo gitlab-ctl tail {{nginx}}
  ```

gitlab-runner

CLI tool for managing GitLab runners.

More information: https://docs.gitlab.com/runner/.

- Register a runner:

```
sudo gitlab-runner register --url {{https://gitlab.example.com}} --registration-token {{token}} --name {{name}}
```

- Register a runner with a Docker executor:

```
sudo gitlab-runner register --url {{https://gitlab.example.com}} --registration-token {{token}} --name {{name}} --executor {{docker}}
```

- Unregister a runner:

```
sudo gitlab-runner unregister --name {{name}}
```

- Display the status of the runner service:

```
sudo gitlab-runner status
```

- Restart the runner service:

```
sudo gitlab-runner restart
```

- Check if the registered runners can connect to GitLab:

```
sudo gitlab-runner verify
```

gitlab

Ruby wrapper and CLI for the GitLab API.

More information: https://narkoz.github.io/gitlab/

- Create a new project:

  ```
  gitlab create_project {{project_name}}
  ```

- Get info about a specific commit:

  ```
  gitlab commit {{project_name}} {{commit_hash}}
  ```

- Get info about jobs in a CI pipeline:

  ```
  gitlab pipeline_jobs {{project_name}} {{pipeline_id}}
  ```

- Start a specific CI job:

  ```
  gitlab job_play {{project_name}} {{job_id}}
  ```

gitmoji

An interactive command-line tool for using emojis on commits.

More information: https://github.com/carloscuesta/gitmoji-cli

- Start the commit wizard:

  `gitmoji --commit`

- Initialize the git hook (so `gitmoji` will be run every time `git commit` is run):

  `gitmoji --init`

- Remove the git hook:

  `gitmoji --remove`

- List all available emojis and their descriptions:

  `gitmoji --list`

- Search emoji list for a list of keywords:

  `gitmoji --search {{keyword1}} {{keyword2}}`

- Update cached list of emojis from main repository:

  `gitmoji --update`

- Configure global preferences:

  `gitmoji --config`

gitsome

A terminal-based interface for GitHub, accessed via the `gh` command.

It also provides menu-style autocomplete suggestions for `git` commands.

More information: https://github.com/donnemartin/gitsome.

- Enter the gitsome shell (optional), to enable autocompletion and interactive help for Git (and gh) commands:

  ```
  gitsome
  ```

- Setup GitHub integration with the current account:

  ```
  gh config user
  ```

- List notifications for the current account (as would be seen in https://github.com/notifications):

  ```
  gh notifications
  ```

- List the current account's starred repos, filtered by a given search string:

  ```
  gh starred "{{python 3}}"
  ```

- View the recent activity feed of a given GitHub repository:

  ```
  gh feed {{tldr-pages/tldr}}
  ```

- View the recent activity feed for a given GitHub user, using the default pager (e.g. `less`):

  ```
  gh feed {{torvalds}} -p
  ```

gixy

Analyz e nginx configuration files.

More information: https://github.com/yandex/gixy.

- Analyz e nginx configuration (default path: `/etc/ng inx/ng inx. conf`):

  ```
  g ixy
  ```

- Analyz e nginx configuration but skip specific tests:

  ```
  g ixy - - sk ips {{http_splitting }}
  ```

- Analyz e nginx configuration with the specific severity level:

  ```
  g ixy {{- l| - ll| - lll}}
  ```

- Analyz e nginx configuration files on the specific path:

  ```
  g ixy {{path/to/config u ration_file_1 }} {{path/to/
  config u ration_file_2 }}
  ```

glab

GitLab CLI tool to help working with GitLab from the command-line.

More information: https://github.com/profclems/glab.

- Create a merge request:

  `glab mr create`

- List merge requests:

  `glab mr list`

- Create a new issue:

  `glab issue create`

- View and filter the current repository's open issues:

  `glab issue list`

- List pipelines:

  `glab pipeline list`

- Clone a repository into a specific directory:

  `glab repo clone {{user}}/{{repository}} {{directory}}`

glances

A cross-platform system monitoring tool.

More information: https://nicolargo.github.io/glances/.

- Run in terminal:

  ```
  glances
  ```

- Run in web server mode to show results in browser:

  ```
  glances -w
  ```

- Run in server mode to allow connections from other Glances clients:

  ```
  glances -s
  ```

- Connect to a Glances server:

  ```
  glances -c {{hostname}}
  ```

- Require a password in (web) server mode:

  ```
  glances -s --password
  ```

glib-compile-resources

Compiles resource files (e.g. images) into a binary resource bundle.

These may be linked into GTK applications using the GResource API.

More information: https://manned.org/glib-compile-resources.

- Compile resources referenced in `file.gresource.xml` to a .gresource binary:

  ```
  glib-compile-resources {{file.gresource.xml}}
  ```

- Compile resources referenced in `file.gresource.xml` to a C source file:

  ```
  glib-compile-resources --generate-source
  {{file.gresource.xml}}
  ```

- Compile resources in `file.gresource.xml` to a chosen target file, with `.c`, `.h` or `.gresource` extension:

  ```
  glib-compile-resources --generate --target={{file.ext}}
  {{file.gresource.xml}}
  ```

- Print a list of resource files referenced in `file.gresource.xml`:

  ```
  glib-compile-resources --generate-dependencies
  {{file.gresource.xml}}
  ```

glow

Render Markdown in the terminal.

More information: https://github.com/charmbracelet/glow.

- Run glow and select a file to view:

  ```
  glow
  ```

- Render a Markdown file to the terminal:

  ```
  glow {{path/to/file}}
  ```

- View a Markdown file using a paginator:

  ```
  glow -p {{path/to/file}}
  ```

- View a file from a URL:

  ```
  glow {{https://example.com/file.md}}
  ```

- View a GitHub/GitLab README:

  ```
  glow {{github.com/owner/repository}}
  ```

gml2gv

Convert a graph from **gml** to **gv** format.

Converters: **gml2gv**, **gv2gml**, **xgl2gv**, **gvpr**, **ph2gv**, **mm2gv**

More information: https://graphviz.org/pdf/gml2gv.1.pdf

- Convert a graph from `gml` to `gv` format:

  ```
  gml2gv -o {{output.gv}} {{input.gml}}
  ```

- Convert a graph using stdin and stdout:

  ```
  cat {{input.gml}} | gml2gv > {{output.gv}}
  ```

- Display help:

  ```
  gml2gv -?
  ```

gnomon

Utility to annotate console logging statements with timestamps and find slow processes.

More information: https://github.com/paypal/gnomon.

- Use UNIX (or DOS) pipes to pipe the stdout of any command through gnomon:

  ```
  {{npm test}} | gnomon
  ```

- Show number of seconds since the start of the process:

  ```
  {{npm test}} | gnomon --type=elapsed-total
  ```

- Show an absolute timestamp in UTC:

  ```
  {{npm test}} | gnomon --type=absolute
  ```

- Set a high threshold of 0.5 seconds for the elapsed time; exceeding which the timestamp will be colored bright red:

  ```
  {{npm test}} | gnomon --high {{0.5}}
  ```

- Set a medium threshold of 0.2 seconds (Timestamp will be colored bright yellow):

  ```
  {{npm test}} | gnomon --medium {{0.2}}
  ```

gnucash-cli

A command-line version of GnuCash.

More information: https://gnucash.org.

- Get quotes for currencies and stocks specified in a file and print them:

  ```
  gnucash-cli --quotes get {{path/to/file.gnucash}}
  ```

- Generate a financial report of a specific type, specified by `--name`:

  ```
  gnucash-cli --report run --name "{{Balance Sheet}}" {{path/to/file.gnucash}}
  ```

gnucash

P ersonal and small-business financial-accounting software.

More information: https://gnucash.org.

- Launch G nuCash and load the previously opened file:

  ```
  g nu cash
  ```

- Launch G nuCash and load the specified file:

  ```
  g nu cash {{path/to/file. g nu cash}}
  ```

- Launch G nuCash and load an empty file:

  ```
  g nu cash - - nofile
  ```

gnuplot

A graph plotter that outputs in several formats.

More information: http://www.gnuplot.info/.

- Start the interactive graph plotting shell:

```
gnuplot
```

- Plot the graph for the specified graph definition file:

```
gnuplot {{path/to/definition.plt}}
```

- Set the output format by executing a command before loading the definition file:

```
gnuplot -e "{{set output "path/to/filename.png" size 1024,768}}" {{path/to/definition.plt}}
```

- Persist the graph plot preview window after gnuplot exits:

```
gnuplot --persist {{path/to/definition.plt}}
```

go bug

Report a bug.

More information: https://golang.org/cmd/go/#hdr-Start_a_bug_report.

- Open a web page to start a bug report:

```
go bug
```

go build

Compile G o sources.

More information: https://golang.org/cmd/go/#_hdr-Compile_packages_and_dependencies

- Compile a file:

```
g o b u ild {{path/to/m ain. g o}}
```

- Compile, specifying the output filename:

```
g o b u ild - o {{path/to/b inary}} {{path/to/sou rce. g o}}
```

- Compile a package:

```
g o b u ild - o {{path/to/b inary}} {{path/to/pack ag e}}
```

- Compile a main package into an executable, with data race detection:

```
g o b u ild - race - o {{path/to/execu tab le}} {{path/to/m ain/
pack ag e}}
```

go clean

Remove object files and cached files.

More information: https://golang.org/cmd/go/#hdr-Remove_object_files_and_cached_files

- Print the remove commands instead of actually removing anything:

  ```
  go clean -n
  ```

- Delete the build cache:

  ```
  go clean -cache
  ```

- Delete all cached test results:

  ```
  go clean -testcache
  ```

- Delete the module cache:

  ```
  go clean -modcache
  ```

go doc

S how documentation for a package or symbol.

More information: https://golang.org/cmd/go/#_hdr-S_how_documentation_for_package_or_symbol

- S how documentation for the current package:

```
g o doc
```

- S how package documentation and exported symbols:

```
g o doc {{encoding /j son}}
```

- S how also documentation of symbols:

```
g o doc - all {{encoding /j son}}
```

- S how also sources:

```
g o doc - all - src {{encoding /j son}}
```

- S how a specific symbol:

```
g o doc - all - src {{encoding /j son. N u m b er}}
```

go env

Manage environment variables used by the Go toolchain.

More information: https://golang.org/cmd/go/#hdr-Print_Go_environment_information

- Show all environment variables:

```
go env
```

- Show a specific environment variable:

```
go env {{GOPATH}}
```

- Set an environment variable to a value:

```
go env -w {{GOBIN}}={{path/to/directory}}
```

- Reset an environment variable's value:

```
go env -u {{GOBIN}}
```

go fix

Update packages to use new APIs.

More information: https://golang.org/cmd/go/#hdr-Update_packages_to_use_new_APIs

- Update packages to use new APIs:

```
go fix {{packages}}
```

go generate

G enerate G o files by running commands within source files.

More information: https://golang.org/cmd/go/#_hdr-G enerate_ G o_ files_ by_ processing_. source

- G enerate G o files by running commands within source files:

```
g o g enerate
```

go list

List packages or modules.

More information: https://golang.org/cmd/go/#_hdr-List_packages_or_modules

- List packages:

```
go list ./...
```

- List standard packages:

```
go list std
```

- List packages in j son format:

```
go list -j son time net/http
```

- List module dependencies and available updates:

```
go list -m -u all
```

go mod

Module maintenance.

More information: https://golang.org/cmd/go/#_hdr-Module_maintenance

- Initialize new module in current directory:

  ```
  go mod init {{module name}}
  ```

- Download modules to local cache:

  ```
  go mod download
  ```

- Add missing and remove unused modules:

  ```
  go mod tidy
  ```

- Verify dependencies have expected content:

  ```
  go mod verify
  ```

- Copy sources of all dependencies into the vendor directory:

  ```
  go mod vendor
  ```

go test

Tests Go packages (files have to end with `_test.go`).

More information: https://golang.org/cmd/go/#hdr-Testing_flags

- Test the package found in the current directory:

  ```
  go test
  ```

- [v]erbosely test the package in the current directory:

  ```
  go test -v
  ```

- Test the packages in the current directory and all subdirectories (note the `./...`):

  ```
  go test -v ./...
  ```

- Test the package in the current directory and run all benchmarks:

  ```
  go test -v -bench .
  ```

- Test the package in the current directory and run all benchmarks for 50 seconds:

  ```
  go test -v -bench . -benchtime {{50s}}
  ```

go version

Print Go version.

More information: https://golang.org/cmd/go/#hdr-Print_Go_version

- Print Go version:

  ```
  go version
  ```

- Print the Go version used to build the named executable file:

  ```
  go version {{path/to/executable}}
  ```

go

Tool for managing go source code.

More information: https://golang.org.

- Download and install a package, specified by its import path:

  `go get {{package_path}}`

- Compile and run a source file (it has to contain a `main` package):

  `go run {{file}}.go`

- Compile a source file into a named executable:

  `go build -o {{executable}} {{file}}.go`

- Compile the package present in the current directory:

  `go build`

- Execute all test cases of the current package (files have to end with `_test.go`):

  `go test`

- Compile and install the current package:

  `go install`

- Initialize a new module in the current directory:

  `go mod init {{module_name}}`

gobuster

Brute-forces hidden paths on web servers and more.

More information: https://github.com/OJ/gobuster

- Discover directories and files that match in the wordlist:

```
gobuster dir --url {{https://example.com/}} --wordlist {{path/to/file}}
```

- Discover subdomains:

```
gobuster dns --domain {{example.com}} --wordlist {{path/to/file}}
```

- Discover Amazon S3 buckets:

```
gobuster s3 --wordlist {{path/to/file}}
```

- Discover other virtual hosts on the server:

```
gobuster vhost --url {{https://example.com/}} --wordlist {{path/to/file}}
```

- Fuzz the value of a parameter:

```
gobuster fuzz --url {{https://example.com/?parameter=FUZZ}} --wordlist {{path/to/file}}
```

- Fuzz the name of a parameter:

```
gobuster fuzz --url {{https://example.com/?FUZZ=value}} --wordlist {{path/to/file}}
```

gocryptfs

E ncrypted overlay filesystem written in G o.

More information: https://github.com/rfj akob/gocryptfs

- Initializ e an encrypted filesystem:

  ```
  g ocryptfs - init {{path/to/cipher_dir}}
  ```

- Mount an encrypted filesystem:

  ```
  g ocryptfs {{path/to/cipher_dir}} {{path/to/m ou nt_point}}
  ```

- Mount with the explicit master key instead of password:

  ```
  g ocryptfs - - m asterk ey {{path/to/cipher_dir}} {{path/to/
  m ou nt_point}}
  ```

- Change the password:

  ```
  g ocryptfs - - passw d {{path/to/cipher_dir}}
  ```

- Make an encrypted snapshot of a plain directory:

  ```
  g ocryptfs - - rev erse {{path/to/plain_dir}} {{path/to/
  cipher_dir}}
  ```

godoc

Show documentation for go packages.

More information: https://godoc.org/.

- Display help for package "fmt":

  ```
  godoc {{fmt}}
  ```

- Display help for the function "Printf" of "fmt" package:

  ```
  godoc {{fmt}} {{Printf}}
  ```

- Serve documentation as a web server on port 6060:

  ```
  godoc -http=:{{6060}}
  ```

- Create an index file:

  ```
  godoc -write_index -index_files={{path/to/file}}
  ```

- Use the given index file to search the docs:

  ```
  godoc -http=:{{6060}} -index -index_files={{path/to/file}}
  ```

godot

An open source 2D and 3D game engine.

More information: https://godotengine.org/.

- Run a project if the current directory contains a `project.godot` file, otherwise open the project manager:

  ```
  godot
  ```

- Edit a project (the current directory must contain a `project.godot` file):

  ```
  godot -e
  ```

- Open the project manager even if the current directory contains a `project.godot` file:

  ```
  godot -p
  ```

- Export a project for a given export preset (the preset must be defined in the project):

  ```
  godot --export {{preset}} {{output_path}}
  ```

- Execute a standalone GDScript file (the script must inherit from `SceneTree` or `MainLoop`):

  ```
  godot -s {{script.gd}}
  ```

gofmt

Tool for formatting Go source code.

More information: https://golang.org/cmd/gofmt/.

- Format a file and display the result to the console:

  ```
  gofmt {{source.go}}
  ```

- Format a file, overwriting the original file in-place:

  ```
  gofmt -w {{source.go}}
  ```

- Format a file, and then simplify the code, overwriting the original file:

  ```
  gofmt -s -w {{source.go}}
  ```

- Print all (including spurious) errors:

  ```
  gofmt -e {{source.go}}
  ```

goimports

U pdates G o import lines, adding missing ones and removing unreferenced ones.

More information: https://godoc.org/golang.org/x/tools/cmd/goimports.

- Display the completed import source file:

  ```
  g oim ports {{file}}. g o
  ```

- Write the result back to the source file instead of the standard output:

  ```
  g oim ports - w  {{file}}. g o
  ```

- Display diffs and write the result back to the source file:

  ```
  g oim ports - w  - d {{file}}. g o
  ```

- S et the import prefix string after 3 rd-party packages (comma-separated list):

  ```
  g oim ports - local {{path/to/pack ag e}} {{file}}. g o
  ```

googler

Search Google from command-line.

More information: https://github.com/jarun/googler

- Search Google for a keyword:

  ```
  google {{keyword}}
  ```

- Search Google and open the first result in web browser:

  ```
  google -j {{keyword}}
  ```

- Show N search results (default 10):

  ```
  google -n {{N}} {{keyword}}
  ```

- Disable automatic spelling correction:

  ```
  google -x {{keyword}}
  ```

- Search one site for a keyword:

  ```
  google -w {{site}} {{keyword}}
  ```

- Show Google search result in JSON format:

  ```
  google --json {{keyword}}
  ```

- Perform in-place self-upgrade:

  ```
  google -u
  ```

- For more help in interactive mode:

  ```
  ?
  ```

gopass

Standard Unix Password Manager for Teams. Written in Go.

More information: https://www.gopass.pw.

- Initialise the configuration settings:

```
gopass init
```

- Create a new entry:

```
gopass new
```

- Show all stores:

```
gopass mounts
```

- Mount a shared Git store:

```
gopass mounts add {{store_name}} {{git_repo_url}}
```

- Search interactively using a keyword:

```
gopass show -f {{keyword}}
```

- Search using a keyword:

```
gopass find {{keyword}}
```

- Sync all mounted stores:

```
gopass sync
```

- Show a particular password entry:

```
gopass {{store_name|path/to/directory|email@email.com}}
```

gops

CLI tool which lists and diagnoses G o processes currently running on your system.

More information: https://github.com/google/gops.

- P rint all go processes running locally:

  `g ops`

- P rint more information about a process:

  `g ops {{pid}}`

- Display a process tree:

  `g ops tree`

- P rint the current stack trace from a target program:

  `g ops stack {{pid| addr}}`

- P rint the current runtime memory statistics:

  `g ops m em stats {{pid| addr}}`

goreload

Live reload utility for Go programs.

More information: https://github.com/acoshift/goreload.

- Set the name of the binary file to watch (defaults to `.goreload`):

  `goreload -b {{path/to/binary}} {{file}}.go`

- Set a custom log prefix (defaults to `goreload`):

  `goreload --log-prefix {{prefix}} {{file}}.go`

- Reload whenever any file changes:

  `goreload --all`

gotty

Share your terminal as a web application.

More information: https://github.com/yudai/gotty.

- Share result of command:

  `gotty {{command}}`

- Share with write permission:

  `gotty -w {{shell}}`

- Share with credential (Basic Auth):

  `gotty -w -c {{username}}:{{password}} {{shell}}`

gource

Renders an animated tree diagram of Git, SVN, Mercurial and Bazaar repositories.

It shows files and directories being created, modified or removed over time.

More information: https://gource.io.

- Run gource in a directory (if it isn't the repository's root directory, the root is sought up from there):

  ```
  gource {{path/to/repository}}
  ```

- Run gource in the current directory, with a custom output resolution:

  ```
  gource -{{width}}x{{height}}
  ```

- Set a custom time scale for the animation:

  ```
  gource -c {{time_scale_multiplier}}
  ```

- Set how long each day should be in the animation (this combines with -c, if provided):

  ```
  gource -s {{seconds}}
  ```

- Set fullscreen mode and a custom background color:

  ```
  gource -f -b {{hex_color_code}}
  ```

- Set a title for the animation:

  ```
  gource --title {{title}}
  ```

gox

A tool for cross-compiling G o programs.

More information: https://github.com/mitchellh/gox.

- Compile G o program in the current directory for all operating systems and architecture combinations:

  ```
  g ox
  ```

- Download and compile a G o program from a remote U R L:

  ```
  g ox {{u rl_1 }} {{u rl_2 }}
  ```

- Compile current directory for a particular operating system:

  ```
  g ox - os= " {{os}}"
  ```

- Compile current directory for a single operating system and architecture combination:

  ```
  g ox - osarch= " {{os}}/{{arch}}"
  ```

gpg-zip

Encrypt files and directories in an archive using GPG.

More information: https://www.gnupg.org/documentation/manuals/gnupg/gpg_002dzip.html

- Encrypt a directory into `archive.gpg` using a passphrase:

  ```
  gpg-zip --symmetric --output {{archive.gpg}} {{path/to/directory}}
  ```

- Decrypt `archive.gpg` into a directory of the same name:

  ```
  gpg-zip --decrypt {{path/to/archive.gpg}}
  ```

- List the contents of the encrypted `archive.gpg`

  ```
  gpg-zip --list-archive {{path/to/archive.gpg}}
  ```

gpg

GNU Privacy Guard.

See `gpg 2` for GNU Privacy Guard 2.

More information: https://gnupg.org.

- Create a GPG public and private key interactively:

  `gpg --full-generate-key`

- Sign `doc.txt` without encryption (writes output to `doc.txt.asc`):

  `gpg --clearsign {{doc.txt}}`

- Encrypt `doc.txt` for alice@example.com (output to `doc.txt.gpg`):

  `gpg --encrypt --recipient {{alice@example.com}} {{doc.txt}}`

- Encrypt `doc.txt` with only a passphrase (output to `doc.txt.gpg`):

  `gpg --symmetric {{doc.txt}}`

- Decrypt `doc.txt.gpg` (output to stdout):

  `gpg --decrypt {{doc.txt.gpg}}`

- Import a public key:

  `gpg --import {{public.gpg}}`

- Export public key for alice@example.com (output to stdout):

  `gpg --export --armor {{alice@example.com}}`

- Export private key for alice@example.com (output to stdout):

  `gpg --export-secret-keys --armor {{alice@example.com}}`

gpg2

GNU Privacy Guard 2.

See `gpg` for GNU Privacy Guard 1.

More information: https://docs.releng.linuxfoundation.org/en/latest/gpg.html.

- List imported keys:

  ```
  gpg2 --list-keys
  ```

- Encrypt a specified file for a specified recipient, writing the output to a new file with `.gpg` appended:

  ```
  gpg2 --encrypt --recipient {{alice@example.com}} {{path/to/doc.txt}}
  ```

- Encrypt a specified file with only a passphrase, writing the output to a new file with `.gpg` appended:

  ```
  gpg2 --symmetric {{path/to/doc.txt}}
  ```

- Decrypt a specified file, writing the result to the standard output:

  ```
  gpg2 --decrypt {{path/to/doc.txt.gpg}}
  ```

- Import a public key:

  ```
  gpg2 --import {{path/to/public_key.gpg}}
  ```

- Export the public key of a specified email address to the standard output:

  ```
  gpg2 --export --armor {{alice@example.com}}
  ```

- Export the private key with a specified email address to the standard output:

  ```
  gpg2 --export-secret-keys --armor {{alice@example.com}}
  ```

gpgv

Verify OpenPGP signatures.

More information: https://www.gnupg.org/documentation/manuals/gnupg/gpgv.html.

- Verify a signed file:

```
gpgv {{path/to/file}}
```

- Verify a signed file using a detached signature:

```
gpgv {{path/to/signature}} {{path/to/file}}
```

- Add a file to the list of keyrings (a single exported key also counts as a keyring):

```
gpgv --keyring {{./alice.keyring}} {{path/to/signature}} {{path/to/file}}
```

gradle

Gradle is an open source build automation system.

More information: https://gradle.org.

- Compile a package:

```
gradle build
```

- Exclude test task:

```
gradle build -x {{test}}
```

- Run in offline mode to prevent Gradle from accessing the network during builds:

```
gradle build --offline
```

- Clear the build directory:

```
gradle clean
```

- Build an Android Package (APK) in release mode:

```
gradle assembleRelease
```

- List the main tasks:

```
gradle tasks
```

- List all the tasks:

```
gradle tasks --all
```

graphml2gv

Convert a graph from **graphml** to **gv** format.

Converters: **graphml2gv**, **gv2graphml**, **xgl2gv**, **graphml2xgl** & **mm2gv**

More information: https://graphviz.org/pdf/graphml2gv.1.pdf

- Convert a graph from `graphml` to `gv` format:

  ```
  graphml2gv -o {{output.gv}} {{input.graphml}}
  ```

- Convert a graph using stdin and stdout:

  ```
  cat {{input.graphml}} | graphml2gv > {{output.gv}}
  ```

- Display help:

  ```
  graphml2gv -?
  ```

grep

Find patterns in files using regular expressions.

More information: https://www.gnu.org/software/grep/manual/grep.html.

- Search for a pattern within a file:

```
grep "{{search_pattern}}" {{path/to/file}}
```

- Search for an exact string (disables regular expressions):

```
grep --fixed-strings "{{exact_string}}" {{path/to/file}}
```

- Search for a pattern in all files recursively in a directory, showing line numbers of matches, ignoring binary files:

```
grep --recursive --line-number --binary-files={{without-match}} "{{search_pattern}}" {{path/to/directory}}
```

- Use extended regular expressions (supports `?`, `+`, `{}`, `()` and `|`), in case-insensitive mode:

```
grep --extended-regexp --ignore-case "{{search_pattern}}" {{path/to/file}}
```

- Print 3 lines of context around, before, or after each match:

```
grep --{{context|before-context|after-context}}={{3}} "{{search_pattern}}" {{path/to/file}}
```

- Print file name and line number for each match:

```
grep --with-filename --line-number "{{search_pattern}}" {{path/to/file}}
```

- Search for lines matching a pattern, printing only the matched text:

```
grep --only-matching "{{search_pattern}}" {{path/to/file}}
```

- Search stdin for lines that do not match a pattern:

```
cat {{path/to/file}} | grep --invert-match "{{search_pattern}}"
```

grex

Simple command-line tool to generate regular expressions.

More information: https://github.com/pemistahl/grex.

- Generate a simple regular expression:

```
grex {{space_separated_string s}}
```

- Generate a case-insensitive regular expression:

```
grex -i {{space_separated_string s}}
```

- Replace digits with '\d':

```
grex -d {{space_separated_string s}}
```

- Replace Unicode word character with '\w':

```
grex -w {{space_separated_string s}}
```

- Replace spaces with '\s':

```
grex -s {{space_separated_string s}}
```

- Add {min,max} quantifier representation for repeating sub-strings:

```
grex -r {{space_separated_string s}}
```

groff

Typesetting program that reads plain text mixed with formatting commands and produces formatted output.

It is the GNU replacement for the `troff` and `nroff` Unix commands for text formatting.

More information: https://www.gnu.org/software/groff.

- Render a man page as plain text, and display the result:

```
groff -man -Tutf8 {{manpage.1}}
```

- Render a man page using the ASCII output device, and display it using a pager:

```
groff -man -Tascii {{manpage.1}} | less
```

- Render a man page into an HTML file:

```
groff -man -Thtml {{manpage.1}} > {{page.html}}
```

- Process a roff file using the `tbl` and `pic` preprocessors, and the `me` macro set:

```
groff -t -p -me -Tutf8 {{foo.me}}
```

- Run a `groff` command with preprocessor and macro options guessed by the `grog` utility:

```
eval "$(grog -Tutf8 {{foo.me}})"
```

groups

Print group memberships for a user.

More information: https://www.gnu.org/software/coreutils/groups.

- Print group memberships for the current user:

  ```
  groups
  ```

- Print group memberships for a specific user:

  ```
  groups {{username}}
  ```

- Print group memberships for a list of users:

  ```
  groups {{username1}} {{username2}} {{username3}}
  ```

grpcurl

Like cURL, but for gRPC: CLI tool for interacting with gRPC servers.

More information: https://github.com/fullstorydev/grpcurl.

- Send an empty request:

```
grpcurl {{grpc.server.com:443}} {{my.custom.server.ServiceMethod}}
```

- Send a request with a header and a body:

```
grpcurl -H "{{Authorization: Bearer $token}}" -d {{'{"foo":"bar"}'}} {{grpc.server.com:443}} {{my.custom.server.ServiceMethod}}
```

- List all services exposed by a server:

```
grpcurl {{grpc.server.com:443}} list
```

- List all methods in a particular service:

```
grpcurl {{grpc.server.com:443}} list {{my.custom.server.Service}}
```

grumphp

A P H P Composer plugin that enables source code q uality checks.

More information: https://github.com/phpro/grumphp.

- R egister the G it hooks:

  ```
  g ru m php g it: init
  ```

- T rigger the pre-commit hook manually:

  ```
  g ru m php g it: pre- com m it
  ```

- Check every versioned file:

  ```
  g ru m php ru n
  ```

grunt

A JavaScript task runner for automating processes.

More information: https://github.com/gruntjs/grunt-cli

- Run the default task process:

  ```
  grunt
  ```

- Run one or more specific space-separated task(s):

  ```
  grunt {{task_name}}
  ```

- Specify an alternative configuration file:

  ```
  grunt --gruntfile {{path/to/file}}
  ```

- Specify an alternative base path for relative files:

  ```
  grunt --base {{path/to/directory}}
  ```

- Specify an additional directory to scan for tasks in:

  ```
  grunt --tasks {{path/to/directory}}
  ```

- Perform a dry-run without writing any files:

  ```
  grunt --no-write
  ```

- List all available options:

  ```
  grunt --help
  ```

gtop

System monitoring dashboard for the terminal.

More information: https://github.com/aksakalli/gtop.

- Show the system stats dashboard:

  ```
  gtop
  ```

- Sort by CPU usage:

  ```
  c
  ```

- Sort by memory usage:

  ```
  m
  ```

guacd

Apache Guacamole proxy daemon.

Support loader for client plugins to interface between the Guacamole protocol and any arbitrary remote desktop protocol (e.g. RDP, VNC, Other).

More information: https://guacamole.apache.org/.

- Bind to a specific port on localhost:

  ```
  guacd -b {{127.0.0.1}} -l {{4823}}
  ```

- Start in debug mode, keeping the process in the foreground:

  ```
  guacd -f -L {{debug}}
  ```

- Start with TLS support:

  ```
  guacd -C {{my-cert.crt}} -K {{my-key.pem}}
  ```

- Write the PID to a file:

  ```
  guacd -p {{path/to/file.pid}}
  ```

guetzli

JPEG image compression utility.

More information: https://github.com/google/guetzli

- Compress a JPEG image:

```
guetzli {{input.jpg}} {{output.jpg}}
```

- Create compressed JPEG image from PNG image:

```
guetzli {{input.png}} {{output.jpg}}
```

- Compress a JPEG image with desired visual quality (84-100):

```
guetzli --quality {{quality_value}} {{input.jpg}} {{output.jpg}}
```

guile

Guile Scheme interpreter.

More information: https://www.gnu.org/software/guile.

- Start the Guile Scheme REPL:

```
guile
```

- Execute the script in a given Scheme file:

```
guile {{script.scm}}
```

- Execute a Scheme expression:

```
guile -c "{{expression}}"
```

- Listen on a port or a Unix domain socket (the default is port 37146) for remote REPL connections:

```
guile --listen={{port_or_socket}}
```

gulp

JavaScript task runner and streaming build system.

Tasks are defined within gulpfile.js at the project root.

More information: https://github.com/gulpjs/gulp-cli

- Run the default task:

  ```
  gulp
  ```

- Run individual tasks:

  ```
  gulp {{task}} {{othertask}}
  ```

gunicorn

Python WSGI HTTPS server.

More information: https://gunicorn.org/.

- Run Python web app:

```
gunicorn {{image path: app_object}}
```

- Listen on port 8080 on localhost:

```
gunicorn --bind {{localhost}}:{{8080}}
{{image path: app_object}}
```

- Turn on live reload:

```
gunicorn --reload {{image path: app_object}}
```

- Use 4 worker processes for handling requests:

```
gunicorn --workers {{4}} {{image path: app_object}}
```

- Use 4 worker threads for handling requests:

```
gunicorn --threads {{4}} {{image path: app_object}}
```

- Run app over HTTPS:

```
gunicorn --certfile {{cert.pem}} --keyfile {{key.pem}}
{{image path: app_object}}
```

gunzip

Extract file(s) from a gzip (.gz) archive.

More information: https://manned.org/gunzip

- Extract a file from an archive, replacing the original file if it exists:

  ```
  gunzip {{archive.tar.gz}}
  ```

- Extract a file to a target destination:

  ```
  gunzip --stdout {{archive.tar.gz}} > {{archive.tar}}
  ```

- Extract a file and keep the archive file:

  ```
  gunzip --keep {{archive.tar.gz}}
  ```

- List the contents of a compressed file:

  ```
  gunzip --list {{file.txt.gz}}
  ```

gv2gml

Convert a graph from **gv** to **gml** format.

Converters: **gml2gv**, **gv2gml**, **xgl2gv**, **gvpr**, **ph2gml**, & **mm2gv**

More information: https://graphviz.org/pdf/gml2gv.1.pdf

- Convert a graph from `gv` to `gml` format:

  ```
  gv2gml -o {{output.gml}} {{input.gv}}
  ```

- Convert a graph using stdin and stdout:

  ```
  cat {{input.gv}} | gv2gml > {{output.gml}}
  ```

- Display help:

  ```
  gv2gml -?
  ```

gv2gxl

Convert a graph from **gv** to **gxl** format.

Converters: **gml2gv**, **gv2gml**, **gxl2gv**, **dot2gxl**, & **gml2gxl**

More information: https://graphviz.org/pdf/gxl2gv.1.pdf

- Convert a graph from `gv` to `gxl` format:

  ```
  gv2gxl -o {{output.gxl}} {{input.gv}}
  ```

- Convert a graph using stdin and stdout:

  ```
  cat {{input.gv}} | gv2gxl > {{output.gxl}}
  ```

- Display help:

  ```
  gv2gxl -?
  ```

gxl2gv

Convert a graph from **gxl** to **gv** format.

Converters: **gml2gv**, **gv2gml**, **gxl2gv**, **dot2gv**, **mm2gv**

More information: https://graphviz.org/pdf/gxl2gv.1.pdf

- Convert a graph from `gxl` to `gv` format:

  ```
  gxl2gv -o {{output.gv}} {{input.gxl}}
  ```

- Convert a graph using stdin and stdout:

  ```
  cat {{input.gxl}} | gxl2gv > {{output.gv}}
  ```

- Display help:

  ```
  gxl2gv -?
  ```

gz ip

Compress/uncompress files with gz ip compression (LZ 7 7).

More information: https://www.gnu.org/software/gz ip/manual/gz ip.html

- Compress a file, replacing it with a gz ipped compressed version:

```
g zip {{file. ext}}
```

- Decompress a file, replacing it with the original uncompressed version:

```
g zip - d {{file. ext}}. g z
```

- Compress a file, keeping the original file:

```
g zip - - k eep {{file. ext}}
```

- Compress a file specifying the output filename:

```
g zip - c {{file. ext}} >  {{com pressed_file. ext. g z}}
```

- Decompress a gz ipped file specifying the output filename:

```
g zip - c - d {{file. ext}}. g z >  {{u ncom pressed_file. ext}}
```

- S pecify the compression level. 1 = F astest (Worst), 9 = S lowest (B est), Default level is 6 :

```
g zip - 9  - c {{file. ext}} >  {{com pressed_file. ext. g z}}
```

hadolint

Dockerfile linter.

More information: https://github.com/hadolint/hadolint.

- Lint a Dockerfile:

```
hadolint {{path/to/D ock erfile}}
```

- Lint a Dockerfile, displaying the output in J S O N format:

```
hadolint - - form at {{j son}} {{path/to/D ock erfile}}
```

- Lint a Dockerfile, displaying the output in a specific format:

```
hadolint - - form at {{tty| j son| check style| codeclim ate| codacy}}
{{path/to/D ock erfile}}
```

- Lint a Dockerfile ignoring specific rules:

```
hadolint - - ig nore {{D L 3 0 0 6 }} - - ig nore {{D L 3 0 0 8 }} {{path/to
D ock erfile}}
```

- Lint multiple Dockerfiles using specific trusted registries:

```
hadolint - - tru sted- reg istry {{dock er. io}} - - tru sted- reg istry
{{exam ple. com }}: {{5 0 0 0 }} {{path/to/D ock erfile}} {{path/to/
another/D ock erfile}}
```

hakyll-init

Generate a new Hakyll sample blog.

More information: https://github.com/jaspervdj/hakyll-init.

- Generate a new Hakyll sample blog:

  `hakyll-init {{path/to/directory}}`

- Show help for `hakyll-init`:

  `hakyll-init --help`

handbrakecli

Command-line interface to the HandBrake video conversion tool.

More information: https://handbrake.fr/.

- Convert a video file to MKV (AAC 160 kbit audio and x264 CRF 20 video):

```
handbrakecli -i {{input.avi}} -o {{output.mkv}} -e x264 -q
-B 160
```

- Resize a video file to 320x40:

```
handbrakecli -i {{input.mp4}} -o {{output.mp4}} -w 320 -l
```

- List available presets:

```
handbrakecli --preset-list
```

- Convert an AVI video to MP4 using the Android preset:

```
handbrakecli --preset="Android" -i {{input.ext}} -o
{{output.mp4}}
```

hangups

Third party command-line client for Google Hangouts.

More information: https://github.com/tdryer/hangups.

- Start hangups:

  ```
  hangups
  ```

- View troubleshooting information and help:

  ```
  hangups -h
  ```

- Set a refresh token for hangups:

  ```
  hangups --token-path {{path/to/token}}
  ```

haxelib

H axe Library Manager.

More information: https://lib.haxe.org/.

- S earch for a H axe library:

  ```
  haxelib search {{k eyw ord}}
  ```

- Install a H axe library:

  ```
  haxelib install {{lib nam e}}
  ```

- U pgrade all installed H axe libraries:

  ```
  haxelib u pg rade
  ```

- Install the development version of a library from a G it repository:

  ```
  haxelib g it {{lib nam e}} {{g it_u rl}}
  ```

head

Output the first part of files.

More information: https://www.gnu.org/software/coreutils/head.

- Output the first few lines of a file:

  ```
  head -n {{count_of_lines}} {{filename}}
  ```

- Output the first few bytes of a file:

  ```
  head -c {{size_in_bytes}} {{filename}}
  ```

- Output everything but the last few lines of a file:

  ```
  head -n -{{count_of_lines}} {{filename}}
  ```

- Output everything but the last few bytes of a file:

  ```
  head -c -{{size_in_bytes}} {{filename}}
  ```

helm

H elm is a package manager for Kubernetes.

More information: https://helm.sh/.

- Create a helm chart:

  ```
  helm create {{chart_nam e}}
  ```

- Add a new helm repository:

  ```
  helm repo add {{repo_nam e}}
  ```

- List helm repositories:

  ```
  helm repo list
  ```

- U pdate helm repositories:

  ```
  helm repo u pdate
  ```

- Delete a helm repository:

  ```
  helm repo rem ov e {{repo_nam e}}
  ```

- Install a helm chart:

  ```
  helm install {{repo_nam e}}/{{chart_nam e}}
  ```

- Download helm chart as a tar archive:

  ```
  helm g et {{chart_release_nam e}}
  ```

- U pdate helm dependencies:

  ```
  helm dependency u pdate
  ```

help2man

Produce simple man pages from an executable' `-s-help` and `--version` output.

More information: https://www.gnu.org/software/help2man

- Generate a man page for an executable:

  `help2man {{executable}}`

- Specify the "name" paragraph in the man page:

  `help2man {{executable}} --name {{name}}`

- Specify the section for the man page (defaults to 1):

  `help2man {{executable}} --section {{section}}`

- Output to a file instead of stdout:

  `help2man {{executable}} --output {{path/to/file}}`

- Display detailed help:

  `help2man --help`

heroku

Create and manage Heroku apps from the command-line.

More information: https://www.heroku.com/.

- Log in to your heroku account:

  ```
  heroku login
  ```

- Create a heroku app:

  ```
  heroku create
  ```

- Show logs for an app:

  ```
  heroku logs --app {{app_name}}
  ```

- Run a one-off process inside a dyno (Heroku virtual machine):

  ```
  heroku run {{process_name}} --app {{app_name}}
  ```

- List dynos (Heroku virtual machines) for an app:

  ```
  heroku ps --app {{app_name}}
  ```

- Permanently destroy an app:

  ```
  heroku destroy --app {{app_name}}
  ```

hexo

A fast, simple & powerful blog framework.

More information: https://hexo.io/.

- Initialize a website:

  ```
  hexo init {{path/to/directory}}
  ```

- Create a new article:

  ```
  hexo new {{layout}} {{title}}
  ```

- Generate static files:

  ```
  hexo generate
  ```

- Start a local server:

  ```
  hexo server
  ```

- Deploy the website:

  ```
  hexo deploy
  ```

- Clean the cache file (`db.json`) and generated files (`public/`):

  ```
  hexo clean
  ```

hexyl

A simple hex viewer for the terminal. Uses colored output to distinguish different categories of bytes.

More information: https://github.com/sharkdp/hexyl.

- Print the hexadecimal representation of a file:

  ```
  hexyl {{path/to/file}}
  ```

- Print the hexadecimal representation of the first n bytes of a file:

  ```
  hexyl -n {{n}} {{path/to/file}}
  ```

- Print bytes 512 through 1024 of a file:

  ```
  hexyl -r {{512}}:{{1024}} {{path/to/file}}
  ```

- Print 512 bytes starting at the 1024th byte:

  ```
  hexyl -r {{1024}}:+{{512}} {{path/to/file}}
  ```

hg add

Adds specified files to the staging area for the next commit in Mercurial.

More information: https://www.mercurial-scm.org/doc/hg.1_.html#_add

- Add files or directories to the staging area:

```
hg add {{path/to/file}}
```

- Add all unstaged files matching a specified pattern:

```
hg add --include {{pattern}}
```

- Add all unstaged files, excluding those that match a specified pattern:

```
hg add --exclude {{pattern}}
```

- Recursively add sub-repositories:

```
hg add --subrepos
```

- Perform a test-run without performing any actions:

```
hg add --dry-run
```

hg branch

Create or show a branch name.

More information: https://www.mercurial-scm.org/doc/hg.1_.html#_branch

- S how the name of the currently active branch:

  ```
  hg b ranch
  ```

- Create a new branch for the next commit:

  ```
  hg b ranch {{b ranch_nam e}}
  ```

hg clone

Create a copy of an existing repository in a new directory.

More information: https://www.mercurial-scm.org/doc/hg.1_.html#_clone

- Clone a repository to a specified directory:

```
hg clone {{remote_repository_source}} {{destination_path}}
```

- Clone a repository to the head of a specific branch, ignoring later commits:

```
hg clone --branch {{branch}} {{remote_repository_source}}
```

- Clone a repository with only the `.hg` directory, without checking out files:

```
hg clone --noupdate {{remote_repository_source}}
```

- Clone a repository to a specific revision, tag or branch, keeping the entire history:

```
hg clone --update {{revision}}
{{remote_repository_source}}
```

- Clone a repository up to a specific revision without any newer history:

```
hg clone --rev {{revision}} {{remote_repository_source}}
```

hg commit

Commit all staged or specified files to the repository.

More information: https://www.mercurial-scm.org/doc/hg.1_.html#_commit

- Commit staged files to the repository:

```
hg commit
```

- Commit a specific file or directory:

```
hg commit {{path/to/file_or_directory}}
```

- Commit with a specific message:

```
hg commit --message {{message}}
```

- Commit all files matching a specified pattern:

```
hg commit --include {{pattern}}
```

- Commit all files, excluding those that match a specified pattern:

```
hg commit --exclude {{pattern}}
```

- Commit using the interactive mode:

```
hg commit --interactive
```

hg init

Create a new repository in the specified directory.

More information: https://www.mercurial-scm.org/doc/hg.1_.html#_init

- Initialise a new repository in the current directory:

```
hg init
```

- Initialise a new repository in the specified directory:

```
hg init {{path/to/directory}}
```

hg log

Display the revision history of the repository.

More information: https://www.mercurial-scm.org/doc/hg.1_.html#_log

- Display the entire revision history of the repository:

```
hg log
```

- Display the revision history with an ASCII graph:

```
hg log --graph
```

- Display the revision history with file names matching a specified pattern:

```
hg log --include {{pattern}}
```

- Display the revision history, excluding file names that match a specified pattern:

```
hg log --exclude {{pattern}}
```

- Display the log information for a specific revision:

```
hg log --rev {{revision}}
```

- Display the revision history for a specific branch:

```
hg log --branch {{branch}}
```

- Display the revision history for a specific date:

```
hg log --date {{date}}
```

- Display revisions committed by a specific user:

```
hg log --user {{user}}
```

hg pull

Pull changes from a specified repository to the local repository.

More information: https://www.mercurial-scm.org/doc/hg.1.html#pull

- Pull from the "default" source path:

```
hg pull
```

- Pull from a specified source repository:

```
hg pull {{path/to/source_repository}}
```

- Update the local repository to the head of the remote:

```
hg pull --update
```

- Pull changes even when the remote repository is unrelated:

```
hg pull --force
```

- Specify a specific revision changeset to pull up to:

```
hg pull --rev {{revision}}
```

- Specify a specific branch to pull:

```
hg pull --branch {{branch}}
```

- Specify a specific bookmark to pull:

```
hg pull --bookmark {{bookmark}}
```

hg push

Push changes from the local repository to a specified destination.

More information: https://www.mercurial-scm.org/doc/hg.1.html#push

- Push changes to the "default" remote path:

```
hg push
```

- Push changes to a specified remote repository:

```
hg push {{path/to/destination_repository}}
```

- Push a new branch if it does not exist (disabled by default):

```
hg push --new-branch
```

- Specify a specific revision changeset to push:

```
hg push --rev {{revision}}
```

- Specify a specific branch to push:

```
hg push --branch {{branch}}
```

- Specify a specific bookmark to push:

```
hg push --bookmark {{bookmark}}
```

hg remove

Remove specified files from the staging area.

More information: https://www.mercurial-scm.org/doc/hg.1_.html#_remove

- Remove files or directories from the staging area:

```
hg remove {{path/to/file}}
```

- Remove all staged files matching a specified pattern:

```
hg remove --include {{pattern}}
```

- Remove all staged files, excluding those that match a specified pattern:

```
hg remove --exclude {{pattern}}
```

- Recursively remove sub-repositories:

```
hg remove --subrepos
```

- Remove files from the repository that have been physically removed:

```
hg remove --after
```

hg root

Display the root location of a H g repository.

More information: https://www.mercurial-scm.org/doc/hg.1_.html#_root

- Display the root location of the current repository:

```
hg root
```

- Display the root location of the specified repository:

```
hg root --cwd {{path/to/directory}}
```

hg serve

Start a standalone Mercurial web server for browsing repositories.

More information: https://www.mercurial-scm.org/doc/hg.1_.html#_serve

- Start a web server instance:

```
hg serve
```

- Start a web server instance on the specified port:

```
hg serve --port {{port}}
```

- Start a web server instance on the specified listening address:

```
hg serve --address {{address}}
```

- Start a web server instance with a specific identifier:

```
hg serve --name {{name}}
```

- Start a web server instance using the specified theme (see the templates directory):

```
hg serve --style {{style}}
```

- Start a web server instance using the specified SSL certificate bundle:

```
hg serve --certificate {{path/to/certificate}}
```

hg status

Show files that have changed in the working directory.

More information: https://www.mercurial-scm.org/doc/hg.1.html#status

- Display the status of changed files:

```
hg status
```

- Display only modified files:

```
hg status --modified
```

- Display only added files:

```
hg status --added
```

- Display only removed files:

```
hg status --removed
```

- Display only deleted (but tracked) files:

```
hg status --deleted
```

- Display changes in the working directory compared to a specified changeset:

```
hg status --rev {{revision}}
```

- Display only files matching a specified glob pattern:

```
hg status --include {{pattern}}
```

- Display files, excluding those that match a specified glob pattern:

```
hg status --exclude {{pattern}}
```

hg update

U pdate the working directory to a specified changeset.

More information: https://www.mercurial-scm.org/doc/hg.1_.html#_update

- U pdate to the tip of the current branch:

```
hg u pdate
```

- U pdate to the specified revision:

```
hg u pdate - - rev {{rev ision}}
```

- U pdate and discard uncommitted changes:

```
hg u pdate - - clean
```

- U pdate to the last commit matching a specified date:

```
hg u pdate - - date {{dd- m m - yyyy}}
```

hg

A command-line interface for Mercurial, a distributed source control management system.

See `hg add`, `hg commit` and other pages for additional information.

More information: https://www.mercurial-scm.org.

- Execute Mercurial command:

  `hg {{command}}`

- Call general help:

  `hg help`

- Call help on a command:

  `hg help {{command}}`

- Check the Mercurial version:

  `hg --version`

history expansion

Reuse and expand the shell history in `sh`, `bash`, `zsh`, `rbash` and `ksh`.

More information: https://www.gnu.org/software/bash/manual/html_node/History-Interaction.

- Run the previous command as root (`!!` is replaced by the previous command):

  `sudo !!`

- Run a command with the last argument of the previous command:

  `{{command}} !$`

- Run a command with the first argument of the previous command:

  `{{command}} !^`

- Run the Nth command of the history:

  `!{{n}}`

- Run the command `n` lines back in the history:

  `!-{{n}}`

- Run the most recent command containing `string`:

  `!?{{string}}?`

- Run the previous command, replacing `string 1` with `string 2`

  `^{{string 1}}^{{string 2}}^`

- Perform a history expansion, but print the command that would be run instead of actually running it:

  `{{!-n}}:p`

history

Command-line history.

More information: https://www.gnu.org/software/bash/manual/html_node/Bash-History-Builtins.html

- Display the commands history list with line numbers:

```
history
```

- Display the last 20 commands (in `zsh` it displays all commands starting from the 20th):

```
history {{20}}
```

- Clear the commands history list (only for current `bash` shell):

```
history -c
```

- Overwrite history file with history of current `bash` shell (often combined with `history -c` to purge history):

```
history -w
```

- Delete the history entry at the specified offset:

```
history -d {{offset}}
```

hive

CLI tool for Apache Hive.

More information: https://cwiki.apache.org/confluence/display/Hive/LanguageManual+Cli

- Start a Hive interactive shell:

  ```
  hive
  ```

- Run HiveQL:

  ```
  hive -e "{{hiveqlquery}}"
  ```

- Run a HiveQL file with a variable substitution:

  ```
  hive --define {{key}}={{value} -f {{path/to/file.sql}}
  ```

- Run a HiveQL with HiveConfig (e.g. `mapred.reduce.tasks=32`):

  ```
  hive --hiveconf {{conf_name}}={{conf_value}}
  ```

hn

Command-line interface for H acker N ews.

More information: https://github.com/rafaelrinaldi/hn-cli.

- V iew stories on H acker N ews:

  ```
  hn
  ```

- V iewnumber of stories on H acker N ews:

  ```
  hn - - lim it {{nu m b er}}
  ```

- V iew stories on H acker N ews, and keep the list open after selecting a link:

  ```
  hn - - k eep- open
  ```

- V iew stories on H acker N ews sorted by submission date:

  ```
  hn - - latest
  ```

home-manager

Manage a user environment using N ix.

More information: https://github.com/rycee/home-manager.

- Activate the configuration defined in `~ /. config /nixpk g s/hom e: nix hom e- m anag er b u ild`
- Activate the configuration and switch to it:

  `hom e- m anag er sw itch`

host

Lookup Domain Name Server.

- Lookup A, AAAA, and MX records of a domain:

  ```
  host {{domain}}
  ```

- Lookup a field (CNAME, TXT,...) of a domain:

  ```
  host -t {{field}} {{domain}}
  ```

- Reverse lookup an IP:

  ```
  host {{ip_address}}
  ```

- Specify an alternate DNS server to query:

  ```
  host {{domain}} {{8.8.8.8}}
  ```

hostapd

Start an access point using a wireless interface.

More information: https://w1.fi/hostapd/

- Start an access point:

  ```
  sudo hostapd {{path/to/hostapd.conf}}
  ```

- Start an access point, forking into the background:

  ```
  sudo hostapd -B {{path/to/hostapd.conf}}
  ```

hostess

An idempotent command-line utility for managing the `/etc/hosts` file.

More information: https://github.com/cbednarski/hostess.

- List domains, target ips and on/off status:

  ```
  hostess list
  ```

- Add a domain pointing to your machine to your hosts file:

  ```
  hostess add {{local.example.com}} {{127.0.0.1}}
  ```

- Remove a domain from your hosts file:

  ```
  hostess del {{local.example.com}}
  ```

- Disable a domain (but don't remove it completely):

  ```
  hostess off {{local.example.com}}
  ```

hostid

Prints the numeric identifier for the current host (not necessarily the IP address).

More information: https://www.gnu.org/software/coreutils/hostid.

- Display the numeric identifier for the current host in hexadecimal:

```
hostid
```

hping

Command-line oriented TCP/IP packet assembler and analyzer.

Inspired by the **ping** command.

More information: http://www.hping.org.

- Ping localhost over TCP:

```
hping3 {{localhost}}
```

- Ping an IP address over TCP on a specific port:

```
hping3 -p {{80}} -S {{192.168.1.1}}
```

- Ping an IP address over UDP on port 80:

```
hping3 --udp -p {{80}} -S {{192.168.1.1}}
```

- Scan a set of TCP ports on a specific IP address:

```
hping3 --scan {{80,3000,9000}} -S {{192.168.1.1}}
```

- Perform a charge test on port 80:

```
hping3 --flood -p {{80}} -S {{192.168.1.1}}
```

hr

Print a horizontal rule in the terminal.

- Print a horizontal rule:

  ```
  hr
  ```

- Print a horizontal rule with a custom string:

  ```
  hr {{string }}
  ```

- Print a multiline horizontal rule:

  ```
  hr {{string _a}} {{string _b }} {{string _c}}
  ```

hsd-cli

The command-line REST tool for the Handshake blockchain.

More information: https://handshake.org.

- Retrieve information about the current server:

  ```
  hsd-cli info
  ```

- Broadcast a local transaction:

  ```
  hsd-cli broadcast {{transaction_hex}}
  ```

- Retrieve a mempool snapshot:

  ```
  hsd-cli mempool
  ```

- View a transaction by address or hash:

  ```
  hsd-cli tx {{address_or_hash}}
  ```

- View a coin by its hash index or address:

  ```
  hsd-cli coin {{hash_index_or_address}}
  ```

- View a block by height or hash:

  ```
  hsd-cli block {{height_or_hash}}
  ```

- Reset the chain to the specified block:

  ```
  hsd-cli reset {{height_or_hash}}
  ```

- Execute an RPC command:

  ```
  hsd-cli rpc {{command}} {{args}}
  ```

hsw-cli

The command-line REST tool for the Handshake wallet.

More information: https://npmjs.com/package/hs-client

- Unlock the current wallet (timeout in seconds):

  ```
  hsw-cli unlock {{passphrase}} {{timeout}}
  ```

- Lock the current wallet:

  ```
  hsw-cli lock
  ```

- View the current wallet's details:

  ```
  hsw-cli get
  ```

- View the current wallet's balance:

  ```
  hsw-cli balance
  ```

- View the current wallet's transaction history:

  ```
  hsw-cli history
  ```

- Send a transaction with the specified coin amount to an address:

  ```
  hsw-cli send {{address}} {{1.05}}
  ```

- View the current wallet's pending transactions:

  ```
  hsw-cli pending
  ```

- View details about a transaction:

  ```
  hsw-cli tx {{transaction_hash}}
  ```

html5-validator

A command-line tool for testing HTML5 validity.

More information: https://github.com/svenkreiss/html5-validator

- Validate a specific file:

```
html5-validator {{path/to/file}}
```

- Validate all HTML files in a specific directory:

```
html5-validator --root {{path/to/directory}}
```

- Show warnings as well as errors:

```
html5-validator --show-warnings {{path/to/file}}
```

- Match multiple files using a glob pattern:

```
html5-validator --root {{path/to/directory}} --match "{{*.html
*.php}}"
```

- Ignore specific directory names:

```
html5-validator --root {{path/to/directory}} --blacklist
"{{node_modules vendor}}"
```

- Output the results in a specific format:

```
html5-validator --format {{gnu|xml|json|text}} {{path/to/
file}}
```

- Output the log at a specific verbosity level:

```
html5-validator --root {{path/to/directory}} --log {{debug|
info|warning}}
```

htop

Display dynamic real-time information about running processes. An enhanced version of `top`

More information: https://htop.dev/.

- Start htop:

  ```
  htop
  ```

- Start htop displaying processes owned by a specific user:

  ```
  htop --user {{username}}
  ```

- Sort processes by a specified `sort_item` (use `htop --sort help` for available options):

  ```
  htop --sort {{sort_item}}
  ```

- See interactive commands while running htop:

  ```
  ?
  ```

- Display help:

  ```
  htop --help
  ```

htpasswd

Create and manage htpasswd files to protect web server directories using basic authentication.

More information: https://httpd.apache.org/docs/current/programs/htpasswd.html.

- Create/overwrite htpasswd file:

```
htpassw d - c {{path/to/file}} {{u sernam e}}
```

- Add user to htpasswd file or update existing user:

```
htpassw d {{path/to/file}} {{u sernam e}}
```

- Add user to htpasswd file in batch mode without an interactive password prompt (for script usage):

```
htpassw d - b  {{path/to/file}} {{u sernam e}} {{passw ord}}
```

- Delete user from htpasswd file:

```
htpassw d - D  {{path/to/file}} {{u sernam e}}
```

- V erify user password:

```
htpassw d - v  {{path/to/file}} {{u sernam e}}
```

- Display a string with username (plain text) and password (md5):

```
htpassw d - nb m  {{u sernam e}} {{passw ord}}
```

http

HTTPie: HTTP client, aims to be easier to use than cURL.

More information: https://httpie.org.

- Download a URL to a file:

```
http --download {{example.org}}
```

- Send form-encoded data:

```
http --form {{example.org}} {{name='bob'}} {{profile_picture@'bob.png'}}
```

- Send JSON object:

```
http {{example.org}} {{name='bob'}}
```

- Specify an HTTP method:

```
http {{HEAD}} {{example.org}}
```

- Include an extra header:

```
http {{example.org}} {{X-MyHeader:123}}
```

- Pass a user name and password for server authentication:

```
http --auth {{username:password}} {{example.org}}
```

- Specify raw request body via stdin:

```
cat {{data.txt}} | http PUT {{example.org}}
```

httpflow

A command-line utility to capture and dump HTTP streams.

More information: https://github.com/six-ddc/httpflow

- Capture traffic on all interfaces:

```
httpflow -i {{any}}
```

- Use a bpf-style capture to filter the results:

```
httpflow {{host httpbin.org or host baidu.com}}
```

- Use a regular expression to filter requests by URLs:

```
httpflow -u '{{regular_expression}}'
```

- Read packets from pcap format binary file:

```
httpflow -r {{out.cap}}
```

- Write the output to a directory:

```
httpflow -w {{path/to/directory}}
```

httping

Measure the latency and throughput of a web server.

More information: https://manned.org/httping.

- Ping the specified URL:

```
httping -g {{url}}
```

- Ping the web server on `host` and `port`:

```
httping -h {{host}} -p {{port}}
```

- Ping the web server on `host` using a TLS connection:

```
httping -l -g https://{{host}}
```

- Ping the web server on `host` using HTTP basic authentication:

```
httping -g http://{{host}} -U {{username}} -P {{password}}
```

httprobe

Take a list of domains and probe for working HTTP and HTTPS servers.

More information: https://github.com/tomnomnom/httprobe.

- Probe a list of domains from a text file:

  ```
  cat {{input_file}} | httprobe
  ```

- Only check for HTTP if HTTPS is not working:

  ```
  cat {{input_file}} | httprobe --prefer-https
  ```

- Probe additional ports with a given protocol:

  ```
  cat {{input_file}} | httprobe -p {{https:2222}}
  ```

- Output all available options:

  ```
  httprobe --help
  ```

httpry

A lightweight packet sniffer for displaying and logging HTTP traffic.

It can be run in real-time displaying the traffic as it is parsed, or as a daemon process that logs to an output file.

More information: http://dumpsterventures.com/jason/httpry/

- Save output to a file:

  ```
  httpry -o {{path/to/file.log}}
  ```

- Listen on a specific interface and save output to a binary pcap format file:

  ```
  httpry {{eth0}} -b {{path/to/file.pcap}}
  ```

- Filter output by a comma-separated list of HTTP verbs:

  ```
  httpry -m {{get|post|put|head|options|delete|trace|connect|patch}}
  ```

- Read from an input capture file and filter by IP:

  ```
  httpry -r {{path/to/file.log}} '{{host 192.168.5.25}}'
  ```

- Run as daemon process:

  ```
  httpry -d -o {{path/to/file.log}}
  ```

hub

A wrapper for Git that adds commands for working with GitHub-based projects.

If set up as instructed by `hub alias`, one can use `git` to run `hub` commands.

More information: https://hub.github.com.

- Clone a repository using its slug (owners can omit the username):

  ```
  hub clone {{username}}/{{repo_name}}
  ```

- Create a fork of the current repository (cloned from another user) under your GitHub profile:

  ```
  hub fork
  ```

- Push the current local branch to GitHub and create a PR for it in the original repository:

  ```
  hub push {{remote_name}} && hub pull-request
  ```

- Create a PR of the current (already pushed) branch, reusing the message from the first commit:

  ```
  hub pull-request --no-edit
  ```

- Create a new branch with the contents of a pull request and switch to it:

  ```
  hub pr checkout {{pr_number}}
  ```

- Upload the current (local-only) repository to your GitHub account:

  ```
  hub create
  ```

- Fetch Git objects from upstream and update local branches:

  ```
  hub sync
  ```

hugo

Template-based static site generator. Uses modules, components, and themes.

More information: https://gohugo.io.

- Create a new Hugo site:

  ```
  hugo new site {{path/to/site}}
  ```

- Create a new Hugo theme (themes may also be downloaded from https://themes.gohugo.io/):

  ```
  hugo new theme {{theme_name}}
  ```

- Create a new page:

  ```
  hugo new {{section_name}}/{{filename}}
  ```

- Build a site to the `./public/` directory:

  ```
  hugo
  ```

- Build a site including pages that are marked as a "draft":

  ```
  hugo --buildDrafts
  ```

- Build a site to a given directory:

  ```
  hugo --destination {{path/to/destination}}
  ```

- Build a site, start up a webserver to serve it, and automatically reload when pages are edited:

  ```
  hugo server
  ```

hunspell

Check spelling.

More information: https://hunspell.github.io/.

- Check the spelling of a file:

  ```
  hu nspell {{path/to/file}}
  ```

- Check the spelling of a file with the en_ U S dictionary:

  ```
  hu nspell - d {{en_U S }} {{path/to/file}}
  ```

- List misspelled words in a file:

  ```
  hu nspell - l {{path/to/file}}
  ```

hydra

Online password guessing tool.

Protocols supported include FTP, HTTP(S), SMTP, SNMP, XMPP, SSH, and more.

More information: https://github.com/vanhauser-thc/thc-hydra.

- Start Hydra's wizard:

```
hydra-wizard
```

- Guess SSH credentials using a given username and a list of passwords:

```
hydra -l {{username}} -P {{path/to/wordlist.txt}} {{host_ip}} {{ssh}}
```

- Guess Telnet credentials using a list of usernames and a single password, specifying a non-standard port and IPv6:

```
hydra -L {{path/to/usernames.txt}} -p {{password}} -s {{port}} -6 {{host_ip}} {{telnet}}
```

- Guess FTP credentials using usernames and passwords lists, specifying the number of threads:

```
hydra -L {{path/to/usernames.txt}} -P {{path/to/wordlist.txt}} -t {{n_tasks}} {{host_ip}} {{ftp}}
```

- Guess MySQL credentials using a username and a passwords list, exiting when a username/password pair is found:

```
hydra -l {{username}} -P {{path/to/wordlist.txt}} -f {{host_ip}} {{mysql}}
```

- Guess RDP credentials using a username and a passwords list, showing each attempt:

```
hydra -l {{username}} -P {{path/to/wordlist.txt}} -V {{rdp://host_ip}}
```

- Guess IMAP credentials on a range of hosts using a list of colon-separated username/password pairs:

```
hydra -C {{path/to/usernamepassword_pairs.txt}} {{imap://[hostrangecidr]}}
```

- Guess POP3 credentials on a list of hosts using usernames and passwords lists, exiting when a username/password pair is found:

```
hydra -L {{path/to/usernames.txt}} -P {{path/to/wordlist.txt}} -M {{path/to/hosts.txt}} -F {{pop3}}
```

hyperfine

A command-line benchmarking tool.

More information: https://github.com/sharkdp/hyperfine/.

- Run a basic benchmark, performing at least 10 runs:

```
hyperfine '{{make}}'
```

- Run a comparative benchmark:

```
hyperfine '{{make target1}}' '{{make target2}}'
```

- Change minimum number of benchmarking runs:

```
hyperfine --min-runs {{7}} '{{make}}'
```

- Perform benchmark with warmup:

```
hyperfine --warmup {{5}} '{{make}}'
```

- Run a command before each benchmark run (to clear caches, etc.):

```
hyperfine --prepare '{{make clean}}' '{{make}}'
```

- Run a benchmark where a single parameter changes for each run:

```
hyperfine --prepare '{{make clean}}' --parameter-scan {{num_threads}} {{1}} {{10}} '{{make -j {num_threads}}}'
```

ibmcloud login

Log in to the IBM Cloud.

More information: https://cloud.ibm.com/docs/cli?topic=cli-ibmcloud_cli#ibmcloud_login

- Log in by using an interactive prompt:

```
ibmcloud login
```

- Log in to a specific API endpoint (default is `cloud.ibm.com`):

```
ibmcloud login -a {{api_endpoint}}
```

- Log in by providing username, password and the targeted region as parameters:

```
ibmcloud login -u {{username}} -p {{password}} -r {{us-south}}
```

- Log in with an API key, passing it as an argument:

```
ibmcloud login --apikey {{api_key_string}}
```

- Log in with an API key, passing it as a file:

```
ibmcloud login --apikey @{{path/to/api_key_file}}
```

- Log in with a federated ID (single sign-on):

```
ibmcloud login --sso
```

ibmcloud

A command-line tool for managing IBM Cloud apps and services.

More information: https://cloud.ibm.com/docs/cli?topic=cli-ibmcloud. cli

- Update `ibmcloud` to the latest version:

  `ibmcloud update`

- Install the Cloud Foundry module for accessing Cloud Foundry services:

  `ibmcloud cf install`

- List all available IBM Cloud regions:

  `ibmcloud regions`

- Display `ibmcloud` version:

  `ibmcloud version`

- Display help:

  `ibmcloud help`

- Display help for a subcommand:

  `ibmcloud help {{subcommand}}`

ical

A H irj i/Islamic calendar and converter for the terminal.

More information: https://manned.org/ical.

- Display the current month' s calendar:

  ```
  ical
  ```

- Convert a G regorian date to a H ij ri date:

  ```
  ical - - g reg orian {{yyyym m dd}}
  ```

- Convert a H irj i date to a G regorian date:

  ```
  ical - - hij ri {{yyyym m dd}}
  ```

iconv

Converts text from one encoding to another.

- Convert file to a specific encoding, and print to stdout:

  ```
  iconv -f {{from_encoding}} -t {{to_encoding}} {{input_file}}
  ```

- Convert file to the current locale's encoding, and output to a file:

  ```
  iconv -f {{from_encoding}} {{input_file}} > {{output_file}}
  ```

- List supported encodings:

  ```
  iconv -l
  ```

id

Display current user and group identity.

More information: https://www.gnu.org/software/coreutils/id.

- Display current user's id (UID), group id (GID) and groups to which they belong:

  ```
  id
  ```

- Display the current user identity as a number:

  ```
  id -u
  ```

- Display the current group identity as a number:

  ```
  id -g
  ```

- Display an arbitrary user's id (UID), group id (GID) and groups to which they belong:

  ```
  id {{username}}
  ```

id3 tag

Tool for reading, writing, and manipulating ID3 v1 and ID3 v2 tags of MP3 files.

- Set artist and title tag of an MP3 file:

  ```
  id3tag --artist={{artist}} --title={{title}} {{path/to/file.mp3}}
  ```

- Set album title of all MP3 files in the current directory:

  ```
  id3tag --album={{album}} {{*.mp3}}
  ```

- Get more help:

  ```
  id3tag --help
  ```

identify

Command-line utility of Image Magick project to describe the format and characteristics of one or more image files.

More information: https://imagemagick.org/script/identify.php.

- Collect dimensions of all jpeg files under current directory:

```
identify -format "%f,%w,%h\n" *.{{jpg}} > {{filelist.csv}}
```

idnits

Check internet-drafts for submission nits.

Looks for violations of Section 2.1 and 2.2 of the requirements listed at https://www.ietf.org/id-info/checklist.

More information: https://tools.ietf.org/tools/idnits/.

- Check a file for nits:

  ```
  idnits {{path/to/file.txt}}
  ```

- Count nits without displaying them:

  ```
  idnits --nitcount {{path/to/file.txt}}
  ```

- Show extra information about offending lines:

  ```
  idnits --verbose {{path/to/file.txt}}
  ```

- Expect the specified year in the boilerplate instead of the current year:

  ```
  idnits --year {{2021}} {{path/to/file.txt}}
  ```

- Assume the document is of the specified status:

  ```
  idnits --doctype {{standard|informational|experimental|bcp|ps|ds}} {{path/to/file.txt}}
  ```

iex

IEx is the interactive shell for Elixir.

More information: https://hexdocs.pm/iex.

- Start an interactive session:

  ```
  iex
  ```

- Start a session that remembers history:

  ```
  iex --erl "-kernel shell_history enabled"
  ```

- Start and load Mix project files:

  ```
  iex -S mix
  ```

if

Simple shell conditional.

See also: `test`, `[`.

- Execute two different commands based on a condition:

```
if {{command}}; then {{echo "true"}}; else {{echo "false"}};
fi
```

- Check if a variable is defined:

```
if [[ -n "{{$VARIABLE}}" ]]; then {{echo "defined"}}; el
{{echo "not defined"}}; fi
```

- Check if a file exists:

```
if [[ -f "{{path/to/file}}" ]]; then {{echo "true"}}; else
{{echo "false"}}; fi
```

- Check if a directory exists:

```
if [[ -d "{{path/to/directory}}" ]]; then {{echo "true"}};
else {{echo "false"}}; fi
```

- Check if a file or directory exists:

```
if [[ -e "{{path/to/file_or_directory}}" ]]; then {{echo
"true"}}; else {{echo "false"}}; fi
```

- List all possible conditions (`test` is an alias to `[`; both are commonly used with `if`):

```
man [
```

ifconfig

Network Interface Configurator.

More information: https://net-tools.sourceforge.io/man/ifconfig.8.html

- View network settings of an ethernet adapter:

```
ifconfig eth0
```

- Display details of all interfaces, including disabled interfaces:

```
ifconfig -a
```

- Disable eth0 interface:

```
ifconfig eth0 down
```

- Enable eth0 interface:

```
ifconfig eth0 up
```

- Assign IP address to eth0 interface:

```
ifconfig eth0 {{ip_address}}
```

ignite

A CLI for R eact N ative boilerplates, plugins, generators, and more.

More information: https://infinite.red/ignite.

- Create a new R eact N ative proj ect:

  ```
  ig nite new {{proj ect_nam e}}
  ```

- G enerate file from a plugin:

  ```
  ig nite g enerate {{plu g in_nam e}} {{filenam e}}
  ```

- Add an Ignite plugin to the proj ect:

  ```
  ig nite add {{plu g in_nam e}}
  ```

- R emove an Ignite plugin from the proj ect:

  ```
  ig nite rem ov e {{plu g in_nam e}}
  ```

imapsync

E mail IMAP tool for syncing, copying and migrating email mailboxes between two imap servers, one way, and without duplicates.

More information: https://imapsync.lamiral.info.

- S ynchroniz e imap account between host1 and host2 :

```
im apsync - - host1 {{host1 }} - - u ser1 {{u ser1 }} - - passw ord1
{{secret1 }} - - host2 {{host2 }} - - u ser2 {{u ser2 }} - - passw ord2
{{secret2 }}
```

import

Capture some or all of an X server screen, and save the image to a file.

Part of the ImageMagick library.

- Capture the entire X server screen in the PostScript image format:

  ```
  import -window root {{output.postscript}}
  ```

- Capture contents of a remote X server screen in the PNG image format:

  ```
  import -window root -display {{remote_host}}:{screen}.{display} {{output.png}}
  ```

- Capture a specific window, given its ID as displayed by `xwininfo`, into the JPEG format:

  ```
  import -window {{window_id}} {{output.jpg}}
  ```

in2csv

Converts various tabular data formats into CSV.

Included in csvkit.

More information: https://csvkit.readthedocs.io/en/latest/scripts/in2csv.html

- Convert an XLS file to CSV:

  ```
  in2csv {{data.xls}}
  ```

- Convert a DBF file to a CSV file:

  ```
  in2csv {{data.dbf}} > {{data.csv}}
  ```

- Convert a specific sheet from an XLSX file to CSV:

  ```
  in2csv --sheet={{sheet_name}} {{data.xlsx}}
  ```

- Pipe a JSON file to in2csv:

  ```
  cat {{data.json}} | in2csv -f json > {{data.csv}}
  ```

indent

Change the appearance of a C/C++ program by inserting or deleting whitespace.

More information: https://www.gnu.org/software/indent/.

- Format C/C++ source according to the Linux style guide, automatically back up the original files, and replace with the indented versions:

```
indent --linux-style {{path/to/source.c}} {{path/to/another_source.c}}
```

- Format C/C++ source according to the GNU style, saving the indented version to a different file:

```
indent --gnu-style {{path/to/source.c}} -o {{path/to/indented_source.c}}
```

- Format C/C++ source according to the style of Kernighan & Ritchie (K&R), no tabs, 3 spaces per indent, and wrap lines at 120 characters:

```
indent --k-and-r-style --indent-level3 --no-tabs --line-length120 {{path/to/source.c}} -o {{path/to/indented_source.c}}
```

infection

A mutation testing framework for PHP.

More information: https://infection.github.io.

- Analyse code using the configuration file (or create one if it does not exist):

  ```
  infection
  ```

- Use a specific number of threads:

  ```
  infection --threads {{number_of_threads}}
  ```

- Specify a minimum Mutation Score Indicator (MSI):

  ```
  infection --min-msi {{percentage}}
  ```

- Specify a minimum covered code MSI:

  ```
  infection --min-covered-msi {{percentage}}
  ```

- Use a specific test framework (defaults to phpunit):

  ```
  infection --test-framework {{phpunit|phpspec}}
  ```

- Only mutate lines of code that are covered by tests:

  ```
  infection --only-covered
  ```

- Display the mutation code that has been applied:

  ```
  infection --show-mutations
  ```

- Specify the log verbosity:

  ```
  infection --log-verbosity {{default|all|none}}
  ```

influx

Influxdb command-line client.

More information: https://docs.influxdata.com/influxdb/v1.7/tools/shell/

- Connect to an Influxdb running on localhost with no credentials:

```
influx
```

- Connect with a specific username (will prompt for a password):

```
influx -username {{username}} -password ""
```

- Connect to a specific host:

```
influx -host {{hostname}}
```

- Use a specific database:

```
influx -database {{database_name}}
```

- Execute a given command:

```
influx -execute "{{influxql_command}}"
```

- Return output in a specific format:

```
influx -execute "{{influxql_command}}" -format {{json|csv|column}}
```

info

Reads documentation stored in the info format.

More information: https://en.wikipedia.org/wiki/Info_(Unix)

- Start reading top-level directory menu:

  ```
  info
  ```

- Start reading at given menu item node from top-level directory:

  ```
  info {{menu_item}}
  ```

- Start reading at second menu item within first menu item manual:

  ```
  info {{first_menu_item}} {{second_menu_item}}
  ```

initdb

Create a PostgreSQL database on disk.

More information: https://www.postgresql.org/docs/9.5/app-initdb.html

- Create a database at `/usr/local/var/postgres`

  ```
  initdb -D /usr/local/var/postgres
  ```

inkmake

G N U Makefile-style S V G exporting using Inkscape' s backend.

More information: https://github.com/wader/inkmake.

- E xport an S V G file executing the specified Inkfile:

  ```
  ink m ak e {{path/to/I nk file}}
  ```

- E xecute an Inkfile and show detailed information:

  ```
  ink m ak e - - v erb ose {{path/to/I nk file}}
  ```

- E xecute an Inkfile, specifying S V G input file(s) and an output file:

  ```
  ink m ak e - - sv g  {{path/to/file. sv g }} - - ou t {{path/to/
  ou tpu t_im ag e}} {{path/to/I nk file}}
  ```

- S pecify a custom Inkscape binary to use as the backend:

  ```
  ink m ak e - - ink scape {{/A pplications/I nk scape. app/C ontents/
  R esou rces/b in/ink scape}} {{path/to/I nk file}}
  ```

- Display help:

  ```
  ink m ak e - - help
  ```

inkscape

An SVG (Scalable Vector Graphics) editing program.

For Inkscape versions up to 0.92.x, use -e instead of -o.

More information: https://inkscape.org.

- Open an SVG file in the Inkscape GUI:

```
inkscape {{filename.svg}}
```

- Export an SVG file into a bitmap with the default format (PNG) and the default resolution (96 DPI):

```
inkscape {{filename.svg}} -o {{filename.png}}
```

- Export an SVG file into a bitmap of 600x400 pixels (aspect ratio distortion may occur):

```
inkscape {{filename.svg}} -o {{filename.png}} -w {{600}} -h {{400}}
```

- Export the drawing (bounding box of all objects) of an SVG file into a bitmap:

```
inkscape {{filename.svg}} -o {{filename.png}} -D
```

- Export a single object, given its ID, into a bitmap:

```
inkscape {{filename.svg}} -i {{id}} -o {{object.png}}
```

- Export an SVG document to PDF, converting all texts to paths:

```
inkscape {{filename.svg}} -o {{filename.pdf}} --export-text-to-path
```

- Duplicate the object with id="path123", rotate the duplicate 90 degrees, save the file, and quit Inkscape:

```
inkscape {{filename.svg}} --select=path123 --verb="{{EditDuplicate;ObjectRotate90;FileSave;FileQuit}}
```

inkview

Inkscape graphical S V G previewer.

Also functions as a slideshow viewer.

More information: http://wiki.inkscape.org/wiki/index.php/Inkview.

- P review an S V G :

  ```
  ink v iew {{path/to/file. sv g }}
  ```

- P review multiple S V G s (use arrow keys to navigate):

  ```
  ink v iew {{path/to/file_a. sv g }} {{path/to/file_b . sv g }} {{path/
  to/file_c. sv g }}
  ```

install

Copy files and set attributes.

Copy files (often executable) to a system location like `/usr/local/bin`, give them the appropriate permissions/ownership.

More information: https://www.gnu.org/software/coreutils/install.

- Copy files to destination:

```
install {{path/to/source}} {{path/to/destination}}
```

- Copy files to destination, setting their ownership:

```
install -o {{user}} {{path/to/source}} {{path/to/destination}}
```

- Copy files to destination, setting their group ownership:

```
install -g {{user}} {{path/to/source}} {{path/to/destination}}
```

- Copy files to destination, setting their mode:

```
install -m {{+x}} {{path/to/source}} {{path/to/destination}}
```

- Copy files and apply access/modification times of source to destination:

```
install -p {{path/to/source}} {{path/to/destination}}
```

interdiff

S how differences between two diff files.

More information: http://freshmeat.sourceforge.net/proj ects/patchutils

- Compare diff files:

```
interdiff {{old_file}} {{new _file}}
```

- Compare diff files, ignoring whitespace:

```
interdiff - w  {{old_file}} {{new _file}}
```

ionic

A framework to build hybrid mobile apps.

More information: https://ionicframework.com/docs/cli.

- Create a new project:

  ```
  ionic start
  ```

- Start a local dev server for app dev/testing:

  ```
  ionic serve
  ```

- Generate new app component, directive, page, pipe, provider or tabs:

  ```
  ionic g {{page}}
  ```

- Show versions of ionic, cordova, environment, etc.:

  ```
  ionic info
  ```

- Run app on an android/ios device:

  ```
  ionic cordova run {{android|ios}} --device
  ```

- Check the health of a ionic app:

  ```
  ionic doctor {{check}}
  ```

ionice

Get or set program I/O scheduling class and priority.

Scheduling classes: 1 (realtime), 2 (best-effort), 3 (idle).

Priority levels: 0 (the highest) - 7 (the lowest).

- Set I/O scheduling class of a running process:

  ```
  ionice -c {{scheduling_class}} -p {{pid}}
  ```

- Run a command with custom I/O scheduling class and priority:

  ```
  ionice -c {{scheduling_class}} -n {{priority}} {{command}}
  ```

- Print the I/O scheduling class and priority of a running process:

  ```
  ionice -p {{pid}}
  ```

ioping

Monitor I/O latency in real time.

More information: https://github.com/koct9i/ioping

- Show disk I/O latency using the default values and the current directory:

```
ioping .
```

- Measure latency on /tmp using 10 requests of 1 megabyte each:

```
ioping -c 10 -s 1M /tmp
```

- Measure disk seek rate on /dev/sdX:

```
ioping -R {{/dev/sdX}}
```

- Measure disk sequential speed on /dev/sdX:

```
ioping -RL {{/dev/sdX}}
```

iotop

Display a table of current I/O usage by processes or threads.

More information: https://manned.org/iotop.

- Start top-like I/O monitor:

```
sudo iotop
```

- Show only processes or threads actually doing I/O :

```
sudo iotop --only
```

- Show I/O usage in non-interactive mode:

```
sudo iotop --batch
```

- Show only I/O usage of processes (default is to show all threads):

```
sudo iotop --processes
```

- Show I/O usage of given PID(s):

```
sudo iotop --pid={{PID}}
```

- Show I/O usage of a given user:

```
sudo iotop --user={{user}}
```

- Show accumulated I/O instead of bandwidth:

```
sudo iotop --accumulated
```

ipcs

Display information about resources used in IPC (Inter-process Communication).

- Specific information about the Message Queue which has the id 32768:

  ```
  ipcs -qi 32768
  ```

- General information about all the IPC:

  ```
  ipcs -a
  ```

iperf

Measure network bandwidth between computers.

More information: https://iperf.fr.

- Run on server:

```
iperf -s
```

- Run on server using UDP mode and set server port to listen on 5001:

```
iperf -u -s -p {{5001}}
```

- Run on client:

```
iperf -c {{server_address}}
```

- Run on client every 2 seconds:

```
iperf -c {{server_address}} -i {{2}}
```

- Run on client with 5 parallel threads:

```
iperf -c {{server_address}} -P {{5}}
```

- Run on client using UDP mode:

```
iperf -u -c {{server_address}} -p {{5001}}
```

iperf3

Traffic generator for testing network bandwidth.

More information: https://iperf.fr.

- Run iperf3 as a server:

```
iperf3 -s
```

- Run an iperf3 server on a specific port:

```
iperf3 -s -p {{port}}
```

- Start bandwidth test:

```
iperf3 -c {{server}}
```

- Run iperf3 in multiple parallel streams:

```
iperf3 -c {{server}} -P {{streams}}
```

- Reverse direction of the test. Server sends data to the client:

```
iperf3 -c {{server}} -R
```

ipfs

Inter P lanetary F ile S ystem.

A peer-to-peer hypermedia protocol. Aims to make the web more open.

More information: https://ipfs.io.

- Add a file from local to the filesystem, pin it and print the relative hash:

  ```
  ipfs add {{filenam e}}
  ```

- Add a directory and its files recursively from local to the filesystem and print the relative hash:

  ```
  ipfs add - r {{directory}}
  ```

- S ave a remote file and give it a name but not pin it:

  ```
  ipfs g et {{hash}} - o {{filenam e}}
  ```

- P in a remote file locally:

  ```
  ipfs pin add {{hash}}
  ```

- Display pinned files:

  ```
  ipfs pin ls
  ```

- U npin a file from the local storage:

  ```
  ipfs pin rm  {{hash}}
  ```

- R emove unpinned files from local storage:

  ```
  ipfs repo g c
  ```

IPython

A Python shell with automatic history, dynamic object introspection, easier configuration, command completion, access to the system shell and more.

More information: https://ipython.org/documentation.html.

- Start an interactive IPython session:

  ```
  ipython
  ```

- Enter an interactive IPython session after running a Python script:

  ```
  ipython -i {{script.py}}
  ```

- Create default IPython profile:

  ```
  ipython profile create
  ```

- Print the path to the directory for the default IPython profile:

  ```
  ipython locate profile
  ```

- Clear the IPython history database, deleting all entries:

  ```
  ipython history clear
  ```

irssi

Text based IRC client.

More information: https://irssi.org.

- Open irssi and connect to a server with a nickname:

  ```
  irssi -n {{nickname}} -c {{irc.example.com}}
  ```

- Open irssi and connect with a specific server on a given port:

  ```
  irssi -c {{irc.example.com}} -p {{port}}
  ```

- View the help:

  ```
  irssi --help
  ```

- Join a channel:

  ```
  /join {{#channelname}}
  ```

- Change active window (starts at 1):

  ```
  /win {{window_number}}
  ```

- Exit the application cleanly and quitting any server(s):

  ```
  /quit
  ```

is-up

Check whether a website is up or down.

More information: https://github.com/sindresorhus/is-up-cli.

- Check the status of the specified website:

```
is- u p {{exam ple. com }}
```

iverilog

Preprocesses and compiles Verilog HDL (IEEE-1364) code, into executable programs for simulation.

More information: http://iverilog.icarus.com/.

- Compile a source file into an executable:

```
iverilog {{source.v}} -o {{executable}}
```

- Also display all warnings:

```
iverilog {{source.v}} -Wall -o {{executable}}
```

- Compile and run explicitly using the VVP runtime:

```
iverilog -o {{executable}} -tvvp {{source.v}}
```

- Compile using Verilog library files from a different path:

```
iverilog {{source.v}} -o {{executable}} -I {{path/to/library_directory}}
```

- Preprocess Verilog code without compiling:

```
iverilog -E {{source.v}}
```

j ar

J ava Applications/Libraries P ackager.

More information: https://docs.oracle.com/j avase/tutorial/deployment/j ar/basicsindex.html.

- R ecursively archive all files in the current directory into a .j ar file:

```
j ar cf {{file. j ar}} *
```

- U nz ip .j ar/.war file to the current directory:

```
j ar - xv f {{file. j ar}}
```

- List a .j ar/.war file content:

```
j ar tf {{path/to/file. j ar}}
```

- List a .j ar/.war file content with verbose output:

```
j ar tv f {{path/to/file. j ar}}
```

j arsigner

Sign and verify Java Archive (JAR) files.

More information: https://docs.oracle.com/javase/9/tools/jarsigner.htm

- Sign a JAR file:

  ```
  jarsigner {{path/to/file.jar}} {{keystore_alias}}
  ```

- Sign a JAR file with a specific algorithm:

  ```
  jarsigner -sigalg {{algorithm}} {{path/to/file.jar}} {{keystore_alias}}
  ```

- Verify the signature of a JAR file:

  ```
  jarsigner -verify {{path/to/file.jar}}
  ```

java

Java Application Launcher.

More information: https://java.com

- Execute a java `class` file that contains a main method by using just the class name:

  ```
  java {{classname}}
  ```

- Execute a `jar` program:

  ```
  java -jar {{filename.jar}}
  ```

- Execute a `jar` program with debug waiting to connect on port 5005:

  ```
  java -
  agentlib:jdwp=transport=dt_socket,server=y,suspend=y,address
  5005 -jar {{filename.jar}}
  ```

- Display JDK, JRE and HotSpot versions:

  ```
  java -version
  ```

- Display usage information for the java command:

  ```
  java -help
  ```

j avac

J ava Application Compiler.

- Compile a `. j av a` file:

  ```
  j av ac {{file. j av a}}
  ```

- Compile several `. j av a` files:

  ```
  j av ac {{file1 . j av a}} {{file2 . j av a}} {{file3 . j av a}}
  ```

- Compile all `. j av a` files in current directory:

  ```
  j av ac {{* . j av a}}
  ```

- Compile a `. j av a` file and place the resulting class file in a specific directory:

  ```
  j av ac - d {{path/to/som e/directory}} {{file. j av a}}
  ```

j avadoc

G enerate J ava AP I documentation in H T ML format from source code.

More information: https://docs.oracle.com/j avase/9 /j avadoc/j avadoc-command.htm

- G enerate documentation for J ava source code and save the result in a directory:

  ```
  j av adoc - d {{path/to/directory/}} {{path/to/
  j av a_sou rce_code}}
  ```

- G enerate documentation with a specific encoding:

  ```
  j av adoc - docencoding {{U T F - 8 }} {{path/to/j av a_sou rce_code}}
  ```

- G enerate documentation excluding some packages:

  ```
  j av adoc - exclu de {{pack ag e_list}} {{path/to/
  j av a_sou rce_code}}
  ```

j c

A utility to convert the output of multiple commands to J S O N .

More information: https://github.com/kellyj onbraz il/j c

- Convert command output to J S O N via pipe:

  ```
  {{ifconfig }} | j c {{- - ifconfig }}
  ```

- Convert command output to J S O N via magic syntax:

  ```
  j c {{ifconfig }}
  ```

- O utput pretty J S O N via pipe:

  ```
  {{ifconfig }} | j c {{- - ifconfig }} - p
  ```

- O utput pretty J S O N via magic syntax:

  ```
  j c - p {{ifconfig }}
  ```

j cal

Display calendar information in the J alali format, with the current day highlighted.

More information: http://www.nongnu.org/j cal/

- Display a calendar for the current month:

  ```
  j cal
  ```

- Display the previous, current, and next months:

  ```
  j cal -3
  ```

- Display a calendar for a specific year (4 digits):

  ```
  j cal {{year}}
  ```

- Display a calendar for a specific month and year:

  ```
  j cal {{year}} {{month}}
  ```

j dupes

A powerful duplicate file finder and an enhanced fork of fdupes.

More information: https://github.com/j bruchon/j dupes

- S earch a single directory:

```
j du pes {{directory}}
```

- S earch multiple directories:

```
j du pes {{directory1 }} {{directory2 }}
```

- S earch all directories recursively:

```
j du pes - - recu rse {{directory}}
```

- S earch directory recursively and let user choose files to preserve:

```
j du pes - - delete - - recu rse {{directory}}
```

- S earch multiple directories and follow subdirectores under directory2 , not directory1 :

```
j du pes {{directory1 }} - - recu rse: {{directory2 }}
```

- S earch multiple directories and keep the directory order in result:

```
j du pes - O {{directory1 }} {{directory2 }} {{directory3 }}
```

j ekyll

A simple, blog-aware, static site generator.

More information: https://j ekyllrb.com

- G enerate a development server that will run at http://localhost:4 0 0 0 /:

  `j ek yll serv e`

- E nable incremental regeneration:

  `j ek yll serv e - - increm ental`

- E nable verbose output:

  `j ek yll serv e - - v erb ose`

- G enerate the current directory into `. /_site`

  `j ek yll b u ild`

- Clean the site (removes site output and `cache` directory) without building:

  `j ek yll clean`

j ello

A command-line J S O N processor using P ython syntax.

More information: [https://github.com/kellyj onbraz il/j .ello](https://github.com/kellyj_onbraz_il/j_ello)

- P retty-print J S O N or J S O N -Lines data from stdin to stdout:

  ```
  cat {{file. j son}} | j ello
  ```

- O utput a schema of J S O N or J S O N Lines data from stdin to stdout (useful for grep):

  ```
  cat {{file. j son}} | j ello - s
  ```

- O utput all elements from arrays (or all the values from obj ects) in J S O N or J S O N -Lines data from stdin to stdout:

  ```
  cat {{file. j son}} | j ello - l
  ```

- O utput the first element in J S O N or J S O N -Lines data from stdin to stdout:

  ```
  cat {{file. j son}} | j ello _[ 0 ]
  ```

- O utput the value of a given key of each element in J S O N or J S O N -Lines data from stdin to stdout:

  ```
  cat {{file. j son}} | j ello ' [ i. {{k ey_nam e}} for i in _] '
  ```

- O utput the value of multiple keys as a new J S O N obj ect (assuming the input J S O N has the keys `k ey_nam e` and `other_k ey_nam e`):

  ```
  cat {{file. j son}} | j ello ' {" {{m y_new _k ey}}" : _. {{k ey_nam e}},
  " {{m y_other_k ey}}" : _. {{other_k ey_nam e}}}'
  ```

- O utput the value of a given key to a string (and disable J S O N output):

  ```
  cat {{file. j son}} | j ello - r ' " {{som e text}}: " + _.
  {{k ey_nam e}}'
  ```

j env

Command-line tool to manage the "JAVA_HOME" environment variable.

More information: https://www.j env.be/

- Add a Java version to jEnv:

```
j env add {{path/to/j dk _hom e}}
```

- Display the current JDK version used:

```
j env v ersion
```

- Display all managed JDKs:

```
j env v ersions
```

- Set the global JDK version:

```
j env g lob al {{j av a_v ersion}}
```

- Set the JDK version for the current shell session:

```
j env shell {{j av a_v ersion}}
```

- Enable a jEnv plugin:

```
j env enab le- plu g in {{plu g in_nam e}}
```

j est

A z ero-configuration J avaS cript testing platform.

More information: https://j estj s.io

- R un all available tests:

  ```
  j est
  ```

- R un the test suites from the given files:

  ```
  j est {{path/to/file1 }} {{path/to/file2 }}
  ```

- R un the test suites from files within the current and subdirectories, whose paths match the given regular expression:

  ```
  j est {{reg u lar_expression1 }} {{reg u lar_expression2 }}
  ```

- R un the tests whose names match the given regular expression:

  ```
  j est - - testN am eP attern {{reg u lar_expression}}
  ```

- R un test suites related to a given source file:

  ```
  j est - - findR elatedT ests {{path/to/sou rce_file. j s}}
  ```

- R un test suites related to all uncommitted files:

  ```
  j est - - onlyC hang ed
  ```

- Watch files for changes and automatically re-run related tests:

  ```
  j est - - w atch
  ```

- S how help:

  ```
  j est - - help
  ```

j etifier

J etifier AndroidX transition tool in npm format, with a react-native compatible style.

More information: https://github.com/mikehardy/j etifier

- Migrate proj ect dependencies to the AndroidX format:

  ```
  j etifier
  ```

- Migrate proj ect dependencies from the AndroidX format:

  ```
  j etifier rev erse
  ```

j hat

J ava H eap Analysis T ool.

More information: https://docs.oracle.com/j avase/8 /docs/technotes/tools/unix/j hat.html

- Analyz e a heap dump (from j map), view via http on port 7 0 0 0 :

  ```
  j hat {{du m p_file. b in}}
  ```

- Analyz e a heap dump, specifying an alternate port for the http server:

  ```
  j hat - p {{port}} {{du m p_file. b in}}
  ```

- Analyz e a dump letting j hat use up to 8 G B R AM (2 -4 x dump siz e recommended):

  ```
  j hat - J - m x8 G {{du m p_file. b in}}
  ```

j hipster

Web application generator using either monolithic or microservices architecture.

More information: https://www.j hipster.tech/

- G enerate a simple full-stack proj ect (monolithic or microservices):

  ```
  j hipster
  ```

- G enerate a simple frontend proj ect:

  ```
  j hipster - - sk ip- serv er
  ```

- G enerate a simple backend proj ect:

  ```
  j hipster - - sk ip- client
  ```

- Apply latest J H ipster updates to the proj ect:

  ```
  j hipster u pg rade
  ```

- Add a new entity to a generated proj ect:

  ```
  j hipster entity {{entity_nam e}}
  ```

- Import a J DL file to configure your application (see: https://start.j hipster.tech/j dl-studio/):

  ```
  j hipster im port- j dl {{first_file. j h second_file. j h . . .
  n_file. j h}}
  ```

- G enerate a CI/CD pipeline for your application:

  ```
  j hipster ci- cd
  ```

- G enerate a Kubernetes configuration for your application:

  ```
  j hipster k u b ernetes
  ```

j igsaw

A Laravel-based static site builder for P H P .

More information: https://j igsaw.tighten.co

- Initialise a proj ect:

  ```
  j ig saw init
  ```

- Initialise a proj ect using a starter template:

  ```
  j ig saw init {{tem plate_nam e}}
  ```

- B uild the site for development:

  ```
  j ig saw b u ild
  ```

- P review the site from the " build_ local" directory:

  ```
  j ig saw serv e
  ```

- B uild the site for production:

  ```
  j ig saw b u ild produ ction
  ```

- P review the site from the " build_ production" directory:

  ```
  j ig saw serv e {{b u ild_produ ction}}
  ```

jmap

Java Memory Map Tool.

- Print shared object mappings for a Java process (output like pmap):

  ```
  jmap {{java_pid}}
  ```

- Print heap summary information:

  ```
  jmap -heap {{filename.jar}} {{java_pid}}
  ```

- Print histogram of heap usage by type:

  ```
  jmap -histo {{java_pid}}
  ```

- Dump contents of the heap into a binary file for analysis with jhat:

  ```
  jmap -dump:format=b,file={{filename}} {{java_pid}}
  ```

j obs

Display status of j obs in the current session.

- S how status of all j obs:

  ```
  j ob s
  ```

- S how status of a particular j ob:

  ```
  j ob s % {{j ob _id}}
  ```

- S how status and process IDs of all j obs:

  ```
  j ob s - l
  ```

- S how process IDs of all j obs:

  ```
  j ob s - p
  ```

j ohn

P assword cracker.

More information: https://www.openwall.com/j ohn/

- Crack password hashes:

```
j ohn {{path/to/hashes. txt}}
```

- S how passwords cracked:

```
j ohn - - show {{path/to/hashes. txt}}
```

- Display users' cracked passwords by user identifier from multiple files:

```
j ohn - - show - - u sers= {{u ser_ids}} {{path/to/hashes* }} {{path/
to/other/hashes* }}
```

- Crack password hashes, using a custom wordlist:

```
j ohn - - w ordlist= {{path/to/w ordlist. txt}} {{path/to/
hashes. txt}}
```

- List available hash formats:

```
j ohn - - list= form ats
```

- Crack password hashes, using a specific hash format:

```
j ohn - - form at= {{m d5 crypt}} {{path/to/hashes. txt}}
```

- Crack password hashes, enabling word mangling rules:

```
j ohn - - ru les {{path/to/hashes. txt}}
```

- R estore an interrupted cracking session from a state file, e.g. `m ycrack . rec`

```
j ohn - - restore= {{path/to/m ycrack . rec}}
```

j oin

J oin lines of two sorted files on a common field.

More information: https://www.gnu.org/software/coreutils/j oin

- J oin two files on the first (default) field:

```
j oin {{file1 }} {{file2 }}
```

- J oin two files using a comma (instead of a space) as the field separator:

```
j oin - t {{' , ' }} {{file1 }} {{file2 }}
```

- J oin field3 of file1 with field1 of file2 :

```
j oin - 1 {{3 }} - 2 {{1 }} {{file1 }} {{file2 }}
```

- P roduce a line for each unpairable line for file1 :

```
j oin - a {{1 }} {{file1 }} {{file2 }}
```

j osm

Extensible OpenStreetMap editor for Java 8+.

More information: https://josm.openstreetmap.de/

- Launch JOSM:

  ```
  josm
  ```

- Launch JOSM in maximized mode:

  ```
  josm --maximize
  ```

- Launch JOSM and set a specific language:

  ```
  josm --language {{de}}
  ```

- Launch JOSM and reset all preferences to their default values:

  ```
  josm --reset-preferences
  ```

- Launch JOSM and download a specific bounding box:

  ```
  josm --download {{minlat,minlon,maxlat,maxlon}}
  ```

- Launch JOSM and download a specific bounding box as raw GPS:

  ```
  josm --downloadgps {{minlat,minlon,maxlat,maxlon}}
  ```

- Launch JOSM without plugins:

  ```
  josm --skip-plugins
  ```

j p2 a

Convert J P E G images to AS CII.

More information: https://csl.name/j p2 .a/

- R ead J P E G image from a file and print in AS CII:

  ```
  j p2 a {{path/to/im ag e. j peg }}
  ```

- R ead J P E G image from a U R L and print in AS CII:

  ```
  j p2 a {{w w w . exam ple. com /im ag e. j peg }}
  ```

- Coloriz e the AS CII output:

  ```
  j p2 a - - colors {{path/to/im ag e. j peg }}
  ```

- S pecify characters to be used for the AS CII output:

  ```
  j p2 a - - chars= ' {{. . - ooxx@ @ }}'  {{path/to/im ag e. j peg }}
  ```

- Write the AS CII output into a file:

  ```
  j p2 a - - ou tpu t= {{path/to/ou tpu t_file. txt}} {{path/to/
  im ag e. j peg }}
  ```

- Write the AS CII output in H T ML file format, suitable for viewing in web browsers:

  ```
  j p2 a - - htm l - - ou tpu t= {{path/to/ou tpu t_file. htm l}} {{path/to/
  im ag e. j peg }}
  ```

j pegoptim

Optimise JPEG images.

More information: https://github.com/tj ko/j pegoptim

- Optimise a set of JPEG images, retaining all associated data:

```
j peg optim {{im ag e1 . j peg }} {{im ag e2 . j peg }} {{im ag eN . j peg }
```

- Optimise JPEG images, stripping all non-essential data:

```
j peg optim - - strip- all {{im ag e1 . j peg }} {{im ag e2 . j peg }}
{{im ag eN . j peg }}
```

- Force the output images to be progressive:

```
j peg optim - - all- prog ressiv e {{im ag e1 . j peg }} {{im ag e2 . j peg
{{im ag eN . j peg }}
```

- Force the output images to have a fixed maximum filesize:

```
j peg optim - - size= {{2 5 0 k }} {{im ag e1 . j peg }} {{im ag e2 . j peg
{{im ag eN . j peg }}
```

j ps

S how J V M P rocess S tatus of current user.

More information: https://docs.oracle.com/en/j ava/j avase/1 1 /tools/j ps.html

- List all J V M processes:

```
j ps
```

- List all J V M processes with only P ID:

```
j ps - q
```

- Display the arguments passed to the processes:

```
j ps - m
```

- Display the full package name of all processes:

```
j ps - l
```

- Display the arguments passed to the J V M:

```
j ps - v
```

jq

A command-line JSON processor that uses a domain-specific language.

More information: https://stedolan.github.io/jq

- Output a JSON file, in pretty-print format:

```
jq . {{file.json}}
```

- Output all elements from arrays (or all the values from objects) in a JSON file:

```
jq '.[]' {{file.json}}
```

- Read JSON objects from a file into an array, and output it (inverse of `jq .[]`):

```
jq --slurp . {{file.json}}
```

- Output the first element in a JSON file:

```
jq '.[0]' {{file.json}}
```

- Output the value of a given key of each element in a JSON text from stdin:

```
cat {{file.json}} | jq 'map(.{{key_name}})'
```

- Output the value of multiple keys as a new JSON object (assuming the input JSON has the keys `key_name` and `other_key_name`):

```
cat {{file.json}} | jq '{{{my_new_key}}: .{{key_name}}, {{my_other_key}}: .{{other_key_name}}}'
```

- Combine multiple filters:

```
cat {{file.json}} | jq 'unique | sort | reverse'
```

- Output the value of a given key to a string (and disable JSON output):

```
cat {{file.json}} | jq --raw-output '"some text: \(.{{key_name}})"'
```

j rnl

A simple j ournal application for your command-line.

More information: http://j rnl.sh

- Insert a new entry with your editor:

  ```
  j rnl
  ```

- Q uickly insert a new entry:

  ```
  j rnl {{today at 3 am }}: {{title}}. {{content}}
  ```

- V iew the last ten entries:

  ```
  j rnl - n {{1 0 }}
  ```

- V iew everything that happened from the start of last year to the start of last march:

  ```
  j rnl - from " {{last year}}" - u ntil {{m arch}}
  ```

- E dit all entries tagged with " texas" and " history" :

  ```
  j rnl {{@ texas}} - and {{@ history}} - - edit
  ```

j son5

A command-line tool for converting J S O N 5 files to J S O N .

More information: https://j son5 .org

- Convert J S O N 5 stdin to J S O N stdout:

```
echo {{inpu t}} | j son5
```

- Convert a J S O N 5 file to J S O N and output to stdout:

```
j son5 {{path/to/inpu t_file. j son5 }}
```

- Convert a J S O N 5 file to the specified J S O N file:

```
j son5 {{path/to/inpu t_file. j son5 }} - - ou t- file {{path/to/
ou tpu t_file. j son}}
```

- V alidate a J S O N 5 file:

```
j son5 {{path/to/inpu t_file. j son5 }} - - v alidate
```

- S pecify the number of spaces to indent by (or " t" for tabs):

```
j son5 - - space {{indent_am ou nt}}
```

- V iew available options:

```
j son5 - - help
```

jstack

Java Stack Trace Tool.

- Print Java stack traces for all threads in a Java process:

  ```
  jstack {{java_pid}}
  ```

- Print mixed mode (Java/C++) stack traces for all threads in a Java process:

  ```
  jstack -m {{java_pid}}
  ```

- Print stack traces from Java core dump:

  ```
  jstack {{/usr/bin/java}} {{file.core}}
  ```

j tbl

Utility to print JSON and JSON Lines data as a table in the terminal.

More information: https://github.com/kellyjonbrazil/jtbl

- Print a table from JSON or JSON Lines input:

  `cat {{file.json}} | jtbl`

- Print a table and specify the column width for wrapping:

  `cat {{file.json}} | jtbl --cols={{width}}`

- Print a table and truncate rows instead of wrapping:

  `cat {{file.json}} | jtbl -t`

- Print a table and don't wrap or truncate rows:

  `cat {{file.json}} | jtbl -n`

j ulia

A high-level, high-performance dynamic programming language for technical computing.

More information: https://docs.j ulialang.org/en/v1 /manual/getting-started/

- S tart a J ulia R E P L session:

```
j u lia
```

- E xecute a J ulia program and exit:

```
j u lia {{prog ram . j l}}
```

- E xecute a J ulia program that takes arguments:

```
j u lia {{prog ram . j l}} {{arg u m ents}}
```

- E valuate a string containing J ulia code:

```
j u lia - e ' {{j u lia_code}}'
```

- E valuate a string of J ulia code, passing arguments to it:

```
j u lia - e ' {{for x in A R G S ;  println( x) ;  end}}'  {{arg u m ents}}
```

- E valuate an expression and print the result:

```
j u lia - E  ' {{( 1  -  cos( pi/4 ) ) /2 }}'
```

- S tart J ulia in parallel mode, using N worker processes:

```
j u lia - p {{N }}
```

j upyter

Web application to create and share documents that contain code, visualiz ations and notes.

P rimarily used for data analysis, scientific computing and machine learning.

More information: https://j upyter.org

- S tart a J upyter notebook server in the current directory:

  ```
  j u pyter noteb ook
  ```

- O pen a specific J upyter notebook:

  ```
  j u pyter noteb ook {{exam ple. ipynb }}
  ```

- E xport a specific J upyter notebook into another format:

  ```
  j u pyter nb conv ert - - to {{htm l| m ark dow n| pdf| script}}
  {{exam ple. ipynb }}
  ```

- S tart a server on a specific port:

  ```
  j u pyter noteb ook - - port= {{port}}
  ```

- List currently running notebook servers:

  ```
  j u pyter noteb ook list
  ```

- S top the currently running server:

  ```
  j u pyter noteb ook stop
  ```

- S tart J upyterLab, if installed, in the current directory:

  ```
  j u pyter lab
  ```

jupytext

Tool to convert Jupyter notebooks to plain text documents, and back again.

More information: https://jupytext.readthedocs.io/en/latest/

- Turn a notebook into a paired `.ipynb`/`.py` notebook:

  ```
  jupytext --set-formats ipynb,py {{notebook.ipynb}}
  ```

- Convert a notebook to a `.py` file:

  ```
  jupytext --to py {{notebook.ipynb}}
  ```

- Convert a `.py` file to a notebook with no outputs:

  ```
  jupytext --to notebook {{notebook.py}}
  ```

- Convert a `.md` file to a notebook and run it:

  ```
  jupytext --to notebook --execute {{notebook.md}}
  ```

- Update the input cells in a notebook and preserve outputs and metadata:

  ```
  jupytext --update --to notebook {{notebook.py}}
  ```

- Update all paired representations of a notebook:

  ```
  jupytext --sync {{notebook.ipynb}}
  ```

j wt

A command-line tool to work with J S O N Web T okens (J WT s).

E ncryption algorithms available are H S 2 5 6 , H S 3 8 4 , H S 5 1 2 , R S 2 5 6 , R S 3 8 4 , R E S 3 8 4 .

More information: https://github.com/mike-engel/j wt-cli

- Decode a J WT :

```
j w t decode {{j w t_string }}
```

- Decode a J WT as a J S O N string:

```
j w t decode - j {{j w t_string }}
```

- E ncode a J S O N string to a J WT :

```
j w t encode - - alg {{H S 2 5 6 }} - - secret {{1 2 3 4 5 6 7 8 9 0 }}
' {{j son_string }}'
```

- E ncode key pair payload to J WT :

```
j w t encode - - alg {{H S 2 5 6 }} - - secret {{1 2 3 4 5 6 7 8 9 0 }} - P
k ey= v alu e
```

k6

Open source load testing tool and SaaS for engineering teams.

More information: https://k6.io

- Run load test locally:

```
k6 run {{script.js}}
```

- Run load test locally with a given number of virtual users and duration:

```
k6 run --vus {{10}} --duration {{30s}} {{script.js}}
```

- Run load test locally with a given environment variable:

```
k6 run -e {{HOSTNAME=example.com}} {{script.js}}
```

- Run load test locally using InfluxDB to store results:

```
k6 run --out influxdb={{http://localhost:8086/k6db}} {{script.js}}
```

- Run load test locally and discard response bodies (significantly faster):

```
k6 run --discard-response-bodies {{script.js}}
```

- Run load test locally using the base JavaScript compatibility mode (significantly faster):

```
k6 run --compatibility-mode=base {{script.js}}
```

- Log in to cloud service using secret token:

```
k6 login cloud --token {{secret}}
```

- Run load test on cloud infrastructure:

```
k6 cloud {{script.js}}
```

k8s-unused-secret-detector

Command-line interface tool for detecting unused Kubernetes secrets.

More information: https://github.com/dtan4/k8s-unused-secret-detector

- Detect unused secrets:

```
k8s-unused-secret-detector
```

- Detect unused secrets in a specific namespace:

```
k8s-unused-secret-detector -n {{namespace}}
```

- Delete unused secrets in a specific namespace:

```
k8s-unused-secret-detector -n {{namespace}} | kubectl delete secret -n {{namespace}}
```

k8s sec

Command-line interface tool to manage Kubernetes secrets.

More information: https://github.com/dtan4/k8s-sec

- List all secrets:

  ```
  k8s sec list
  ```

- List a specific secret as a base64-encoded string:

  ```
  k8s sec list {{secret_name}} --b ase64
  ```

- Set a secret's value:

  ```
  k8s sec set {{secret_name}} {{key=value}}
  ```

- Set a base64-encoded value:

  ```
  k8s sec set --b ase64 {{secret_name}} {{key=encoded_value}}
  ```

- Unset a secret:

  ```
  k8s sec unset {{secret_name}}
  ```

- Load secrets from a file:

  ```
  k8s sec load -f {{path/to/file}} {{secret_name}}
  ```

- Dump secrets to a file:

  ```
  k8s sec dump -f {{path/to/file}} {{secret_name}}
  ```

kafkacat

Apache Kafka producer and consumer tool.

More information: https://github.com/edenhill/kafkacat.

- Consume messages starting with the newest offset:

  ```
  kafkacat -C -t {{topic}} -b {{brokers}}
  ```

- Consume messages starting with the oldest offset and exit after the last message is received:

  ```
  kafkacat -C -t {{topic}} -b {{brokers}} -o beginning -e
  ```

- Consume messages as a Kafka consumer group:

  ```
  kafkacat -G {{group_id}} {{topic}} -b {{brokers}}
  ```

- Publish message by reading from stdin:

  ```
  echo {{message}} | kafkacat -P -t {{topic}} -b {{brokers}}
  ```

- Publish messages by reading from a file:

  ```
  kafkacat -P -t {{topic}} -b {{brokers}} {{path/to/file}}
  ```

- List metadata for all topics and brokers:

  ```
  kafkacat -L -b {{brokers}}
  ```

- List metadata for a specific topic:

  ```
  kafkacat -L -t {{topic}} -b {{brokers}}
  ```

- Get offset for a topic/partition for a specific point in time:

  ```
  kafkacat -Q -t {{topic}}:{{partition}}:{{unix_timestamp}} -b
  {{brokers}}
  ```

kaggle

Official CLI for Kaggle implemented in Python 3.

More information: https://github.com/Kaggle/kaggle-api.

- View current configuration values:

  ```
  kaggle config view
  ```

- Download a specific file from a competition dataset:

  ```
  kaggle competitions download {{competition}} -f {{filename}}
  ```

kahlan

A unit and Behaviour Driven Development test framework for PHP.

More information: https://kahlan.github.io.

- Run all specifications in the "spec" directory:

  ```
  kahlan
  ```

- Run specifications using a specific configuration file:

  ```
  kahlan --config={{path/to/configuration_file}}
  ```

- Run specifications and output using a reporter:

  ```
  kahlan --reporter={{dot|bar|json|tap|verbose}}
  ```

- Run specifications with code coverage (detail can be between 0 and 4):

  ```
  kahlan --coverage={{detail_level}}
  ```

kak

Kakoune is a mode-based code editor implementing the "multiple selections" paradigm.

Data can be selected and simultaneously edited in different locations, using multiple selections; users can also connect to the same session for collaborative editing.

More information: https://kakoune.org.

- Open a file and enter normal mode, to execute commands:

  ```
  kak {{path/to/file}}
  ```

- Enter insert mode from normal mode, to write text into the file:

  ```
  i
  ```

- Escape insert mode, to go back to normal mode:

  ```
  <Escape>
  ```

- Replace all instances of "foo" in the current file with "bar":

  ```
  %s{{foo}}<Enter>c{{bar}}<Escape>
  ```

- Un-select all secondary selections, and keep only the main one:

  ```
  <Space>
  ```

- Search for numbers and select the first two:

  ```
  /\d+<Enter>N
  ```

- Insert the contents of a file:

  ```
  !cat {{path/to/file}}<Enter>
  ```

- Save the current file:

  ```
  :w<Enter>
  ```

kate

KDE Text Editor.

More information: https://kate-editor.org/.

- Launch Kate and open specific files:

```
kate {{path/to/file1}} {{path/to/file2}}
```

- Open a remote document in Kate:

```
kate {{https://example.com/path/to/file}}
```

- Launch Kate, creating a new instance even if one is already open:

```
kate --new
```

- Open a file in Kate with the cursor at the specific line:

```
kate --line {{line_number}} {{path/to/file}}
```

- Open a file in Kate with the cursor at the specific line and column:

```
kate --line {{line_number}} --column {{column_number}} {{path/to/file}}
```

- Launch Kate, creating a new temporary file with contents read from stdin:

```
cat {{path/to/file}} | kate --stdin
```

- Display help:

```
kate --help
```

kdeconnect-cli

KDE Connect CLI.

More information: https://kdeconnect.kde.org.

- List all devices:

```
kdeconnect-cli --list-devices
```

- List available (paired and reachable) devices:

```
kdeconnect-cli --list-available
```

- Request pairing with a specific device, specifying its ID:

```
kdeconnect-cli --pair --device {{device_id}}
```

- Ring a device, specifying its name:

```
kdeconnect-cli --ring --name {{device_name}}
```

- Share an URL or file with a paired device, specifying its ID:

```
kdeconnect-cli --share {{URL | path/to/file}} --device
{{device_id}}
```

- Send an SMS with an optional attachment to a specific number:

```
kdeconnect-cli --name {{device_name}} --send-sms {{message}}
--destination {{phone_number}} --attachment {{path/to/file}}
```

- Unlock a specific device:

```
kdeconnect-cli --name {{device_name}} --unlock
```

- Simulate a key press on a specific device:

```
kdeconnect-cli --name {{device_name}} --send-keys {{key}}
```

keepass2

A light-weight password manager.

More information: https://keepass.info.

- Start KeePass2, opening the most recently-opened password database:

  ```
  keepass2
  ```

- Start KeePass2, opening a specific password database:

  ```
  keepass2 {{path/to/database.kdbx}}
  ```

- Use a specific key file to open a password database:

  ```
  keepass2 {{path/to/database.kdbx}} -keyfile:{{path/to/keyfile.key}}
  ```

keepassxc-cli

Command-line interface for KeepassX C.

More information: https://manned.org/keepassxc-cli.

- S earch entries:

```
k eepassxc- cli look u p {{path/to/datab ase_file}} {{nam e}}
```

- List the contents of a folder:

```
k eepassxc- cli ls {{path/to/datab ase_file}} {{/path/to/
directory}}
```

- Add an entry with an auto-generated password:

```
k eepassxc- cli add - - g enerate {{path/to/datab ase_file}}
{{entry_nam e}}
```

- Delete an entry:

```
k eepassxc- cli rm  {{path/to/datab ase_file}} {{entry_nam e}}
```

- Copy an entry' s password to the clipboard:

```
k eepassxc- cli clip {{path/to/datab ase_file}} {{entry_nam e}}
```

- Copy a T O T P code to the clipboard:

```
k eepassxc- cli clip - - totp {{path/to/datab ase_file}}
{{entry_nam e}}
```

- G enerate a passphrase with 7 words:

```
k eepassxc- cli dicew are - - w ords {{7 }}
```

- G enerate a password with 1 6 printable AS CII characters:

```
k eepassxc- cli g enerate - - low er - - u pper - - nu m eric - - special -
leng th {{1 6 }}
```

keybase

Key directory that maps social media identities to encryption keys in a publicly auditable manner.

More information: https://keybase.io/docs/command_line

- Follow another user:

```
keybase follow {{username}}
```

- Add a new proof:

```
keybase prove {{service}} {{service_username}}
```

- Sign a file:

```
keybase sign --infile {{input_file}} --outfile {{output_file}}
```

- Verify a signed file:

```
keybase verify --infile {{input_file}} --outfile {{output_file}}
```

- Encrypt a file:

```
keybase encrypt --infile {{input_file}} --outfile {{output_file}} {{receiver}}
```

- Decrypt a file:

```
keybase decrypt --infile {{input_file}} --outfile {{output_file}}
```

- Revoke current device, log out, and delete local data:

```
keybase deprovision
```

khal

A text-based calendar and scheduling application for the command-line.

More information: https://lostpackets.de/khal.

- S tart khal on interactive mode:

  ```
  ik hal
  ```

- P rint all events scheduled in personal calendar for the next seven days:

  ```
  k hal list - a {{personal}} {{today}} {{7 d}}
  ```

- P rint all events scheduled not in personal calendar for tomorrow at 1 0 :0 0 :

  ```
  k hal at - d {{personal}} {{tom orrow }} {{1 0 : 0 0 }}
  ```

- P rint a calendar with a list of events for the next three months:

  ```
  k hal calendar
  ```

- Add new event to personal calendar:

  ```
  k hal new  - a {{personal}} {{2 0 2 0 - 0 9 - 0 8 }} {{1 8 : 0 0 }} {{1 8 : 3
  " {{D entist appointm ent}}"
  ```

kill

Sends a signal to a process, usually related to stopping the process.

All signals except for SIGKILL and SIGSTOP can be intercepted by the process to perform a clean exit.

- Terminate a program using the default SIGTERM (terminate) signal:

  ```
  kill {{process_id}}
  ```

- List available signal names (to be used without the `SIG` prefix):

  ```
  kill -l
  ```

- Terminate a background job:

  ```
  kill %{{job_id}}
  ```

- Terminate a program using the SIGHUP (hang up) signal. Many daemons will reload instead of terminating:

  ```
  kill -{{1|HUP}} {{process_id}}
  ```

- Terminate a program using the SIGINT (interrupt) signal. This is typically initiated by the user pressing `Ctrl+C`:

  ```
  kill -{{2|INT}} {{process_id}}
  ```

- Signal the operating system to immediately terminate a program (which gets no chance to capture the signal):

  ```
  kill -{{9|KILL}} {{process_id}}
  ```

- Signal the operating system to pause a program until a SIGCONT ("continue") signal is received:

  ```
  kill -{{17|STOP}} {{process_id}}
  ```

- Send a `SIGUSR1` signal to all processes with the given GID (group id):

  ```
  kill -{{SIGUSR1}} -{{group_id}}
  ```

killall

Send kill signal to all instances of a process by name (must be exact name).

All signals except SIGKILL and SIGSTOP can be intercepted by the process, allowing a clean exit.

- Terminate a process using the default SIGTERM (terminate) signal:

  `killall {{process_name}}`

- List available signal names (to be used without the 'SIG' prefix):

  `killall --list`

- Interactively ask for confirmation before termination:

  `killall -i {{process_name}}`

- Terminate a process using the SIGINT (interrupt) signal, which is the same signal sent by pressing `Ctrl+C`:

  `killall -INT {{process_name}}`

- Force kill a process:

  `killall -KILL {{process_name}}`

kind

Tool for running local Kubernetes clusters using Docker container "nodes".

Designed for testing Kubernetes itself, but may be used for local development or continuous integration.

More information: https://github.com/kubernetes-sigs/kind.

- Create a local Kubernetes cluster:

  ```
  kind create cluster --name {{cluster_name}}
  ```

- Delete one or more clusters:

  ```
  kind delete clusters {{cluster_name}}
  ```

- Get details about clusters, nodes, or the kubeconfig:

  ```
  kind get {{clusters|nodes|kubeconfig}}
  ```

- Export the kubeconfig or the logs:

  ```
  kind export {{kubeconfig|logs}}
  ```

knife

CLI for interacting with a Chef server from a local Chef repo.

More information: https://docs.chef.io/knife.html.

- Bootstrap a new node:

  ```
  knife bootstrap {{fqdn_or_ip}}
  ```

- List all registered nodes:

  ```
  knife node list
  ```

- Show a node:

  ```
  knife node show {{node_name}}
  ```

- Edit a node:

  ```
  knife node edit {{node_name}}
  ```

- Edit a role:

  ```
  knife role edit {{role_name}}
  ```

- View a data bag:

  ```
  knife data bag show {{data_bag_name}} {{data_bag_item}}
  ```

- Upload a local cookbook to the Chef server:

  ```
  knife cookbook upload {{cookbook_name}}
  ```

kompose

A tool to convert docker-compose applications to Kubernetes.

More information: https://github.com/kubernetes/kompose.

- Deploy a dockeriz ed application to Kubernetes:

  ```
  k om pose u p - f {{dock er- com pose. ym l}}
  ```

- Delete instantiated services/deployments from Kubernetes:

  ```
  k om pose dow n - f {{dock er- com pose. ym l}}
  ```

- Convert a docker-compose file into Kubernetes resources file:

  ```
  k om pose conv ert - f {{dock er- com pose. ym l}}
  ```

kops

Create, destroy, upgrade and maintain Kubernetes clusters from the command-line.

More information: https://github.com/kubernetes/kops/.

- Create a cluster from the configuration specification:

  ```
  k ops create clu ster - f {{clu ster_nam e. yam l}}
  ```

- Create a new ssh public key:

  ```
  k ops create secret sshpu b lick ey {{k ey_nam e}} - i {{~ /. ssh/
  id_rsa. pu b }}
  ```

- E xport the cluster configuration to the ~ /. k u b e/config file

  ```
  k ops export k u b ecfg  {{clu ster_nam e}}
  ```

- G et the cluster configuration as yaml:

  ```
  k ops g et clu ster {{clu ster_nam e}} - o yam l
  ```

- Delete a cluster:

  ```
  k ops delete clu ster {{clu ster_nam e}} - - yes
  ```

kosmorro

Compute the ephemerides and the events for a given date, at a given position on Earth.

More information: http://kosmorro.space.

- Get ephemerides for Paris, France:

```
kosmorro --latitude={{48.7996}} --longitude={{2.3511}}
```

- Get ephemerides for Paris, France, in the UTC+2 timezone:

```
kosmorro --latitude={{48.7996}} --longitude={{2.3511}}
timezone={{2}}
```

- Get ephemerides for Paris, France, on June 9th, 2020:

```
kosmorro --latitude={{48.7996}} --longitude={{2.3511}}
date={{2020-06-09}}
```

- Generate a PDF (note: TeX Live must be installed):

```
kosmorro --format={{pdf}} --output={{path/to/file.pdf}}
```

kotlin

Kotlin Application Launcher.

More information: https://kotlinlang.org.

- Run a jar file:

  ```
  kotlin {{filename.jar}}
  ```

- Display Kotlin and JVM version:

  ```
  kotlin -version
  ```

ksh

Korn Shell, a Bash-compatible command-line interpreter.

See also `history` and `expand` for history expansion.

More information: http://kornshell.com.

- Start an interactive shell session:

  ```
  ksh
  ```

- Execute a command and then exit:

  ```
  ksh -c "{{command}}"
  ```

- Execute a script:

  ```
  ksh {{path/to/script.ksh}}
  ```

- Execute a script, printing each command before executing it:

  ```
  ksh -x {{path/to/script.ksh}}
  ```

kube-capacity

A simple CLI that provides an overview of the resource requests, limits, and utilization in a Kubernetes cluster.

Combine the best parts of `kubectl top` and `kubectl describe` into a CLI focused on cluster resources.

More information: https://github.com/robscott/kube-capacity.

- Output a list of nodes with the total CPU and Memory resource requests and limits:

  ```
  kube-capacity
  ```

- Include pods:

  ```
  kube-capacity -p
  ```

- Include utilization:

  ```
  kube-capacity -u
  ```

kube-fzf

Shell commands for command-line fuzzy searching of Kubernetes Pods.

See also `kubectl` for related commands.

More information: https://github.com/thecasualcoder/kube-fzf

- Get pod details (from current namespace):

  `findpod`

- Get pod details (from all namespaces):

  `findpod -a`

- Describe a pod:

  `describepod`

- Tail pod logs:

  `tailpod`

- Exec into a pod's container:

  `execpod {{shell_command}}`

- Port-forward a pod:

  `pfpod {{port_number}}`

kubeadm

Command-line interface for creating and managing Kubernetes clusters.

More information: https://kubernetes.io/docs/reference/setup-tools/kubeadm.

- Create a Kubernetes master node:

```
kubeadm init
```

- Bootstrap a Kubernetes worker node and join it to a cluster:

```
kubeadm join --token {{token}}
```

- Create a new bootstrap token with a TTL of 12 hours:

```
kubeadm token create --ttl {{12h0m0s}}
```

- Check if the Kubernetes cluster is upgradeable and which versions are available:

```
kubeadm upgrade plan
```

- Upgrade Kubernetes cluster to a specified version:

```
kubeadm upgrade apply {{version}}
```

- View the kubeadm ConfigMap containing the cluster's configuration:

```
kubeadm config view
```

- Revert changes made to the host by 'kubeadm init' or 'kubeadm join':

```
kubeadm reset
```

kubectl describe

Show details of Kubernetes objects and resources.

More information: https://kubernetes.io/docs/reference/generated/kubectl/kubectl-commands#describe

- Show details of pods in a namespace:

  ```
  kubectl describe pods -n {{namespace}}
  ```

- Show details of nodes in a namespace:

  ```
  kubectl describe nodes -n {{namespace}}
  ```

- Show the details of a specific pod in a namespace:

  ```
  kubectl describe pods {{pod_name}} -n {{namespace}}
  ```

- Show the details of a specific node in a namespace:

  ```
  kubectl describe nodes {{node_name}} -n {{namespace}}
  ```

- Show details of Kubernetes objects defined in a YAML manifest:

  ```
  kubectl describe -f {{path/to/manifest}}.yaml
  ```

kubectl get

Get Kubernetes objects and resources.

More information: https://kubernetes.io/docs/reference/generated/kubectl/kubectl-commands#_get

- Get all namespaces in the current cluster:

  ```
  kubectl get namespaces
  ```

- Get nodes in a specified namespace:

  ```
  kubectl get nodes -n {{namespace}}
  ```

- Get pods in a specified namespace:

  ```
  kubectl get pods -n {{namespace}}
  ```

- Get deployments in a specified namespace:

  ```
  kubectl get deployments -n {{namespace}}
  ```

- Get services in a specified namespace:

  ```
  kubectl get services -n {{namespace}}
  ```

- Get Kubernetes objects defined in a YAML manifest:

  ```
  kubectl get -f {{path/to/manifest}}.yaml
  ```

kubectl run

Run pods in Kubernetes. Specifies pod generator to avoid deprecation error in some K8S versions.

More information: https://kubernetes.io/docs/reference/generated/kubectl/kubectl-commands#_run

- Run an nginx pod and expose port 80:

```
kubectl run --generator=run-pod/v1 nginx --image=nginx --
80
```

- Run an nginx pod, setting the TEST_VAR environment variable:

```
kubectl run --generator=run-pod/v1 nginx --image=nginx --
env="TEST_VAR=testing"
```

- Show API calls that would be made to create an nginx container:

```
kubectl run --generator=run-pod/v1 nginx --image=nginx --
run
```

- Run an Ubuntu pod interactively, never restart it, and remove it when it exits:

```
kubectl run --generator=run-pod/v1 -it temp-ubuntu --
image=ubuntu:20.04 --restart=Never --rm --  /bin/bash
```

- Run an Ubuntu pod, overriding the default command with echo, and specifying custom arguments:

```
kubectl run --generator=run-pod/v1 temp-ubuntu --
image=ubuntu:20.04 --command --  echo arg1 arg2 arg3
```

kubectl

Command-line interface for running commands against Kubernetes clusters.

See also `kubectl describe` and other pages for additional information.

More information: https://kubernetes.io/docs/reference/kubectl/.

- List information about a resource with more details:

  `kubectl get {{pod|service|deployment|ingress|...}} -owide`

- Update specified pod with the label 'unhealthy' and the value 'true':

  `kubectl label pods {{name}} unhealthy=true`

- List all resources with different types:

  `kubectl get all`

- Display resource (CPU/Memory/Storage) usage of nodes or pods:

  `kubectl top {{pod|node}}`

- Print the address of the master and cluster services:

  `kubectl cluster-info`

- Display an explanation of a specific field:

  `kubectl explain {{pods.spec.containers}}`

- Print the logs for a container in a pod or specified resource:

  `kubectl logs {{pod_name}}`

- Run command in an existing pod:

  `kubectl exec {{pod_name}} -- {{ls /}}`

kubectx

Utility to manage and switch between `kubectl` contexts.

More information: https://github.com/ahmetb/kubectx.

- List the contexts:

  ```
  kubectx
  ```

- Switch to a named context:

  ```
  kubectx {{name}}
  ```

- Switch to the previous context:

  ```
  kubectx -
  ```

- Delete a named context:

  ```
  kubectx -d {{name}}
  ```

kubens

Utility to switch between Kubernetes namespaces.

More information: https://github.com/ahmetb/kubectx.

- List the namespaces:

  ```
  kubens
  ```

- Change the active namespace:

  ```
  kubens {{name}}
  ```

- Switch to the previous namespace:

  ```
  kubens -
  ```

kubetail

Utility to tail multiple Kubernetes pod logs at the same time.

More information: https://github.com/johanhaleby/kubetail

- Tail the logs of multiple pods (whose name starts with "my_app") in one go:

  ```
  kubetail {{my_app}}
  ```

- Tail only a specific container from multiple pods:

  ```
  kubetail {{my_app}} -c {{my_container}}
  ```

- To tail multiple containers from multiple pods:

  ```
  kubetail {{my_app}} -c {{my_container_1}} -c
  {{my_container_2}}
  ```

- To tail multiple applications at the same time separate them by comma:

  ```
  kubetail {{my_app_1}},{{my_app_2}}
  ```

kustomize

Kustomize is a tool to easily deploy resources for Kubernetes.

More information: https://github.com/kubernetes-sigs/kustomize

- Create kustomization file with resources and namespace:

  ```
  kustomize create --resources {{deployment.yaml,service.yaml}} --namespace {{staging}}
  ```

- Build kustomization file and deploy it with `kubectl`:

  ```
  kustomize build . | kubectl apply -f -
  ```

- Set an image in the kustomization file:

  ```
  kustomize edit set image {{busybox=alpine:3.6}}
  ```

- Search for Kubernetes resources in the current directory to be added to the kustomization file:

  ```
  kustomize create --autodetect
  ```

lambo new

A super-powered `la ra vel  n ew` for Laravel and V alet.

More information: https://github.com/tighten/lambo.

- Create a new Laravel application:

  ```
  lam b o  new  {{app_nam e}}
  ```

- Install the application in a specific path:

  ```
  lam b o  new  - - path= {{path/to/directory}} {{app_nam e}}
  ```

- Include authentication scaffolding:

  ```
  lam b o  new  - - au th {{app_nam e}}
  ```

- Include a specific frontend:

  ```
  lam b o  new  - - {{v u e| b ootstrap| react}} {{app_nam e}}
  ```

- Install npm dependencies after the proj ect has been created:

  ```
  lam b o  new  - - node {{app_nam e}}
  ```

- Create a V alet site after the proj ect has been created:

  ```
  lam b o  new  - - link  {{app_nam e}}
  ```

- Create a new MyS Q L database with the same name as the proj ect:

  ```
  lam b o  new  - - create- db  - - db u ser= {{u ser}} - -
  db passw ord= {{passw ord}} {{app_nam e}}
  ```

- O pen a specific editor after the proj ect has been created:

  ```
  lam b o  new  - - editor= " {{editor}}"  {{app_nam e}}
  ```

lambo

A super-powered `laravel new` for Laravel and Valet.

See also `lambo new` for additional command flags.

More information: https://github.com/tighten/lambo.

- Create a new Laravel application:

  ```
  lambo new {{app_name}}
  ```

- Open the configuration in your default editor:

  ```
  lambo edit-config
  ```

- Open the configuration in a specific editor:

  ```
  lambo edit-config --editor="{{path/to/editor}}"
  ```

- Open the configuration file that is run after new applications have been scaffolded:

  ```
  lambo edit-after
  ```

laravel-z ero

A command-line installer for the Laravel Z ero framework.

More information: https://laravel-z ero.com

- Create a new Laravel Z ero application:

  ```
  larav el- zero new {{nam e}}
  ```

- U pdate the installer to the latest version:

  ```
  larav el- zero self- u pdate
  ```

- List the available installer commands:

  ```
  larav el- zero list
  ```

laravel

A command-line installer for the Laravel framework.

More information: https://laravel.com.

- Create a new Laravel application:

  ```
  larav el new {{nam e}}
  ```

- U se the latest development release:

  ```
  larav el new {{nam e}} - - dev
  ```

- O verwrite if the directory already exists:

  ```
  larav el new {{nam e}} - - force
  ```

- Install the Laravel J etstream scaffolding:

  ```
  larav el new {{nam e}} - - j et
  ```

- Install the Laravel J etstream scaffolding with a specific stack:

  ```
  larav el new {{nam e}} - - j et - - stack {{liv ew ire| inertia}}
  ```

- Install the Laravel J etstream scaffolding with support for teams:

  ```
  larav el new {{nam e}} - - j et - - team s
  ```

- List the available installer commands:

  ```
  larav el list
  ```

last

View the last logged in users.

- View last logins, their duration and other information as read from `/var/log/wtmp`:

  ```
  last
  ```

- Specify how many of the last logins to show:

  ```
  last -n {{logins_count}}
  ```

- Print the full date and time for entries and then display the hostname column last to prevent truncation:

  ```
  last -F -a
  ```

- View all logins by a specific user and show the ip address instead of the hostname:

  ```
  last {{username} -i
  ```

- View all recorded reboots (i.e., the last logins of the pseudo user "reboot"):

  ```
  last reboot
  ```

- View all recorded shutdowns (i.e., the last logins of the pseudo user "shutdown"):

  ```
  last shutdown
  ```

latex

Compile a DVI document from LaTeX source files.

More information: https://www.latex-project.org

- Compile a DVI document:

  ```
  latex {{source.tex}}
  ```

- Compile a DVI document, specifying an output directory:

  ```
  latex -output-directory={{path/to/directory}} {{source.tex}}
  ```

- Compile a DVI document, halting on each error:

  ```
  latex -halt-on-error {{source.tex}}
  ```

latexmk

Compile LaTeX source files into finished documents.

Automatically does multiple runs when needed.

More information: https://mg.readthedocs.io/latexmk.html.

- Compile a DVI (Device Independent file) document from every source:

  ```
  latexmk
  ```

- Compile a DVI document from a specific source file:

  ```
  latexmk {{source.tex}}
  ```

- Compile a pdf document:

  ```
  latexmk -pdf {{source.tex}}
  ```

- Force the generation of a document even if there are errors:

  ```
  latexmk -f {{source.tex}}
  ```

- Clean up temporary tex files created for a specific tex file:

  ```
  latexmk -c {{source.tex}}
  ```

- Clean up all temporary tex files in the current directory:

  ```
  latexmk -c
  ```

ldapsearch

CLI utility for q uerying an LDAP directory.

- Q uery an LDAP server for all items that are a member of the given group and return the obj ect' s displayN ame value:

```
ldapsearch - D ' {{adm in_D N }}' - w ' {{passw ord}}' - h
{{ldap_host}} - b {{b ase_ou }} ' {{m em b erO f= g rou p1 }}'
displayN am e
```

- Q uery an LDAP server with a no-newline password file for all items that are a member of the given group and return the obj ect' s displayN ame value:

```
ldapsearch - D ' {{adm in_D N }}' - y ' {{passw ord_file}}' - h
{{ldap_host}} - b {{b ase_ou }} ' {{m em b erO f= g rou p1 }}'
displayN am e
```

- R eturn 5 items that match the given filter:

```
ldapsearch - D ' {{adm in_D N }}' - w ' {{passw ord}}' - h
{{ldap_host}} - b {{b ase_ou }} ' {{m em b erO f= g rou p1 }}' - z 5
displayN am e
```

- Wait up to 7 seconds for a response:

```
ldapsearch - D ' {{adm in_D N }}' - w ' {{passw ord}}' - h
{{ldap_host}} - b {{b ase_ou }} ' {{m em b erO f= g rou p1 }}' - l 7
displayN am e
```

- Invert the filter:

```
ldapsearch - D ' {{adm in_D N }}' - w ' {{passw ord}}' - h
{{ldap_host}} - b {{b ase_ou }} ' ( ! ( m em b erO f= {{g rou p1 }}) ) '
displayN am e
```

- R eturn all items that are part of multiple groups, returning the display name for each item:

```
ldapsearch - D ' {{adm in_D N }}' - w ' {{passw ord}}' - h
{{ldap_host}} ' ( & ( {{m em b erO f= g rou p1 }}) ( {{m em b erO f= g rou p2
( {{m em b erO f= g rou p3 }}) ) ' " displayN am e"
```

- R eturn all items that are members of at least 1 of the specified groups:

```
ldapsearch -D '{{admin_DN}}' -w '{{password}}' -h
{{ldap_host}} '(|({{member_of=group1}})({{member_of=group1
({{member_of=group3}}))' displayname
```

- Combine multiple boolean logic filters:

```
ldapsearch -D '{{admin_DN}}' -w '{{password}}' -h
{{ldap_host}} '(&({{member_of=group1}})({{member_of=group2
({{member_of=group3}}))' displayname
```

leave

Remind when it's time to leave.

To remove reminders use `kill $(pidof leave)`

- Set a reminder at a given time:

  ```
  leave {{time_to_leave}}
  ```

- Remind to leave at noon:

  ```
  leave {{1200}}
  ```

- Set a reminder in a specific amount of time:

  ```
  leave +{{amount_of_time}}
  ```

- Remind to leave in 4 hours and 4 minutes:

  ```
  leave +{{0404}}
  ```

lebab

A JavaScript modernizer for transpiling code to ES6/ES7.

Transformations must be provided for all examples.

More information: https://github.com/lebab/lebab.

- Display a list of the available transformations:

  ```
  lebab --help
  ```

- Transpile using one or more comma-separated transformations:

  ```
  lebab --transform {{transformation}}
  ```

- Transpile a file to stdout:

  ```
  lebab {{path/to/input_file}}
  ```

- Transpile a file to the specified output file:

  ```
  lebab {{path/to/input_file}} --outfile {{path/to/output_file}}
  ```

- Replace all `.js` files in-place in the specified directory, glob or file:

  ```
  lebab --replace {{directory|glob|file}}
  ```

lein

Manage Clojure projects with declarative configuration.

More information: https://leiningen.org.

- Generate scaffolding for a new project based on a template:

  ```
  lein new {{template-name}} {{project-name}}
  ```

- Start a REPL session either with the project or standalone:

  ```
  lein repl
  ```

- Run the project's `-main` function with optional args:

  ```
  lein run {{args}}
  ```

- Run the project's tests:

  ```
  lein test
  ```

- Package up the project files and all its dependencies into a jar file:

  ```
  lein uberjar
  ```

less

Open a file for interactive reading, allowing scrolling and search.

More information: https://greenwoodsoftware.com/less/.

- Open a file:

  `less {{source_file}}`

- Page down / up:

  `<Space> (down), b (up)`

- Go to end / start of file:

  `G (end), g (start)`

- Forward search for a string (press `n`/`N` to go to next/previous match):

  `/{{something}}`

- Backward search for a string (press `n`/`N` to go to next/previous match):

  `?{{something}}`

- Follow the output of the currently opened file:

  `F`

- Open the current file in an editor:

  `v`

- Exit:

  `q`

lex

Lexical analyser generator.

G iven the specification for a lexical analyser, generates C code implementing it.

- G enerate an analyser from a Lex file:

  ```
  lex {{analyser. l}}
  ```

- S pecify the output file:

  ```
  lex {{analyser. l}} - - ou tfile {{analyser. c}}
  ```

- Compile a C file generated by Lex:

  ```
  cc {{path/to/lex. yy. c}} - - ou tpu t {{execu tab le}}
  ```

license

Create license files for open-source projects.

More information: https://nishanths.github.io/license.

- Print a license to stdout, using the defaults (auto-detected author name, and current year):

  ```
  license {{license_name}}
  ```

- Generate a license and save it to a file:

  ```
  license -o {{filename}} {{license_name}}
  ```

- List all available licenses:

  ```
  license ls
  ```

- Generate a license with custom author name and year:

  ```
  license --name {{author}} --year {{release_year}} {{license_name}}
  ```

light-arionum-cli

The PHP light wallet for the Arionum cryptocurrency.

More information: https://github.com/arionum/lightWalletCLI.

- Generate a new public/private key pair:

  ```
  light-arionum-cli
  ```

- Display the balance of the current address:

  ```
  light-arionum-cli balance
  ```

- Display the balance of the specified address:

  ```
  light-arionum-cli balance {{address}}
  ```

- Send a transaction with an optional message:

  ```
  light-arionum-cli send {{address}} {{value}} {{optional_message}}
  ```

- Export the current wallet information:

  ```
  light-arionum-cli export
  ```

- Display information about the current block:

  ```
  light-arionum-cli block
  ```

- Display information about the current address' transactions:

  ```
  light-arionum-cli transactions
  ```

- Display information about a specific transaction:

  ```
  light-arionum-cli transaction {{transaction_id}}
  ```

lighthouse

Analyzes web applications and web pages, collecting modern performance metrics and insights on developer best practices.

More information: https://github.com/GoogleChrome/lighthouse

- Generate an HTML report for a specific website and save it to a file in the current directory:

  ```
  lighthouse {{https://example.com}}
  ```

- Generate a JSON report and print it:

  ```
  lighthouse --output {{json}} {{https://example.com}}
  ```

- Generate a JSON report and save it to a specific file:

  ```
  lighthouse --output {{json}} --output-path {{path/to/file.json}} {{https://example.com}}
  ```

- Generate a report using the browser in headless mode without logging to stdout:

  ```
  lighthouse --quiet --chrome-flags="{{--headless}}" {{https://example.com}}
  ```

- Generate a report, using the HTTP header key/value pairs in the specified JSON file for all requests:

  ```
  lighthouse --extra-headers={{path/to/file.json}} {{https://example.com}}
  ```

- Generate a report for specific categories only:

  ```
  lighthouse --only-categories={{performance,accessibility,best-practices,seo,pwa}} {{https://example.com}}
  ```

- Generate a report with device emulation and all throttling disabled:

  ```
  lighthouse --screenEmulation.disabled --throttling-method={{provided}} --no-emulatedUserAgent {{https://example.com}}
  ```

- Display help:

  ```
  lighthouse --help
  ```

lilypond

Typeset music and/or produce MIDI from file.

More information: https://lilypond.org.

- Compile a lilypond file into a PDF:

  ```
  lilypond {{path/to/file}}
  ```

- Compile into the specified format:

  ```
  lilypond --formats={{format1,format2}} {{path/to/file}}
  ```

- Compile the specified file, suppressing progress updates:

  ```
  lilypond -s {{path/to/file}}
  ```

- Compile the specified file, and also specify the output filename:

  ```
  lilypond --output={{path/to/output_file}} {{path/to/input_file}}
  ```

- Show the current version of lilypond:

  ```
  lilypond --version
  ```

link

Create a hard link to an existing file.

F or more options, see the `ln` command.

More information: https://www.gnu.org/software/coreutils/link.

- Create a hard link from a new file to an existing file:

```
link {{path/to/existing_file}} {{path/to/new_file}}
```

linkchecker

Command-line client to check HTML documents and websites for broken links.

More information: https://linkchecker.github.io/linkchecker/.

- Find broken links on https://example.com/:

```
linkchecker {{https://example.com/}}
```

- Also check URLs that point to external domains:

```
linkchecker --check-extern {{https://example.com/}}
```

- Ignore URLs that match a specific regular expression:

```
linkchecker --ignore-url {{regular_expression}} {{https://
example.com/}}
```

- Output results to a CSV file:

```
linkchecker --fileout {{csv}}/{{path/to/file}} {{https://
example.com/}}
```

live-server

A simple development http server with live reload capability.

More information: https://www.npmjs.com/package/live-server

- Serve an `index.html` file and reload on changes:

  ```
  live-server
  ```

- Specify a port (default is `8080`) from which to serve a file:

  ```
  live-server --port={{8081}}
  ```

- Specify a given file to serve:

  ```
  live-server --open={{about.html}}
  ```

- Proxy all requests for `ROUTE` to `URL`:

  ```
  live-server --proxy={{/}}:{{http:localhost:3000}}
  ```

lldb

The LLVM Low-Level Debugger.

More information: https://lldb.llvm.org.

- Debug an executable:

```
lldb {{executable}}
```

- Attach `lldb` to a running process with a given PID:

```
lldb -p {{pid}}
```

- Wait for a new process to launch with a given name, and attach to it:

```
lldb -w -n {{process_name}}
```

llvm-dis

Converts LLVM bitcode files into human-readable LLVM Intermediate Representation (IR).

More information: https://www.llvm.org/docs/CommandGuide/llvm-dis.html

- Convert a bitcode file as LLVM IR and write the result to stdout:

```
llvm-dis {{path/to/input.bc}} -o -
```

- Convert a bitcode file to an LLVM IR file with the same filename:

```
llvm-dis {{path/to/file.bc}}
```

- Convert a bitcode file to LLVM IR, writing the result to the specified file:

```
llvm-dis {{path/to/input.bc}} -o {{path/to/output.ll}}
```

lmms

Free, open source, cross-platform digital audio workstation.

Render a `.mmp` or `.mmpz` project file, dump a `.mmpz` as XML, or start the GUI.

More information: https://lmms.io.

- Start the GUI:

  ```
  lmms
  ```

- Start the GUI and load external config:

  ```
  lmms --config {{path/to/config.xml}}
  ```

- Start the GUI and import MIDI or Hydrogen file:

  ```
  lmms --import {{path/to/midi/or/hydrogen/file}}
  ```

- Start the GUI with a specified window size:

  ```
  lmms --geometry {{x_size}}x{{y_size}}+{{x_offset}}+{{y_offset}}
  ```

- Dump a `.mmpz` file:

  ```
  lmms dump {{path/to/mmpz/file.mmpz}}
  ```

- Render a project file:

  ```
  lmms render {{path/to/mmpz_or_mmp/file}}
  ```

- Render the individual tracks of a project file:

  ```
  lmms rendertracks {{path/to/mmpz_or_mmp/file}} {{path/to/dump/directory}}
  ```

- Render with custom samplerate, format, and as a loop:

  ```
  lmms render --samplerate {{8820 0}} --format {{ogg}} --loop output {{path/to/output/file.ogg}}
  ```

ln

Creates links to files and directories.

More information: https://www.gnu.org/software/coreutils/ln.

- Create a symbolic link to a file or directory:

  `ln - s {{/path/to/file_or_directory}} {{path/to/sym link }}`

- O verwrite an existing symbolic link to point to a different file:

  `ln - sf {{/path/to/new _file}} {{path/to/sym link }}`

- Create a hard link to a file:

  `ln {{/path/to/file}} {{path/to/hardlink }}`

loc

Tool written in Rust that counts lines of code.

More information: https://github.com/cgag/loc.

- Print lines of code in the current directory:

  ```
  loc
  ```

- Print lines of code in the target directory:

  ```
  loc {{path/to/directory}}
  ```

- Print lines of code with stats for individual files:

  ```
  loc --files
  ```

- Print lines of code without .gitignore (etc.) files (e.g. two -u flags will additionally count hidden files and dirs):

  ```
  loc -u
  ```

locust

Load-testing tool to determine number of concurrent users a system can handle.

More information: https://locust.io.

- Load-test "example.com" with web interface using locustfile.py:

```
locust --host={{http://example.com}}
```

- Use a different test file:

```
locust --locustfile={{test_file.py}} --host={{http://example.com}}
```

- Run test without web interface, spawning 1 user a second until there are 100 users:

```
locust --no-web --clients={{100}} --hatch-rate={{1}} --host={{http://example.com}}
```

- Start locust in master mode:

```
locust --master --host={{http://example.com}}
```

- Connect locust slave to master:

```
locust --slave --host={{http://example.com}}
```

- Connect locust slave to master on a different machine:

```
locust --slave --master-host={{master_hostname}} --host={{http://example.com}}
```

logname

Shows the user's login name.

More information: https://www.gnu.org/software/coreutils/logname.

- Display the currently logged in user's name:

```
logname
```

logstash

An Elasticsearch ETL (extract, transform and load) tool.

Commonly used to load data from various sources (such as databases and log files) into Elasticsearch.

More information: https://www.elastic.co/products/logstash.

- Check validity of a logstash configuration:

  ```
  logstash --config test --config {{logstash_config.conf}}
  ```

- Run logstash using configuration:

  ```
  sudo logstash --config {{logstash_config.conf}}
  ```

- Run logstash with the most basic inline configuration string:

  ```
  sudo logstash -e 'input {} filter {} output {}'
  ```

lolcat

Put a rainbow in everything you `cat` to the console.

More information: https://github.com/busyloop/lolcat.

- Print a file to the console in rainbow colors:

  `lolcat {{path/to/file}}`

- Print the result of a text-producing command in rainbow colors:

  `{{fortune}} | lolcat`

- Print a file to the console with animated rainbow colors:

  `lolcat -a {{path/to/file}}`

- Print a file to the console with 24-bit (truecolor) rainbow colors:

  `lolcat -t {{path/to/file}}`

lorem

Create more or less random lorem ipsum text.

- Print the specified number of words:

  ```
  lorem -n {{20}}
  ```

- Print 10 lines of Goethe's Faust:

  ```
  lorem -l {{10}} --faust
  ```

- Print 5 sentences of Poe's Raven:

  ```
  lorem -s {{5}} --raven
  ```

- Print 40 random characters from Boccacio's Decameron:

  ```
  lorem --randomize -c {{40}} --decamerone
  ```

lp

Print files.

- Print the output of a command to the default printer (see `lpstat` command):

```
echo "test" | lp
```

- Print a file to the default printer:

```
lp {{path/to/filename}}
```

- Print a file to a named printer (see `lpstat` command):

```
lp -d {{printer_name}} {{path/to/filename}}
```

- Print N copies of file to default printer (replace N with desired number of copies):

```
lp -n {{N}} {{path/to/filename}}
```

- Print only certain pages to the default printer (print pages 1, 3-5, and 16):

```
lp -P 1,3-5,16 {{path/to/filename}}
```

- Resume printing a job:

```
lp -i {{job_id}} -H resume
```

lpass

Command-line interface for the LastPass password manager.

More information: https://github.com/lastpass/lastpass-cli.

- Log in to your LastPass account, by entering your master password when prompted:

  ```
  lpass login {{username}}
  ```

- Show login status:

  ```
  lpass status
  ```

- List all sites grouped by category:

  ```
  lpass ls
  ```

- Generate a new password for gmail.com with the identifier `myinbox` and add to LastPass:

  ```
  lpass generate --username {{username}} --url {{gmail.com}} {{myinbox}} {{password_length}}
  ```

- Show password for a specified entry:

  ```
  lpass show {{myinbox}} --password
  ```

lpr

CUPS tool for printing files.

See also `lpstat` and `lpadmin` for listing and configuring printers.

More information: http://www.CUPS.org

- Print a file to the default printer:

  ```
  lpr {{path/to/file}}
  ```

- Print 2 copies:

  ```
  lpr -# {{2}} {{path/to/file}}
  ```

- Print to a named printer:

  ```
  lpr -P {{printer}} {{path/to/file}}
  ```

- Print either a single page (e.g. 2) or a range of pages (e.g. 2-16):

  ```
  lpr -o page-ranges={{2|2-16}} {{path/to/file}}
  ```

- Print double sided either in portrait (long) or in landscape (short):

  ```
  lpr -o sides={{two-sided-long-edge|two-sided-short-edge}}
  {{path/to/file}}
  ```

- Set page size (more options may be available depending on setup):

  ```
  lpr -o media={{a4|letter|legal}} {{path/to/file}}
  ```

- Print multiple pages per sheet:

  ```
  lpr -o number-up={{2|4|6|9|16}} {{path/to/file}}
  ```

lpstat

S how status information about printers.

- List printers present on the machine and whether they are enabled for printing:

  ```
  lpstat -p
  ```

- S how the default printer:

  ```
  lpstat -d
  ```

- Display all available status information:

  ```
  lpstat -t
  ```

- S how a list of print j obs q ueued by the specified user:

  ```
  lpstat -u {{user}}
  ```

ls

List directory contents.

More information: https://www.gnu.org/software/coreutils/ls.

- List files one per line:

  ```
  ls -1
  ```

- List all files, including hidden files:

  ```
  ls -a
  ```

- List all files, with trailing `/` added to directory names:

  ```
  ls -F
  ```

- Long format list (permissions, ownership, size, and modification date) of all files:

  ```
  ls -la
  ```

- Long format list with size displayed using human readable units (KiB, MiB, GiB):

  ```
  ls -lh
  ```

- Long format list sorted by size (descending):

  ```
  ls -lS
  ```

- Long format list of all files, sorted by modification date (oldest first):

  ```
  ls -ltr
  ```

- Only list directories:

  ```
  ls -d {{*/}}
  ```

lsof

Lists open files and the corresponding processes.

Note: Root privileges (or sudo) is required to list files opened by others.

- Find the processes that have a given file open:

  ```
  lsof {{path/to/file}}
  ```

- Find the process that opened a local internet port:

  ```
  lsof -i :{{port}}
  ```

- Only output the process ID (PID):

  ```
  lsof -t {{path/to/file}}
  ```

- List files opened by the given user:

  ```
  lsof -u {{username}}
  ```

- List files opened by the given command or process:

  ```
  lsof -c {{process_or_command_name}}
  ```

- List files opened by a specific process, given its PID:

  ```
  lsof -p {{PID}}
  ```

- List open files in a directory:

  ```
  lsof +D {{path/to/directory}}
  ```

- Find the process that is listening on a local TCP port:

  ```
  lsof -iTCP:{{port}} -sTCP:LISTEN
  ```

lt

Localtunnel exposes your localhost to the world for easy testing and sharing.

More information: https://github.com/localtunnel/localtunnel.

- Start tunnel from a specific port:

  ```
  lt --port {{8000}}
  ```

- Specify the upstream server doing the forwarding:

  ```
  lt --port {{8000}} --host {{host}}
  ```

- Request a specific subdomain:

  ```
  lt --port {{8000}} --subdomain {{subdomain}}
  ```

- Print basic request info:

  ```
  lt --port {{8000}} --print-requests
  ```

- Open the tunnel URL in the default web browser:

  ```
  lt --port {{8000}} --open
  ```

lua

A powerful, light-weight embeddable programming language.

More information: https://www.lua.org.

- Start an interactive Lua shell:

  ```
  lua
  ```

- Execute a Lua script:

  ```
  lua {{script_name.lua}} {{--optional-argument}}
  ```

- Execute a Lua expression:

  ```
  lua -e '{{print("Hello World")}}'
  ```

luac

Lua bytecode compiler.

More information: https://www.lua.org.

- Compile a Lua source file to Lua bytecode:

  ```
  lu ac - o {{b yte_code. lu ac}} {{sou rce. lu a}}
  ```

- Do not include debug symbols in the output:

  ```
  lu ac - s - o {{b yte_code. lu ac}} {{sou rce. lu a}}
  ```

lumen

A command-line installer for the Lumen micro-framework.

More information: https://lumen.laravel.com.

- Create a new Lumen application:

  ```
  lu m en new  {{application_nam e}}
  ```

- List the available installer commands:

  ```
  lu m en list
  ```

lwp-request

Simple command-line HTTP client.

Built with libwww-perl.

More information: https://metacpan.org/pod/lwp-request

- Make a simple GET request:

```
lwp-request -m GET {{http://example.com/some/path}}
```

- Upload a file with a POST request:

```
lwp-request -m POST {{http://example.com/some/path}} < {{path/to/file}}
```

- Make a request with a custom user agent:

```
lwp-request -H 'User-Agent: {{user_agent}}' -m {{METHOD}} {{http://example.com/some/path}}
```

- Make a request with HTTP authentication:

```
lwp-request -C {{username}}:{{password}} -m {{METHOD}} {{http://example.com/some/path}}
```

- Make a request and print request headers:

```
lwp-request -U -m {{METHOD}} {{http://example.com/some/pa
```

- Make a request and print response headers and status chain:

```
lwp-request -E -m {{METHOD}} {{http://example.com/some/pa
```

lynx

Command-line web browser.

More information: https://lynx.browser.org.

- Visit a website:

  ```
  lynx {{example.com}}
  ```

- Apply restrictions for anonymous account:

  ```
  lynx -anonymous {{example.com}}
  ```

- Turn on mouse support, if available:

  ```
  lynx -use_mouse {{example.com}}
  ```

- Force color mode on, if available:

  ```
  lynx -color {{example.com}}
  ```

- Open a link, using a specific file to read and write cookies:

  ```
  lynx -cookie_file={{path/to/file}} {{example.com}}
  ```

- Navigate forwards and backwards through the links on a page:

  ```
  Up arrow key, Down arrow key
  ```

- Go back to the previously displayed page:

  ```
  Left arrow key or u
  ```

- Exit:

  ```
  q then y
  ```

lz4

Compress or decompress .lz4 files.

More information: https://github.com/lz4/lz4

- Compress a file:

```
lz4 {{file}}
```

- Decompress a file:

```
lz4 -d {{file.lz4}}
```

- Decompress a file and write to stdout:

```
lz4 -dc {{file.lz4}}
```

- Package and compress a directory and its contents:

```
tar cvf - {{path/to/directory}} | lz4 - {{dir.tar.lz4}}
```

- Decompress and unpack a directory and its contents:

```
lz4 -d {{dir.tar.lz4}} | tar -xv
```

- Compress a file using the best compression:

```
lz4 -9 {{file}}
```

lzop

Compress or decompress files with LZO compression.

More information: https://www.lzop.org/

- Compress a file into a new file with the `.lzo` suffix:

  ```
  lzop {{file}}
  ```

- Decompress a file:

  ```
  lzop -d {{file}}.lzo
  ```

- Compress a file, while specifying the compression level. 0 = Worst, 9 = Best (Default level is 3):

  ```
  lzop -{{level}} {{file}}
  ```

m4

Macro processor.

More information: https://www.gnu.org/software/m4.

- Process macros in a file:

```
m4 {{path/to/file}}
```

- Define a macro before processing files:

```
m4 -D {{macro_name}}={{macro_value}} {{path/to/file}}
```

magento

A CLI for managing the Magento PHP framework.

More information: https://magento.com.

- Enable one or more space-separated modules:

  `magento module:enable {{module(s)}}`

- Disable one or more space-separated modules:

  `magento module:disable {{module(s)}}`

- Update the database after enabling modules:

  `magento setup:upgrade`

- Update code and dependency injection configuration:

  `magento setup:di:compile`

- Deploy static assets:

  `magento setup:static-content:deploy`

- Enable maintenance mode:

  `magento maintenance:enable`

- Disable maintenance mode:

  `magento maintenance:disable`

- List all available commands:

  `magento list`

magick

Create, edit, compose, or convert bitmap images.

ImageMagick version 7 + . S **`con vert`** for versions 6 and below.

More information: https://imagemagick.org/.

- Convert file type:

  ```
  m ag ick {{im ag e. png }} {{im ag e. j pg }}
  ```

- R esiz e an image, making a new copy:

  ```
  m ag ick conv ert - resize {{1 0 0 x1 0 0 }} {{im ag e. j pg }}
  {{im ag e. j pg }}
  ```

- Create a G IF using images:

  ```
  m ag ick {{* . j pg }} {{im ag es. g if}}
  ```

- Create checkerboard pattern:

  ```
  m ag ick - size {{6 4 0 x4 8 0 }} pattern: check erb oard
  {{check erb oard. png }}
  ```

- Convert images to individual P DF pages:

  ```
  m ag ick {{* . j pg }} + adj oin {{pag e- % d. pdf}}
  ```

mail

The command operates on the user's mailbox if no argument is given.

To send an email the message body is built from standard input.

- Send a typed email message. The command-line below continues after pressing Enter key. Input CC email-id (optional) press Enter key. Input message text (can be multiline). Press Ctrl-D key to complete the message text:

```
mail --subject="{{subject line}}" {{to_user@example.com}}
```

- Send an email that contains file content:

```
mail --subject="{{$HOSTNAME filename.txt}}"
{{to_user@example.com}} < {{path/to/filename.txt}}
```

- Send a `tar.gz` file as an attachment:

```
tar cvzf - {{path/to/directory1 path/to/directory2}} |
uuencode {{data.tar.gz}} | mail --subject="{{subject_line}}
{{to_user@example.com}}
```

mailx

Send and receive mail.

- Send mail (the content should be typed after the command, and ended with `Ctrl+D`)

  ```
  mailx -s "{{subject}}" {{to_addr}}
  ```

- Send mail with content passed from another command:

  ```
  echo "{{content}}" | mailx -s "{{subject}}" {{to_addr}}
  ```

- Send mail with content read from a file:

  ```
  mailx -s "{{subject}}" {{to_addr}} < {{content.txt}}
  ```

- Send mail to a recipient and CC to another address:

  ```
  mailx -s "{{subject}}" -c {{cc_addr}} {{to_addr}}
  ```

- Send mail specifying the sender address:

  ```
  mailx -s "{{subject}}" -r {{from_addr}} {{to_addr}}
  ```

- Send mail with an attachment:

  ```
  mailx -a {{file}} -s "{{subject}}" {{to_addr}}
  ```

make

Task runner for targets described in Makefile.

Mostly used to control the compilation of an executable from source code.

More information: https://www.gnu.org/software/make/manual/make.html.

- Call the first target specified in the Makefile (usually named "all"):

```
make
```

- Call a specific target:

```
make {{target}}
```

- Call a specific target, executing 4 jobs at a time in parallel:

```
make -j {{4}} {{target}}
```

- Use a specific Makefile:

```
make --file {{file}}
```

- Execute make from another directory:

```
make --directory {{directory}}
```

- Force making of a target, even if source files are unchanged:

```
make --always-make {{target}}
```

- Override variables defined in the Makefile by the environment:

```
make --environment-overrides {{target}}
```

makebuildserver

Create an F -Droid build server virtual machine.

More information: https://f-droid.org/en/docs/B uild_ S erver_ S etup/

- Create a new virtual machine or update an existing one (if available):

  ```
  m ak eb u ildserv er
  ```

- F orce creating a fresh virtual machine:

  ```
  m ak eb u ildserv er - - clean
  ```

makensis

Cross-platform compiler for N S IS installers.

It compiles a N S IS script into a Windows installer executable.

More information: https://nsis.sourceforge.io/Docs/Chapter3 .html

- Compile a N S IS script:

```
m ak ensis {{path/to/file. nsi}}
```

- Compile a N S IS script in strict mode (treat warnings as errors):

```
m ak ensis - W X {{path/to/file. nsi}}
```

- P rint help for a specific command:

```
m ak ensis - C M D H E L P {{com m and}}
```

makepasswd

Generate and encrypt passwords.

More information: https://manpages.debian.org/stretch/makepasswd/makepasswd.1.en.html

- Generate a random password (8 to 10 characters long, containing letters and numbers):

```
makepasswd
```

- Generate a 10 characters long password:

```
makepasswd --chars {{10}}
```

- Generate a 5 to 10 characters long password:

```
makepasswd --minchars {{5}} --maxchars {{10}}
```

- Generate a password containing only the characters "b", "a" or "r":

```
makepasswd --string {{bar}}
```

man

Format and display manual pages.

More information: https://www.man7.org/linux/man-pages/man1/man.1.html

- Display the man page for a command:

  ```
  man {{command}}
  ```

- Display the man page for a command from section 7:

  ```
  man {{command}}.{{7}}
  ```

- Display the path searched for manpages:

  ```
  man --path
  ```

- Display the location of a manpage rather than the manpage itself:

  ```
  man -w {{command}}
  ```

- Display the man page using a specific locale:

  ```
  man {{command}} --locale={{locale}}
  ```

- Search for manpages containing a search string:

  ```
  man -k "{{search_string}}"
  ```

mat2

Anonymise various file formats by removing metadata.

More information: https://0xacab.org/jvoisin/mat2

- List supported file formats:

```
mat2 --list
```

- Remove metadata from a file:

```
mat2 {{path/to/file}}
```

- Remove metadata from a file and print detailed output to the console:

```
mat2 --verbose {{path/to/file}}
```

- Show metadata in a file without removing it:

```
mat2 --show {{path/to/file}}
```

- Partially remove metadata from a file:

```
mat2 --lightweight {{path/to/file}}
```

matlab

Numerical computation environment by MathWorks.

More information: https://uk.mathworks.com/help/matlab/.

- Run without splash screen during startup:

  ```
  matlab -nosplash
  ```

- Execute a MATLAB statement:

  ```
  matlab -r "{{matlab_statement}}"
  ```

- Run a MATLAB script:

  ```
  matlab -r "run({{path/to/script.m}})"
  ```

mc

Midnight Commander, a terminal based file manager.

N avigate the directory structure using the arrow keys, the mouse or by typing the commands into the terminal.

More information: https://midnight-commander.org.

- S tart `m c`:

  ```
  m c
  ```

- S tart `m c` in black and white:

  ```
  m c -b
  ```

md5 sum

Calculate MD5 cryptographic checksums.

More information: https://www.gnu.org/software/coreutils/md5 sum

- Calculate the MD5 checksum for a file:

```
m d5 su m {{filenam e1 }}
```

- Calculate MD5 checksums for multiple files:

```
m d5 su m {{filenam e1 }} {{filenam e2 }}
```

- R ead a file of MD5 S U Ms and verify all files have matching checksums:

```
m d5 su m - c {{filenam e. m d5 }}
```

- Calculate a MD5 checksum from the standard input:

```
echo " {{text}}" | m d5 su m
```

mdp

A command-line based tool to make presentations from Markdown files.

More information: https://github.com/visit1_9_8_5_/mdp

- Launch a presentation in the terminal from a Markdown file:

  ```
  m dp {{presentation. m d}}
  ```

- Disable fading transitions:

  ```
  m dp - - nofade {{presentation. m d}}
  ```

- Invert font colors to use in terminals with light background:

  ```
  m dp - - inv ert {{presentation. m d}}
  ```

- Disable transparency in transparent terminals:

  ```
  m dp - - notrans {{presentation. m d}}
  ```

mediainfo

Display metadata from video and audio files.

More information: https://mediaarea.net/MediaInfo.

- Display metadata for a given file in the console:

```
m ediainfo {{file}}
```

- S tore the output to a given file along with displaying in the console:

```
m ediainfo - - L og file= {{ou t. txt}} {{file}}
```

- Display the list of metadata attributes that can be extracted:

```
m ediainfo - - I nfo- P aram eters
```

meld

Graphical diffing and merging tool.

More information: https://meldmerge.org/.

- Start meld:

```
meld
```

- Compare 2 files:

```
meld {{path/to/file_1}} {{path/to/file_2}}
```

- Compare 2 directories:

```
meld {{path/to/directory_1}} {{path/to/directory_2}}
```

- Compare 3 files:

```
meld {{path/to/file_1}} {{path/to/file_2}} {{path/to/file_3}}
```

- Open a comparison as a new tab in a pre-existing meld instance:

```
meld --newtab {{path/to/file_1}} {{path/to/file_2}}
```

- Compare multiple sets of files:

```
meld --diff {{path/to/file_1}} {{path/to/file_2}} --diff
{{path/to/file_3}} {{path/to/file_4}}
```

mesg

Check or set a terminal's ability to receive messages from other users, usually from the write command.

See also **write**.

- Check terminal's openness to write messages:

  ```
  mesg
  ```

- Disable receiving messages from the write command:

  ```
  mesg n
  ```

- Enable receiving messages from the write command:

  ```
  mesg y
  ```

meshlabserver

Command-line interface for the MeshLab 3D mesh processing software.

- Convert an STL file to an OBJ file:

  ```
  meshlabserver -i {{input.stl}} -o {{output.obj}}
  ```

- Convert a WRL file to a OFF file, including the vertex and face normals in the output mesh:

  ```
  meshlabserver -i {{input.wrl}} -o {{output.off}} -om vn fn
  ```

- Dump a list of all the available processing filters into a file:

  ```
  meshlabserver -d {{filename}}
  ```

- Process a 3D file using a filter script created in the MeshLab GUI (Filters > Show current filter script > Save Script):

  ```
  meshlabserver -i {{input.ply}} -o {{output.ply}} -s {{filter_script.mlx}}
  ```

- Process a 3D file using a filter script, writing the output of the filters into a log file:

  ```
  meshlabserver -i {{input.x3d}} -o {{output.x3d}} -s {{filter_script.mlx}} -l {{log file}}
  ```

meson

S Cons-like build system that uses python as a front-end language and N inj a as a building backend.

More information: https://mesonbuild.com.

- G enerate a c proj ect with a given name and version:

  ```
  m eson init - - lang u ag e= {{c}} - - nam e= {{m yproj ect}} - -
  v ersion= {{0 . 1 }}
  ```

- Configure the `b u ilddir` with default values:

  ```
  m eson setu p {{b u ild_dir}}
  ```

- B uild the proj ect:

  ```
  m eson com pile - C {{path/to/b u ild_dir}}
  ```

- S how the help:

  ```
  m eson - - help
  ```

- S how version info:

  ```
  m eson - - v ersion
  ```

meteor

Full-stack JavaScript platform for building web applications.

More information: https://meteor.com.

- Run a meteor project from its root directory in development mode:

  ```
  meteor
  ```

- Create a project under the given directory:

  ```
  meteor create {{path/to/directory}}
  ```

- Display the list of packages the project is currently using:

  ```
  meteor list
  ```

- Add a package to the project:

  ```
  meteor add {{package_name}}
  ```

- Remove a package from the project:

  ```
  meteor remove {{package_name}}
  ```

- Create a production build of the project as a tarball under the given directory:

  ```
  meteor build {{path/to/directory}}
  ```

micro

Micro is a modern and intuitive terminal-based text editor.

Y ou can use your keyboard, but also your mouse to navigate and/or select text.

More information: https://micro-editor.github.io.

- O pen a file:

  ```
  m icro {{file}}
  ```

- Cut the entire line:

  ```
  C trl + K
  ```

- S earch for a pattern in the file (press `C trl + N` / `C trl + P` to go to next/previous match):

  ```
  C trl + F " {{pattern}}" < E nter>
  ```

- E xecute a command:

  ```
  C trl + E {{com m and}} < E nter>
  ```

- P erform a substitution in the whole file:

  ```
  C trl + E replaceall " {{pattern}}" " {{replacem ent}}" < E nter>
  ```

- Q uit:

  ```
  C trl + Q
  ```

middleman

Static site generator written in Ruby.

More information: https://middlemanapp.com/.

- Create a new Middleman project:

  ```
  middleman init "{{project_name}}"
  ```

- Start local server for current project on port 4567:

  ```
  middleman server
  ```

- Start local server for current project on a specified port:

  ```
  middleman server -p "{{port}}"
  ```

- Build the project in the current directory to prepare for deployment:

  ```
  bundle exec middleman build
  ```

- Deploy the Middleman project in the current directory:

  ```
  middleman deploy
  ```

minetest

Multiplayer infinite-world block sandbox.

See also `minetest server`, the server-only binary.

More information: https://wiki.minetest.net/Minetest.

- Start minetest in client mode:

  ```
  minetest
  ```

- Start minetest in server mode by hosting a specific world:

  ```
  minetest --server --world {{name}}
  ```

- Write logs to a specific file:

  ```
  minetest --logfile {{path/to/file}}
  ```

- Only write errors to the console:

  ```
  minetest --quiet
  ```

minetestserver

Multiplayer infinite-world block sandbox server.

See also `minetest`, the graphical client.

More information: https://wiki.minetest.net/Setting_up_a_server

- Start the server:

  `minetestserver`

- List available worlds:

  `minetestserver --world list`

- Specify the world name to load:

  `minetestserver --world {{world_name}}`

- List the available game IDs:

  `minetestserver --gameid list`

- Specify a game to use:

  `minetestserver --gameid {{game_id}}`

- Listen on a specific port:

  `minetestserver --port {{3456 7}}`

- Migrate to a different data backend:

  `minetestserver --migrate {{sqlite3 | leveldb | redis}}`

- Start an interactive terminal after starting the server:

  `minetestserver --terminal`

minifab

Utility tool that automates the setup and deployment of Hyperledger Fabric networks.

More information: https://github.com/hyperledger-labs/minifabric.

- Bring up the default Hyperledger Fabric network:

  ```
  minifab up -i {{minifab_version}}
  ```

- Bring down the Hyperledger Fabric network:

  ```
  minifab down
  ```

- Install chaincode onto a specified channel:

  ```
  minifab install -n {{chaincode_name}}
  ```

- Install a specific chaincode version onto a channel:

  ```
  minifab install -n {{chaincode_name}} -v
  {{chaincode_version}}
  ```

- Initialize the chaincode after installation/upgrade:

  ```
  minifab approve,commit,initialize,discover
  ```

- Invoke a chaincode method with the specified arguments:

  ```
  minifab invoke -n {{chaincode_name}} -p '"{{method_name}}",
  "{{arg0}}", "{{arg1}}", ...'
  ```

- Make a query on the ledger:

  ```
  minifab blockquery {{block_number}}
  ```

- Quickly run an application:

  ```
  minifab apprun -l {{application_language}}
  ```

minikube

Tool to run Kubernetes locally.

More information: https://github.com/kubernetes/minikube.

- Start the cluster:

  ```
  minikube start
  ```

- Get the IP address of the cluster:

  ```
  minikube ip
  ```

- Access a service named my_service exposed via a node port and get the URL:

  ```
  minikube service {{my_service}} --url
  ```

- Open the Kubernetes dashboard in a browser:

  ```
  minikube dashboard
  ```

- Stop the running cluster:

  ```
  minikube stop
  ```

- Delete the cluster:

  ```
  minikube delete
  ```

minisign

A dead simple tool to sign files and verify signatures.

More information: https://jedisct1.github.io/minisign/

- Generate a new keypair at the default location:

```
minisign -G
```

- Sign a file:

```
minisign -Sm {{path/to/file}}
```

- Sign a file, adding a trusted (signed) and an untrusted (unsigned) comment in the signature:

```
minisign -Sm {{path/to/file}} -c "{{untrusted comment}}" -t "{{trusted comment}}"
```

- Verify a file and the trusted comments in its signature using the specified public key file:

```
minisign -Vm {{path/to/file}} -p {{path/to/public key.pub}}
```

- Verify a file and the trusted comments in its signature, specifying a public key as a Base64 encoded literal:

```
minisign -Vm {{path/to/file}} -P "{{public_key_base64}}"
```

mitmdump

View, record, and programmatically transform HTTP traffic.

The command-line counterpart to mitmproxy.

More information: https://docs.mitmproxy.org/stable/overview-tools/#_mitmdump

- Start a proxy and save all output to a file:

  ```
  mitmdump -w {{filename}}
  ```

- Filter a saved traffic file to just POST requests:

  ```
  mitmdump -nr {{input_filename}} -w {{output_filename}} "{{~m post}}"
  ```

- Replay a saved traffic file:

  ```
  mitmdump -nc {{filename}}
  ```

mitmproxy

An interactive man-in-the-middle HTTP proxy.

More information: https://mitmproxy.org.

- Start mitmproxy with default settings:

  ```
  mitmproxy
  ```

- Start mitmproxy bound to custom address and port:

  ```
  mitmproxy -b {{ip_address}} -p {{port}}
  ```

mix

Mix is a build tool that provides tasks for creating, compiling, and testing Elixir projects, managing its dependencies, and more.

More information: https://hexdocs.pm/mix.

- Execute a particular file:

  ```
  mix run {{my_script.exs}}
  ```

- Create a new project:

  ```
  mix new {{project_name}}
  ```

- Compile project:

  ```
  mix compile
  ```

- Run project tests:

  ```
  mix test
  ```

- List all mix commands:

  ```
  mix help
  ```

mixxx

Free and open source cross-platform DJ software.

More information: https://mixxx.org/manual/latest/chapters/appendix.html#_command-line-options.

- Start the Mixxx GUI in fullscreen:

  ```
  mixxx --fullscreen
  ```

- Start in safe developer mode to debug a crash:

  ```
  mixxx --developer --safemode
  ```

- Debug a malfunction:

  ```
  mixxx --debug Assertbreak --developer --log level trace
  ```

- Start mixxx using the specified settings file:

  ```
  mixxx --resourcepath {{mixxx/res/controllers}} --settingspath {{path/to/settings-file}}
  ```

- Debug a custom controller mapping:

  ```
  mixxx --controllerDebug --resourcepath {{path/to/mapping-directory}}
  ```

- Show command-line help:

  ```
  mixxx --help
  ```

mkcert

T ool for making locally-trusted development certificates.

More information: https://github.com/F iloS ottile/mkcert

- Install the local CA in the system trust store:

  ```
  m k cert - install
  ```

- G enerate certificate and private key for a given domain:

  ```
  m k cert {{exam ple. org }}
  ```

- G enerate certificate and private key for multiple domains:

  ```
  m k cert {{exam ple. org }} {{m yapp. dev }} {{1 2 7 . 0 . 0 . 1 }}
  ```

- G enerate wildcard certificate and private key for a given domain and its subdomains:

  ```
  m k cert " {{* . exam ple. it}}"
  ```

- U ninstall the local CA:

  ```
  m k cert - u ninstall
  ```

mkdir

Creates a directory.

More information: https://www.gnu.org/software/coreutils/mkdir.

- Create a directory in current directory or given path:

  ```
  m k dir {{directory}}
  ```

- Create directories recursively (useful for creating nested dirs):

  ```
  m k dir - p {{path/to/directory}}
  ```

mkfifo

Makes F IF O s (named pipes).

More information: https://www.gnu.org/software/coreutils/mkfifo.

- Create a named pipe at a given path:

```
m k fifo {{path/to/pipe}}
```

mktemp

Create a temporary file or directory.

More information: https://www.gnu.org/software/autogen/mktemp.html.

- Create an empty temporary file and return the absolute path to it:

```
m k tem p
```

- Create a temporary directory and return the absolute path to it:

```
m k tem p - d
```

- Create a temporary file with a specified suffix:

```
m k tem p - - su ffix " {{. txt}}"
```

mktorrent

A CLI utility to create BitTorrent metainfo files.

More information: https://github.com/Rudde/mktorrent

- Create a torrent with 2^21 KB as the piece size:

  ```
  mktorrent -a {{tracker_announce_url}} -l {{21}} -o {{path/to/example.torrent}} {{path/to/file_or_directory}}
  ```

- Create a private torrent with a 2^21 KB piece size:

  ```
  mktorrent -p -a {{tracker_announce_url}} -l {{21}} -o {{path/to/example.torrent}} {{path/to/file_or_directory}}
  ```

- Create a torrent with a comment:

  ```
  mktorrent -c "{{comment}}" -a {{tracker_announce_url}} -l {{21}} -o {{path/to/example.torrent}} {{path/to/file_or_directory}}
  ```

- Create a torrent with multiple trackers:

  ```
  mktorrent -a {{tracker_announce_url,tracker_announce_url_2}} -l {{21}} -o {{path/to/example.torrent}} {{path/to/file_or_directory}}
  ```

- Create a torrent with web seed URLs:

  ```
  mktorrent -a {{tracker_announce_url}} -w {{web_seed_url}} -l {{21}} -o {{path/to/example.torrent}} {{path/to/file_or_directory}}
  ```

mkvmerge

Merge and extract multimedia streams.

More information: https://mkvtoolnix.download/doc/mkvmerge.html.

- Display information about a Matroska file:

```
mkvmerge --identify {{path/to/file.mkv}}
```

- Extract the audio from track 1 of a specific file:

```
mkvextracttracks {{path/to/file.mkv}} {{1}}:{{path/to/output.webm}}
```

- Extract the subtitle from track 3 of a specific file:

```
mkvextracttracks {{path/to/file.mkv}} {{3}}:{{path/to/subs.srt}}
```

mlr

Miller is like `awk`, `sed`, `cut`, `join`, and `sort` for name-indexed data such as CSV, TSV, and tabular JSON.

More information: https://johnkerl.org/miller/doc

- Pretty-print a CSV file in a tabular format:

```
mlr --icsv --opprint cat {{example.csv}}
```

- Receive JSON data and pretty print the output:

```
echo '{"hello": "world"}' | mlr --ijson --opprint cat
```

- Sort alphabetically on a field:

```
mlr --icsv --opprint sort -f {{field}} {{example.csv}}
```

- Sort in descending numerical order on a field:

```
mlr --icsv --opprint sort -nr {{field}} {{example.csv}}
```

- Convert CSV to JSON, performing calculations and display those calculations:

```
mlr --icsv --ojson put '$ {{newField1}} = $ {{oldFieldA}}/$
{{oldFieldB}}' {{example.csv}}
```

- Receive JSON and format the output as vertical JSON:

```
echo '{"hello": "world", "foo": "bar"}' | mlr --ijson --o
--jvstack cat
```

- Filter lines of a compressed CSV file treating numbers as strings:

```
mlr --prepipe 'gunzip' --csv filter -S '$ {{fieldName}} =~
"{{regular_expression}}"' {{example.csv.gz}}
```

mm2gv

Convert a graph from Matrix Market **mm** format to **gv** format.

Converters: **gml2gv**, **gv2gml**, **gv2gxl**, **gxl2gv**, **dot2gv**, & **mm2gv**.

More information: https://graphviz.org/pdf/mm2gv.1.pdf

- Convert a graph from mm to gv format:

  ```
  mm2gv -o {{output.gv}} {{input.mm}}
  ```

- Convert a graph using stdin and stdout:

  ```
  cat {{input.mm}} | mm2gv > {{output.gv}}
  ```

- Display help:

  ```
  mm2gv -?
  ```

mmdc

CLI for mermaid, a diagram generation tool with a domain-specific language.

A mermaid definition file is taken as input and a SVG, PNG, or PDF file is generated as output.

More information: https://mermaid-js.github.io/mermaid/

- Convert a file to the specified format (automatically determined from the file extension):

  ```
  mmdc --input {{input.mmd}} --output {{output.svg}}
  ```

- Specify the theme of the chart:

  ```
  mmdc --input {{input.mmd}} --output {{output.svg}} --theme
  {{forest|dark|neutral|default}}
  ```

- Specify the background color of the chart (e.g. `lime`, `#D8064F`, or `transparent`):

  ```
  mmdc --input {{input.mmd}} --output {{output.svg}} --
  backgroundColor {{color}}
  ```

mmls

Display the partition layout of a volume system.

More information: https://wiki.sleuthkit.org/index.php?_title=_Mmls

- Display the partition table stored in an image file:

```
mmls {{path/to/image_file}}
```

- Display the partition table with an additional column for the partition size:

```
mmls -B -i {{path/to/image_file}}
```

- Display the partition table in a split EWF image:

```
mmls -i ewf {{image.e01}} {{image.e02}}
```

- Display nested partition tables:

```
mmls -t {{nested_table_type}} -o {{offset}} {{path/to/image_file}}
```

mmv

Move and rename files in bulk.

- Rename all files with a certain extension to a different extension:

  ```
  mmv "*{{.old_extension}}" "#1{{.new_extension}}"
  ```

- Copy `report6part4.txt` to `./french/rapport6partie4.txt` along with all similarly named files:

  ```
  mmv -c "{{report*part*.txt}}" "{{./french/rapport#1partie#2.txt}}"
  ```

- Append all `.txt` files into one file:

  ```
  mmv -a "{{*.txt}}" "{{all.txt}}"
  ```

- Convert dates in filenames from "M-D-Y" format to "D-M-Y" format:

  ```
  mmv "{{[0-1][0-9]-[0-3][0-9]-[0-9][0-9][0-9
  " {{#3#4-#1#2-#5#6#7#8.txt}}"
  ```

mocha

Execute Mocha JavaScript test runner.

More information: https://mochajs.org

- Run tests with default configuration or as configured in `mocha.opts`:

  `mocha`

- Run tests contained at a specific location:

  `mocha {{directory/with/tests}}`

- Run tests that match a specific grep pattern:

  `mocha --grep {{regular_expression}}`

- Run tests on changes to JavaScript files in the current directory and once initially:

  `mocha --watch`

- Run tests with a specific reporter:

  `mocha --reporter {{reporter}}`

mogrify

Perform operations on multiple images, such as resizing, cropping, flipping, and adding effects.

Changes are applied directly to the original file.

More information: https://imagemagick.org/script/mogrify.php.

- Resize all JPEG images in the directory to 50% of their initial size:

```
mogrify -resize {{50%}} {{*.jpg}}
```

- Resize all images starting with "DSC" to 800x600:

```
mogrify -resize {{800x600}} {{DSC*}}
```

- Convert all PNG images in the directory to JPEG:

```
mogrify -format {{jpg}} {{*.png}}
```

- Halve the saturation of all image files in the current directory:

```
mogrify -modulate {{100,50}} {{*}}
```

- Double the brightness of all image files in the current directory:

```
mogrify -modulate {{200}} {{*}}
```

molecule

Molecule helps testing ansible roles.

More information: https://molecule.readthedocs.io.

- Create a new ansible role:

  ```
  m olecu le init role - - role- nam e {{role_nam e}}
  ```

- R un tests:

  ```
  m olecu le test
  ```

- S tart the instance:

  ```
  m olecu le create
  ```

- Configure the instance:

  ```
  m olecu le conv erg e
  ```

- Log in into the instance:

  ```
  m olecu le log in
  ```

mongo

MongoDB interactive shell client.

More information: https://docs.mongodb.com/manual/reference/program/mongo.

- Connect to a database:

```
mongo {{database}}
```

- Connect to a database running on a given host on a given port:

```
mongo --host {{host}} --port {{port}} {{database}}
```

- Connect to a database with a given username; user will be prompted for password:

```
mongo --username {{username}} {{database}} --password
```

- Evaluate a JavaScript expression on the database:

```
mongo --eval '{{JSON.stringify(db.foo.findOne())}}'
{{database}}
```

mongod

The MongoDB database server.

More information: https://docs.mongodb.com/manual/reference/program/mongod.

- Specify a config file:

```
mongod --config {{filename}}
```

- Specify the port to listen on:

```
mongod --port {{port}}
```

- Specify database profiling level. 0 is off, 1 is only slow operations, 2 is all:

```
mongod --profile {{0 | 1 | 2}}
```

mongodump

Utility to export the contents of a MongoDB instance.

More information: https://docs.mongodb.com/database-tools/mongodump/.

- Create a dump of all databases (this will place the files inside a directory called "dump"):

```
mongodump
```

- Specify an output location for the dump:

```
mongodump --out {{path/to/directory}}
```

- Create a dump of a given database:

```
mongodump --db {{database_name}}
```

- Create a dump of a given collection within a given database:

```
mongodump --collection {{collection_name}} --db {{database_name}}
```

- Connect to a given host running on a given port, and create a dump:

```
mongodump --host {{host}} --port {{port}}
```

- Create a dump of a given database with a given username; user will be prompted for password:

```
mongodump --username {{username}} {{database}} --password
```

mongoexport

Produce exports of data stored in a MongoDB instance formatted as JSON or CSV.

More information: https://docs.mongodb.com/database-tools/mongoexport/.

- Export a collection to stdout, formatted as JSON:

  ```
  mongoexport --uri={{connection_string}} --
  collection={{collection_name}}
  ```

- Export the documents in the specified collection that match a query to a JSON file:

  ```
  mongoexport --db={{database_name}} --
  collection={{collection_name}} --query="{{query_object}}" --
  out={{path/to/file.json}}
  ```

- Export documents as a JSON array instead of one object per line:

  ```
  mongoexport --collection={{collection_name}} --jsonArray
  ```

- Export documents to a CSV file:

  ```
  mongoexport --collection={{collection_name}} --type={{csv}}
  --fields="{{field1,field2,...}}" --out={{path/to/file.csv}
  ```

- Export documents that match the query in the specified file to a CSV file, omitting the list of field names on the first line:

  ```
  mongoexport --collection={{collection_name}} --type={{csv}}
  --fields="{{field1,field2,...}}" --queryFile={{path/to/fil
  --noHeaderLine --out={{path/to/file.csv}}
  ```

- Export documents to stdout, formatted as human-readable JSON:

  ```
  mongoexport --uri={{mongodb_uri}} --
  collection={{collection_name}} --pretty
  ```

- Display help:

  ```
  mongoexport --help
  ```

mongoimport

Imports content from a JSON, CSV, or TSV file into a MongoDB database.

More information: https://docs.mongodb.com/database-tools/mongoimport/.

- Import a JSON file into a specific collection:

```
mongoimport --file={{path/to/file.json}} --
uri={{mongodb_uri}} --collection={{collection_name}}
```

- Import a CSV file, using the first line of the file to determine field names:

```
mongoimport --type={{csv}} --file={{path/to/file.csv}} --
db={{database_name}} --collection={{collection_name}}
```

- Import a JSON array, using each element as a separate document:

```
mongoimport --jsonArray --file={{path/to/file.json}}
```

- Import a JSON file using a specific mode and a query to match existing documents:

```
mongoimport --file={{path/to/file.json}} --mode={{delete|
merge|upsert}} --upsertFields="{{field1,field2,...}}"
```

- Import a CSV file, reading field names from a separate CSV file and ignoring fields with empty values:

```
mongoimport --type={{csv}} --file={{path/to/file.csv}} --
fieldFile={{path/to/field_file.csv}} --ignoreBlanks
```

- Display help:

```
mongoimport --help
```

mongorestore

Utility to import a collection or database from a binary dump into a MongoDB instance.

More information: https://docs.mongodb.com/database-tools/mongorestore/.

- Import a bson data dump from a directory to a MongoDB database:

  ```
  mongorestore --db {{database name}} {{path/to/directory}}
  ```

- Import a bson data dump from a directory to a given database in a MongoDB server host, running at a given port, with user authentication (user will be prompted for password):

  ```
  mongorestore --host {{database_host:port}} --db {{database name}} --u sernam e {{u sernam e}} {{path/to/directory}} --passw ord
  ```

- Import a collection from a bson file to a MongoDB database:

  ```
  mongorestore --db {{database name}} {{path/to/file}}
  ```

- Import a collection from a bson file to a given database in a MongoDB server host, running at a given port, with user authentication (user will be prompted for password):

  ```
  mongorestore --host {{database_host:port}} --db {{database name}} --u sernam e {{u sernam e}} {{path/to/file}} --passw ord
  ```

monodevelop

Cross platform IDE for C#, F# and more.

More information: https://www.monodevelop.com/.

- Start MonoDevelop:

  ```
  monodevelop
  ```

- Open a specific file:

  ```
  monodevelop {{path/to/file}}
  ```

- Open a specific file with the caret at a specific position:

  ```
  monodevelop {{path/to/file}}; {{line_number}}; {{column_number}}
  ```

- Force opening a new window instead of switching to an existing one:

  ```
  monodevelop --new-window
  ```

- Disable redirection of stdout and stderr to a log file:

  ```
  monodevelop --no-redirect
  ```

- Enable performance monitoring:

  ```
  monodevelop --perf-log
  ```

monodis

The Mono Common Intermediate Language (CIL) disassembler.

More information: https://www.mono-project.com/docs/tools+libraries/tools/monodis/

- Disassemble an assembly to textual CIL:

```
monodis {{path/to/assembly.exe}}
```

- Save the output to a file:

```
monodis --output={{path/to/output.il}} {{path/to/assembly.exe}}
```

- Show information about an assembly:

```
monodis --assembly {{path/to/assembly.dll}}
```

- List the references of an assembly:

```
monodis --assemblyref {{path/to/assembly.exe}}
```

- List all the methods in an assembly:

```
monodis --method {{path/to/assembly.exe}}
```

- Show a list of resources embedded within an assembly:

```
monodis --manifest {{path/to/assembly.dll}}
```

- Extract all the embedded resources to the current directory:

```
monodis --mresources {{path/to/assembly.dll}}
```

monop

Finds and displays signatures of Types and methods inside .NET assemblies.

- Show the structure of a Type built-in of the .NET Framework:

  ```
  monop {{System.String}}
  ```

- List the types in an assembly:

  ```
  monop -r:{{path/to/assembly.exe}}
  ```

- Show the structure of a Type in a specific assembly:

  ```
  monop -r:{{path/to/assembly.dll}} {{Namespace.Path.To.Type}}
  ```

- Only show members defined in the specified Type:

  ```
  monop -r:{{path/to/assembly.dll}} --only-declared
  {{Namespace.Path.To.Type}}
  ```

- Show private members:

  ```
  monop -r:{{path/to/assembly.dll}} --private
  {{Namespace.Path.To.Type}}
  ```

- Hide obsolete members:

  ```
  monop -r:{{path/to/assembly.dll}} --filter-obsolete
  {{Namespace.Path.To.Type}}
  ```

- List the other assemblies that a specified assembly references:

  ```
  monop -r:{{path/to/assembly.dll}} --refs
  ```

montage

ImageMagick image montage tool.

Tiles images into a customisable grid.

More information: https://imagemagick.org/script/montage.php.

- Tile images into a grid, automatically resizing images larger than the grid cell size:

  ```
  montage {{image1.png}} {{image2.jpg}} {{imageN.png}} montage.jpg
  ```

- Tile images into a grid, automatically calculating the grid cell size from the largest image:

  ```
  montage {{image1.png}} {{image2.jpg}} {{imageN.png}} -geometry +0+0 montage.jpg
  ```

- Set the grid cell size and resize images to fit it before tiling:

  ```
  montage {{image1.png}} {{image2.jpg}} {{imageN.png}} -geometry 640x480+0+0 montage.jpg
  ```

- Limit the number of rows and columns in the grid, causing input images to overflow into multiple output montages:

  ```
  montage {{image1.png}} {{image2.jpg}} {{imageN.png}} -geometry +0+0 -tile 2x3 montage_%d.jpg
  ```

- Resize and crop images to completely fill their grid cells before tiling:

  ```
  montage {{image1.png}} {{image2.jpg}} {{imageN.png}} -geometry +0+0 -resize 640x480^ -gravity center -crop 640x480+0+0 montage.jpg
  ```

more

Open a file for interactive reading, allowing scrolling and search.

More information: https://manned.org/more.

- Open a file:

  ```
  more {{path/to/file}}
  ```

- Open a file displaying from a specific line:

  ```
  more +{{line_number}} {{path/to/file}}
  ```

- Display help:

  ```
  more --help
  ```

- Go to the next page:

  ```
  <Space>
  ```

- Search for a string (press `n` to go to the next match):

  ```
  /{{something}}
  ```

- Exit:

  ```
  q
  ```

- Display help about interactive commands:

  ```
  h
  ```

moro

T rack work time.

More information: https://moro.j s.org

- Invoke `m oro` without parameters, to set the current time as the start of the working day:

  ```
  m oro
  ```

- S pecify a custom time for the start of the working day:

  ```
  m oro hi {{0 9 : 3 0 }}
  ```

- Invoke `m oro` without parameters a second time, to set the current time at the end of the working day:

  ```
  m oro
  ```

- S pecify a custom time for the end of the working day:

  ```
  m oro b ye {{1 7 : 3 0 }}
  ```

- Add a note on the current working day:

  ```
  m oro note {{3  hou rs on proj ect F oo}}
  ```

- S how a report of time logs and notes for the current working day:

  ```
  m oro report
  ```

- S how a report of time logs and notes for all working days on record:

  ```
  m oro report - - all
  ```

mosh

Mobile Shell (`mosh`) is a robust and responsive replacement for SSH.

`mosh` persists connections to remote servers while roaming between networks.

More information: https://mosh.org.

- Connect to a remote server:

  ```
  mosh {{username}}@{{remote_host}}
  ```

- Connect to a remote server with a specific identity (private key):

  ```
  mosh --ssh="ssh -i {{path/to/key_file}}" {{username}}@{{remote_host}}
  ```

- Connect to a remote server using a specific port:

  ```
  mosh --ssh="ssh -p {{2222}}" {{username}}@{{remote_host}}
  ```

- Run a command on a remote server:

  ```
  mosh {{remote_host}} -- {{command and -with -flags}}
  ```

- Select Mosh UDP port (useful when `{{remote_host}}` is behind a NAT):

  ```
  mosh -p {{124}} {{username}}@{{remote_host}}
  ```

- Usage when `mosh-server` binary is outside standard path:

  ```
  mosh --server={{path/to/bin/}}mosh-server {{remote_host}}
  ```

mosq uitto

An MQ T T broker.

More information: https://mosq uitto.org/

- S tart mosq uitto:

  ```
  m osq u itto
  ```

- S pecify a configuration file to use:

  ```
  m osq u itto - - config - file {{path/to/file. conf}}
  ```

- Listen on a specific port:

  ```
  m osq u itto - - port {{8 8 8 3 }}
  ```

- Daemoniz e by forking into the background:

  ```
  m osq u itto - - daem on
  ```

mosquitto_passwd

Manage password files for mosquitto.

See also `mosquitto`, the MQTT server that this manages.

More information: https://mosquitto.org/man/mosquitto_passwd-1.html

- Add a new user to a password file (will prompt to enter the password):

  `mosquitto_passwd {{path/to/password_file}} {{username}}`

- Create the password file if it doesn't already exist:

  `mosquitto_passwd -c {{path/to/password_file}} {{username}}`

- Delete the specified username instead:

  `mosquitto_passwd -D {{path/to/password_file}} {{username}}`

- Upgrade an old plain-text password file to a hashed password file:

  `mosquitto_passwd -U {{path/to/password_file}}`

mosq uitto_ pub

A simple MQ T T version 3 .1 .1 client that will publish a single message on a topic and exit.

More information: https://mosq uitto.org/man/mosq uitto_ pub-1 .html

- P ublish a temperature value of 3 2 on the topic `sensors/tem peratu re` to 1 9 2 .1 6 8 .1 .1 (default to `localhost`) with Q uality of S ervice (Q oS) set to 1 :

  ```
  m osq u itto_pu b  - h {{1 9 2 . 1 6 8 . 1 . 1 }} - t {{sensors/tem peratu re
  m  {{3 2 }} - q  {{1 }}
  ```

- P ublish timestamp and temperature data on the topic `sensors/tem peratu re` to a remote host on a non-standard port:

  ```
  m osq u itto_pu b  - h {{1 9 2 . 1 6 8 . 1 . 1 }} - p {{1 8 8 5 }} - t {{senso
  tem peratu re}} - m  " {{1 2 6 6 1 9 3 8 0 4  3 2 }}"
  ```

- P ublish light switch status and retain the message on the topic `sw itches/k itchen_lig hts/statu s` to a remote host because there may be a long period of time between light switch events:

  ```
  m osq u itto_pu b  - r - h " {{iot. eclipse. org }}"  - t {{sw itches/
  k itchen_lig hts/statu s}} - m  " {{on}}"
  ```

- S end the contents of a file (`data. txt`) as a message and publish it to `sensors/tem peratu re` topic:

  ```
  m osq u itto_pu b  - t {{sensors/tem peratu re}} - f {{data. txt}}
  ```

- S end the contents of a file (`data. txt`), by reading from stdin and send the entire input as a message and publish it to `sensors/tem peratu re` topic:

  ```
  m osq u itto_pu b  - t {{sensors/tem peratu re}} - s <  {{data. txt}}
  ```

- R ead newline delimited data from stdin as a message and publish it to `sensors/tem peratu re` topic:

  ```
  {{echo data. txt}} |  m osq u itto_pu b  - t {{sensors/tem peratu re}}
  - l
  ```

mosq uitto_ sub

A simple MQ T T version 3 .1 .1 client that will subscribe to topics and print the messages that it receives.

More information: https://mosq uitto.org/man/mosq uitto_ sub-1 .html

- S ubscribe to the topic `sensors/tem peratu re` information with Q uality of S ervice (`Q oS`) set to 1 . (T he default hostname is `localhost` and port 1 8 8 3):

  ```
  m osq u itto_su b  - t {{sensors/tem peratu re}} - q  {{1 }}
  ```

- S ubscribe to all broker status messages publishing on `iot. eclipse. org` port 1 8 8 5 and print published messages verbosely:

  ```
  m osq u itto_su b  - v  - h " iot. eclipse. org "  - p 1 8 8 5  - t {{\ $ S Y
  ```

- S ubscribe to multiple topics matching a given pattern. (+ takes any metric name):

  ```
  m osq u itto_su b  - t {{sensors/m achines/+ /tem peratu re/+ }}
  ```

most

Open one or several files for interactive reading, allowing scrolling and search.

- Open a file:

  ```
  most {{path/to/file}}
  ```

- Open several files:

  ```
  most {{path/to/file1}} {{path/to/file2}}
  ```

- Open a file at the first occurrence of "string":

  ```
  most {{file}} +/{{string}}
  ```

- Move through opened files:

  ```
  :On
  ```

- Jump to the 100th line:

  ```
  {{100}}j
  ```

- Edit current file:

  ```
  e
  ```

- Split the current window in half:

  ```
  <CTRL-x> o
  ```

- Exit:

  ```
  Q
  ```

mount

Provides access to an entire filesystem in one directory.

- Show all mounted filesystems:

  ```
  mount
  ```

- Mount a device to a directory:

  ```
  mount -t {{filesystem_type}} {{path/to/device_file}} {{path/to/target_directory}}
  ```

- Mount a CD-ROM device (with the filetype ISO 9660) to `/cdrom` (readonly):

  ```
  mount -t {{iso9660}} -o ro {{/dev/cdrom}} {{/cdrom}}
  ```

- Mount all the filesystem defined in `/etc/fstab`:

  ```
  mount -a
  ```

- Mount a specific filesystem described in `/etc/fstab` (e.g. `/dev/sda1 /my_drive ext2 defaults 0 2`):

  ```
  mount {{/my_drive}}
  ```

- Mount a directory to another directory:

  ```
  mount --bind {{path/to/old_dir}} {{path/to/new_dir}}
  ```

mp4box

MPEG-4 Systems Toolbox - Muxes streams into MP4 container.

More information: https://gpac.wp.imt.fr/mp4box

- Display information about an existing MP4 file:

```
mp4box -info {{filename}}
```

- Add an SRT subtitle file into an MP4 file:

```
mp4box -add {{input_subs.srt}}:lang=eng -add {{input.mp4}} {{output.mp4}}
```

- Combine audio from one file and video from another:

```
mp4box -add {{input1.mp4}}#audio -add {{input2.mp4}}#video {{output.mp4}}
```

mpc

Music P layer Client.

P rogram for controlling the Music P layer Daemon (MP D).

More information: https://www.musicpd.org/clients/mpc.

- T oggle play/pause:

  ```
  m pc tog g le
  ```

- S top playing:

  ```
  m pc stop
  ```

- S how information about the currently playing song:

  ```
  m pc statu s
  ```

- P lay next song:

  ```
  m pc next
  ```

- P lay previous song:

  ```
  m pc prev
  ```

- F orward or rewind the currently playing song:

  ```
  m pc [ + - ] {{seconds}}
  ```

mpg3 2 1

H igh P erformance MP E G 1 .0 /2 .0 /2 .5 Audio P layer for Layer 1 , 2 , and 3 .

Mpg3 2 1 was written (sometime in 1 9 9 9) to be a drop-in replacement for the (previously) non-free mpg1 2 3 player.

More information: http://mpg3 2 1 .sourceforge.net/

- P lay an audio source exactly N times (N = 0 means forever):

```
m pg 3 2 1 - l {{N }} {{path/to/file_a| U R L }} {{path/to/file_b | U R L
{{. . . }}
```

- P lay a directory recursively:

```
m pg 3 2 1 - B {{path/to/directory}}
```

- E nable B asic Keys (* or / - Increase or decrease volume, n - S kip song, m - Mute/ unmute.) while playing:

```
m pg 3 2 1 - K {{path/to/file_a| U R L }} {{path/to/file_b | U R L }}
{{. . . }}
```

- P lay files randomly until interrupted:

```
m pg 3 2 1 - Z {{path/to/file_a| U R L }} {{path/to/file_b | U R L }}
{{. . . }}
```

- S huffl e the files before playing them once:

```
m pg 3 2 1 - z {{path/to/file_a| U R L }} {{path/to/file_b | U R L }}
{{. . . }}
```

- P lay all files in the current directory and subdirectories, randomly (until interrupted), with B asic Keys enabled:

```
m pg 3 2 1 - B - Z - K .
```

mpv

A audio/video player based on MPlayer.

More information: https://mpv.io.

- Play a video or audio file:

```
mpv {{file}}
```

- Play a video or audio file from an URL:

```
mpv '{{https://www.youtube.com/watch?v=dQw4w9WgXcQ}}'
```

- Jump backward/forward 5 seconds:

```
LEFT <or> RIGHT
```

- Jump backward/forward 1 minute:

```
DOWN <or> UP
```

- Decrease or increase playback speed by 10%:

```
[ <or> ]
```

- Play a file at a specified speed (0.01 to 100, default 1):

```
mpv --speed {{speed}} {{file}}
```

- Play a file using a profile defined in the `mpv.conf` file:

```
mpv --profile {{profile_name}} {{file}}
```

- Display the output of webcam or other video input device:

```
mpv /dev/{{video0}}
```

mr

Manage all of your version control repositories at once.

More information: https://myrepos.branchable.com.

- Register a repository:

  ```
  mr register
  ```

- Update repositories in 5 concurrent jobs:

  ```
  mr -j {{5}} update
  ```

- Print the status of all repositories:

  ```
  mr status
  ```

- Checkout all repositories to the latest version:

  ```
  mr checkout
  ```

msbuild

The Microsoft build tool for Visual Studio project solutions.

More information: https://docs.microsoft.com/visualstudio/msbuild.

- Build the first project file in the current directory:

  ```
  msbuild
  ```

- Build a specific project file:

  ```
  msbuild {{path/to/project_file}}
  ```

- Set one or more semicolon-separated targets to build:

  ```
  msbuild {{path/to/project_file}} /target:{{targets}}
  ```

- Set one or more semicolon-separated properties:

  ```
  msbuild {{path/to/project_file}} /property:{{name=value}}
  ```

- Set the build tools version to use:

  ```
  msbuild {{path/to/project_file}} /toolsversion:{{version}}
  ```

- Display detailed information at the end of the log about how the project was configured:

  ```
  msbuild {{path/to/project_file}} /detailedsummary
  ```

- Display detailed help information:

  ```
  msbuild /help
  ```

mscore

This command is an alias of **`musescore`**.

- View documentation for the original command:

  `tldr musescore`

msfvenom

Manually generate payloads for metasploit.

More information: https://github.com/rapid7/metasploit-framework/wiki/How-to-use-msfvenom.

- List payloads:

```
msfvenom -l payloads
```

- List formats:

```
msfvenom -l formats
```

- Show payload options:

```
msfvenom -p {{payload}} --list-options
```

- Create an ELF binary with a reverse TCP handler:

```
msfvenom -p linux/x64/meterpreter/reverse_tcp
LHOST={{local_ip}} LPORT={{local_port}} -f elf > {{path/to/
binary}}
```

- Create an EXE binary with a reverse TCP handler:

```
msfvenom -p windows/x64/meterpreter/reverse_tcp
LHOST={{local_ip}} LPORT={{local_port}} -f exe > {{path/to/
binary.exe}}
```

msmtp

An SMTP client.

It reads text from standard input and sends it to an SMTP server.

More information: https://marlam.de/msmtp.

- Send an email using the default account configured in `~/.msmtprc`

  ```
  echo "{{Hello world}}" | msmtp {{to@example.org}}
  ```

- Send an email using a specific account configured in `~/.msmtprc`

  ```
  echo "{{Hello world}}" | msmtp --account={{account_name}} {{to@example.org}}
  ```

- Send an email without a configured account. The password should be specified in the `~/.msmtprc` file:

  ```
  echo "{{Hello world}}" | msmtp --host={{localhost}} --port={{999}} --from={{from@example.org}} {{to@example.org}}
  ```

mtr

Matt' s T raceroute: combined traceroute and ping tool.

More information: https://bitwiz ard.nl/mtr

- T raceroute to a host and continuously ping all intermediary hops:

  ```
  m tr {{host}}
  ```

- Disable IP address and host name mapping:

  ```
  m tr - n {{host}}
  ```

- G enerate output after pinging each hop 1 0 times:

  ```
  m tr - w {{host}}
  ```

- F orce IP IP v4 or IP V 6 :

  ```
  m tr - 4 {{host}}
  ```

- Wait for a given time (in seconds) before sending another packet to the same hop:

  ```
  m tr - i {{seconds}} {{host}}
  ```

multipass

CLI to manage U buntu virtual machines using native hypervisors.

More information: https://multipass.run/.

- List the aliases that can be used to launch an instance:

  ```
  m u ltipass find
  ```

- Launch a new instance, set its name and use a cloud-init configuration file:

  ```
  m u ltipass lau nch - n {{instance_nam e}} - - clou d- init
  {{config u ration_file}}
  ```

- List all the created instances and some of their properties:

  ```
  m u ltipass list
  ```

- S tart a specific instance by name:

  ```
  m u ltipass start {{instance_nam e}}
  ```

- S how the properties of an instance:

  ```
  m u ltipass info {{instance_nam e}}
  ```

- O pen a shell prompt on a specific instance by name:

  ```
  m u ltipass shell {{instance_nam e}}
  ```

- Delete an instance by name:

  ```
  m u ltipass delete {{instance_nam e}}
  ```

- Mount a directory into a specific instance:

  ```
  m u ltipass m ou nt {{path/to/local/directory}}
  {{instance_nam e}}: {{path/to/targ et/directory}}
  ```

multitail

Extension of tail.

More information: https://manned.org/multitail.

- Tail all files matching a pattern in a single stream:

```
multitail -Q 1 '{{pattern}}'
```

- Tail all files in a directory in a single stream:

```
multitail -Q 1 '{{directory}}/*'
```

- Automatically add new files to a window:

```
multitail -Q {{pattern}}
```

- Show 5 logfiles while merging 2 and put them in 2 columns with only one in the left column:

```
multitail -s 2 -sn 1,3 {{mergefile}} -I {{file1}} {{file2}} {{file3}} {{file4}}
```

mupdf

MuPDF is a lightweight PDF, XPS, and E-book viewer.

More information: https://www.mupdf.com.

- Open a PDF on the first page:

```
mupdf {{filename}}
```

- Open a PDF on page 3:

```
mupdf {{filename}} {{3}}
```

- Open a password secured PDF:

```
mupdf -p {{password}} {{filename}}
```

- Open a PDF with an initial zoom level, specified as DPI, of 72:

```
mupdf -r {{72}} {{filename}}
```

- Open a PDF with inverted color:

```
mupdf -I {{filename}}
```

- Open a PDF tinted red #FF0000 (hexadecimal color syntax RRGGBB):

```
mupdf -C {{FF0000}}
```

- Open a PDF without anti-aliasing (0 = off, 8 = best):

```
mupdf -A {{0}}
```

musescore

MuseScore 3 sheet music editor.

More information: https://musescore.org/en/handbook/3/command-line-options

- Use a specific audio driver:

```
musescore --audio-driver {{jack|alsa|portaudio|pulse}}
```

- Set the MP3 output bitrate in kbit/s:

```
musescore --bitrate {{bitrate}}
```

- Start MuseScore in debug mode:

```
musescore --debug
```

- Enable experimental features, such as layers:

```
musescore --experimental
```

- Export the given file to the specified output file. The file type depends on the given extension:

```
musescore --export-to {{output_file}} {{input_file}}
```

- Print a diff between the given scores:

```
musescore --diff {{path/to/file1}} {{path/to/file2}}
```

- Specify a MIDI import operations file:

```
musescore --midi-operations {{path/to/file}}
```

mutagen

Real-time file synchronization and network forwarding tool.

More information: https://mutagen.io.

- Start a synchronization session between a local directory and a remote host:

  ```
  mutagen sync create --name={{session_name}} {{/path/to/local/directory/}} {{user}}@{{host}}:{{/path/to/remote/directory/}}
  ```

- Start a synchronization session between a local directory and a Docker container:

  ```
  mutagen sync create --name={{session_name}} {{/path/to/local/directory/}} docker://{{user}}@{{container_name}}{{/path/to/remote/directory/}}
  ```

- Stop a running session:

  ```
  mutagen sync terminate {{session_name}}
  ```

- Start a project:

  ```
  mutagen project start
  ```

- Stop a project:

  ```
  mutagen project terminate
  ```

- List running sessions for the current project:

  ```
  mutagen project list
  ```

mutool

Convert PDF files, query information and extract data.

More information: https://mupdf.com/docs.

- Convert pages 1-10 into 10 PNG images:

```
mutool convert -o {{image%d.png}} {{file.pdf}} {{1-10}}
```

- Convert pages 2, 3 and 5 of a PDF into text in the standard output:

```
mutool draw -F {{txt}} {{file.pdf}} {{2,3,5}}
```

- Concatenate two PDFs:

```
mutool merge -o {{output.pdf}} {{input1.pdf}} {{input2.pdf}}
```

- Query information about all content embedded in a PDF:

```
mutool info {{input.pdf}}
```

- Extract all images, fonts and resources embedded in a PDF out into the current directory:

```
mutool extract {{input.pdf}}
```

- Print the outline (table of contents) of a PDF:

```
mutool show {{input.pdf}} outline
```

mutt

Command-line email client.

More information: http://mutt.org.

- Open the specified mailbox:

```
mutt -f {{mailbox}}
```

- Send an email and specify a subject and a cc recipient:

```
mutt -s {{subject}} -c {{cc@example.com}}
{{recipient@example.com}}
```

- Send an email with files attached:

```
mutt -a {{file1}} {{file2}} -- {{recipient@example.com}}
```

- Specify a file to include as the message body:

```
mutt -i {{file}} {{recipient@example.com}}
```

- Specify a draft file containing the header and the body of the message, in RFC5322 format:

```
mutt -H {{file}} {{recipient@example.com}}
```

mv

Move or rename files and directories.

More information: https://www.gnu.org/software/coreutils/mv.

- Move a file to an arbitrary location:

```
mv {{source}} {{target}}
```

- Move files into another directory, keeping the filenames:

```
mv {{source1}} {{source2}} {{source3}} {{target_directory}}
```

- Do not prompt for confirmation before overwriting existing files:

```
mv -f {{source}} {{target}}
```

- Prompt for confirmation before overwriting existing files, regardless of file permissions:

```
mv -i {{source}} {{target}}
```

- Do not overwrite existing files at the target:

```
mv -n {{source}} {{target}}
```

- Move files in verbose mode, showing files after they are moved:

```
mv -v {{source}} {{target}}
```

mvn

Apache Maven.

Tool for building and managing Java-based projects.

More information: https://maven.apache.org.

- Compile a project:

  ```
  mvn compile
  ```

- Compile and package the compiled code in its distributable format, such as a `jar`:

  ```
  mvn package
  ```

- Compile and package, skipping unit tests:

  ```
  mvn package -Dmaven.test.skip=true
  ```

- Install the built package in local maven repository. (This will invoke the compile and package commands too):

  ```
  mvn install
  ```

- Delete build artifacts from the target directory:

  ```
  mvn clean
  ```

- Do a clean and then invoke the package phase:

  ```
  mvn clean package
  ```

- Clean and then package the code with a given build profile:

  ```
  mvn clean -P {{profile}} package
  ```

- Run a class with a main method:

  ```
  mvn exec:java -Dexec.mainClass="{{com.example.Main}}" -
  Dexec.args="{{arg1 arg2}}"
  ```

mycli

A command-line client for MyS Q L that can do auto-completion and syntax highlighting.

More information: https://mycli.net.

- Connect to a local database on port 3 3 0 6 , using the current user' s username:

  ```
  m ycli {{datab ase_nam e}}
  ```

- Connect to a database (user will be prompted for a password):

  ```
  m ycli - u {{u sernam e}} {{datab ase_nam e}}
  ```

- Connect to a database on another host:

  ```
  m ycli - h {{datab ase_host}} - P {{port}} - u {{u sernam e}}
  {{datab ase_nam e}}
  ```

mysq l

T he MyS Q L command-line tool.

More information: https://www.mysq l.com/

- Connect to a database:

```
m ysq l {{datab ase_nam e}}
```

- Connect to a database, user will be prompted for a password:

```
m ysq l - u {{u ser}} - - passw ord {{datab ase_nam e}}
```

- Connect to a database on another host:

```
m ysq l - h {{datab ase_host}} {{datab ase_nam e}}
```

- Connect to a database through a U nix socket:

```
m ysq l - - sock et {{path/to/sock et. sock }}
```

- E xecute S Q L statements in a script file (batch file):

```
m ysq l - e " sou rce {{filenam e. sq l}}" {{datab ase_nam e}}
```

- R estore a database from a backup created with `m ysq ldu mp` (user will be prompted for a password):

```
m ysq l - - u ser {{u ser}} - - passw ord {{datab ase_nam e}} < {{path/
to/b ack u p. sq l}}
```

- R estore all databases from a backup (user will be prompted for a password):

```
m ysq l - - u ser {{u ser}} - - passw ord < {{path/to/b ack u p. sq l}}
```

mysqld

Start the MySQL database server.

More information: https://dev.mysql.com/doc/refman/en/mysqld.html

- Start the MySQL database server:

  ```
  mysqld
  ```

- Start the server, printing error messages to the console:

  ```
  mysqld --console
  ```

- Start the server, saving logging output to a custom log file:

  ```
  mysqld --log={{path/to/file.log}}
  ```

- Print the default arguments and their values and exit:

  ```
  mysqld --print-defaults
  ```

- Start the server, reading arguments and values from a file:

  ```
  mysqld --defaults-file={{path/to/file}}
  ```

- Start the server and listen on a custom port:

  ```
  mysqld --port={{port}}
  ```

- Show all help options and exit:

  ```
  mysqld --verbose --help
  ```

mysqldump

Backups MySQL databases.

See also `mysql` for restoring databases.

More information: https://dev.mysql.com/doc/refman/en/mysqldump.html

- Create a backup (user will be prompted for a password):

```
mysqldump -u {{user}} -p {{database_name}} --result-file={{path/to/file.sql}}
```

- Backup a specific table redirecting the output to a file (user will be prompted for a password):

```
mysqldump -u {{user}} -p {{database_name}} {{table_name}} > {{path/to/file.sql}}
```

- Backup all databases redirecting the output to a file (user will be prompted for a password):

```
mysqldump -u {{user}} -p --all-databases > {{path/to/file.sql}}
```

- Backup all databases from a remote host, redirecting the output to a file (user will be prompted for a password):

```
mysqldump --host={{ip_or_hostname}} -u {{user}} -p --all-databases > {{path/to/file.sql}}
```

mytop

Display MySQL server performance info like `top`

More information: http://jeremy.zawodny.com/mysql/mytop/mytop.html

- Start mytop:

```
mytop
```

- Connect with a specified username and password:

```
mytop -u {{user}} -p {{password}}
```

- Connect with a specified username (the user will be prompted for a password):

```
mytop -u {{user}} --prompt
```

- Do not show any idle (sleeping) threads:

```
mytop -u {{user}} -p {{password}} --noidle
```

nano

Simple, easy to use command-line text editor. An enhanced, free Pico clone.

More information: https://nano-editor.org.

- Open a new file in nano:

  ```
  nano
  ```

- Open a specific file:

  ```
  nano {{path/to/file}}
  ```

- Open a specific file, positioning the cursor at the specified line and column:

  ```
  nano +{{line}},{{column}} {{path/to/file}}
  ```

- Open a specific file and enable soft wrapping:

  ```
  nano --softwrap {{path/to/file}}
  ```

- Open a specific file and indent new lines to the previous lines' indentation:

  ```
  nano --autoindent {{path/to/file}}
  ```

- Open nano and create a backup file (`file~`) when saving edits:

  ```
  nano --backup {{path/to/file}}
  ```

nasm

The Netwide Assembler, a portable 80x86 assembler.

More information: https://nasm.us.

- Assemble `source.asm` into a binary file `source`, in the (default) raw binary format:

  `nasm {{source.asm}}`

- Assemble `source.asm` into a binary file `output_file`, in the specified format:

  `nasm -f {{format}} {{source.asm}} -o {{output_file}}`

- List valid output formats (along with basic nasm help):

  `nasm -hf`

- Assemble and generate an assembly listing file:

  `nasm -l {{list_file}} {{source.asm}}`

- Add a directory (must be written with trailing slash) to the include file search path before assembling:

  `nasm -i {{path/to/include_dir/}} {{source.asm}}`

nativefier

Command-line tool to create a desktop app for any web site with minimal configuration.

More information: https://github.com/j iahaog/nativefier

- Make a desktop app for a website:

  ```
  nativ efier {{u rl}}
  ```

- Create a desktop app with a custom name:

  ```
  nativ efier - - nam e {{nam e}} {{u rl}}
  ```

- U se a custom icon, should be a P N G :

  ```
  nativ efier - - icon {{path/to/icon. png }} {{u rl}}
  ```

nbtscan

Scans networks for NetBIOS name information.

More information: https://github.com/resurrecting-open-source-projects/nbtscan

- Scan a network for NetBIOS names:

  `nbtscan {{192.168.0.1/24}}`

- Scan a single IP address:

  `nbtscan {{192.168.0.1}}`

- Display verbose output:

  `nbtscan -v {{192.168.0.1/24}}`

- Display output in `/etc/hosts` format:

  `nbtscan -e {{192.168.0.1/24}}`

- Read IP addresses / networks to scan from a file:

  `nbtscan -f {{path/to/file.txt}}`

nc

Netcat is a versatile utility for working with TCP or UDP data.

More information: https://nmap.org/ncat.

- Listen on a specified port and print any data received:

```
nc -l {{port}}
```

- Connect to a certain port:

```
nc {{ip_address}} {{port}}
```

- Set a timeout:

```
nc -w {{timeout_in_seconds}} {{ipaddress}} {{port}}
```

- Keep the server up after the client detaches:

```
nc -k -l {{port}}
```

- Keep the client up even after EOF:

```
nc -q {{timeout}} {{ip_address}}
```

- Scan the open ports of a specified host:

```
nc -v -z {{ip_address}} {{port}}
```

- Act as proxy and forward data from a local TCP port to the given remote host:

```
nc -l {{local_port}} | nc {{hostname}} {{remote_port}}
```

ncc

Compile a Node.js application into a single file.

Supports TypeScript, binary addons and dynamic requires.

More information: https://github.com/vercel/ncc.

- Bundle a Node.js application:

  ncc build {{path/to/file.js}}

- Bundle and minify a Node.js application:

  ncc build --minify {{path/to/file.js}}

- Bundle and minify a Node.js application and generate source maps:

  ncc build --source-map {{path/to/file.js}}

- Automatically recompile on changes to source files:

  ncc build --watch {{path/to/file.js}}

- Bundle a Node.js application into a temporary directory and run it for testing:

  ncc run {{path/to/file.js}}

- Clean the `ncc` cache:

  ncc clean cache

ncmpcpp

A command-line music player client for the Music Player Daemon.

More information: https://rybczak.net/ncmpcpp

- Connect to a music player daemon on a given host and port:

  ```
  ncmpcpp --host {{ip}} --port {{port}}
  ```

- Display metadata of the current song to console:

  ```
  ncmpcpp --current-song
  ```

- Use a specified configuration file:

  ```
  ncmpcpp --config {{file}}
  ```

- Use a different set of key bindings from a file:

  ```
  ncmpcpp --bindings {{file}}
  ```

neato

Render an image of a **linear undirected** network graph from a **graphviz** file.

Layouts: **dot**, **neato**, **twopi**, **circo**, **fdp**, **sfdp**, **osage** & **patchwork**

More information: https://graphviz.org/doc/info/command.html

- Render a `png` image with a filename based on the input filename and output format (uppercase -O):

  ```
  neato -T {{png}} -O {{path/to/input.gv}}
  ```

- Render a `svg` image with the specified output filename (lowercase -o):

  ```
  neato -T {{svg}} -o {{path/to/image.svg}} {{path/to/input.gv}}
  ```

- Render the output in `ps`, `pdf`, `svgz`, `fig`, `png`, `gif`, `jpg`, `jpeg`, or `dot` format:

  ```
  neato -T {{format}} -O {{path/to/input.gv}}
  ```

- Render a `gif` image using stdin and stdout:

  ```
  echo "{{graph {this--that}}}" | neato -T {{gif}} > {{path/to/image.gif}}
  ```

- Display help:

  ```
  neato -?
  ```

ned

Is like `grep` but with powerful replace capabilities.

Unlike `sed`, as it isn't restricted to line oriented editing.

More information: https://github.com/nevdelap/ned.

- Recursively search starting in the current directory, ignoring case:

  ```
  ned --ignore-case --recursive '{{^[dl]og}}' {{.}}
  ```

- Search always showing colored output:

  ```
  ned --colors '{{^[dl]og}}' {{.}}
  ```

- Search never showing colored output:

  ```
  ned --colors=never '{{^[dl]og}}' {{.}}
  ```

- Search ignoring certain files:

  ```
  ned --recursive --exclude '{{*.htm}}' '{{^[dl]og}}' {{.}}
  ```

- Simple replace:

  ```
  ned '{{dog}}' --replace '{{cat}}' {{.}}
  ```

- Replace using numbered group references:

  ```
  ned '{{the ([a-z]+) dog and the ([a-z]+) dog}}' --replace
  '{{the $2 dog and the $1 dog}}' {{.}}
  ```

- Replace changing case:

  ```
  ned '{{(([a-z]+) dog}}' --case-replacements --replace
  '{{\U$1\E! dog}}' --stdout {{.}}
  ```

- Preview results of a find and replace without updating the target files:

  ```
  ned '{{^[sb]ad}}' --replace '{{happy}}' --stdout {{.}}
  ```

neofetch

CLI tool to display information about your operating system, software and hardware.

More information: https://github.com/dylanaraps/neofetch.

- Return the default config, and create it if it's the first time the program runs:

  ```
  neofetch
  ```

- Trigger an info line from appearing in the output, where 'infoname' is the function name in the config file, e.g. memory:

  ```
  neofetch --{{enable|disable}} {{infoname}}
  ```

- Hide/Show OS architecture:

  ```
  neofetch --os_arch {{on|off}}
  ```

- Enable/Disable CPU brand in output:

  ```
  neofetch --cpu_brand {{on|off}}
  ```

netlify

Deploy sites and configure continuous deployment to the N etlify platform.

More information: https://www.netlify.com/docs/cli/.

- Log in to the N etlify account:

  ```
  netlify log in
  ```

- Deploy the contents of a directory to N etlify:

  ```
  netlify deploy
  ```

- Configure continuous deployment for a new or an existing site:

  ```
  netlify init
  ```

- S tart a local dev server:

  ```
  netlify dev
  ```

newman

Collection runner for P ostman.

More information: https://github.com/postmanlabs/newman.

- R un a collection (from a file):

  ```
  new m an  ru n {{path/to/collection. j son}}
  ```

- R un a collection (from a U R L):

  ```
  new m an  ru n {{https: //w w w . g etpostm an. com /collections/6 3 1 6 4 3
  f6 9 5 cab 7 - 6 8 7 8 - eb 5 5 - 7 9 4 3 - ad8 8 e1 ccfd6 5 - J sL v }}
  ```

newsboat

An R S S /Atom feed reader for text terminals.

More information: https://newsboat.org/.

- F irst import feed U R Ls from an O P ML file:

  ```
  new sb oat - i {{m y- feeds. xm l}}
  ```

- Alternatively, add feeds manually:

  ```
  echo {{http: //exam ple. com /path/to/feed}} > >  " $
  {H O M E }/. new sb oat/u rls"
  ```

- S tart newsboat and refresh all feeds on startup:

  ```
  new sb oat - r
  ```

- S ee keyboard shortcuts (the most relevant are visible in the status line):

  ```
  ?
  ```

next

R eact framework that uses server-side rendering for building optimiz ed web applications.

More information: https://nextj s.org/docs

- S tart the current application in development mode:

  ```
  next dev
  ```

- S tart the current application and listen on a specific port:

  ```
  next dev - - port {{port}}
  ```

- B uild the current application optimiz ed for production:

  ```
  next b u ild
  ```

- S tart the compiled application in production mode:

  ```
  next start
  ```

- S tart the compiled application and listen on a specific port:

  ```
  next start - - port {{port}}
  ```

- E xport the current application to static H T ML pages:

  ```
  next export
  ```

- Display the N ext.j s telemetry status:

  ```
  next telem etry
  ```

- Display help for a subcommand:

  ```
  next {{b u ild| dev | export| start| telem etry}} - - help
  ```

nextflow

Tool for running computational pipelines. Mostly used for bioinformatics workflows.

More information: https://www.nextflow.io

- Run a pipeline, use cached results from previous runs:

  ```
  nextflow run {{main.nf}} -resume
  ```

- Run a specific release of a remote workflow from GitHub:

  ```
  nextflow run {{user/repo}} -revision {{release_tag}}
  ```

- Run with a given work directory for intermediate files, save execution report:

  ```
  nextflow run {{workflow}} -work-dir {{path/to/directory}} -with-report {{report.html}}
  ```

- Show details of previous runs in current directory:

  ```
  nextflow log
  ```

- Remove cache and intermediate files for a specific run:

  ```
  nextflow clean -force {{run_name}}
  ```

- List all downloaded projects:

  ```
  nextflow list
  ```

- Pull the latest version of a remote workflow from Bitbucket:

  ```
  nextflow pull {{user/repo}} -hub bitbucket
  ```

- Update Nextflow:

  ```
  nextflow self-update
  ```

nf-core

The nf-core framework tools, to create, check and develop best-practice guidelines for Nextflow.

More information: https://nf-co.re/tools.

- List existing pipelines on nf-core:

  ```
  nf-core list
  ```

- Create a new pipeline skeleton:

  ```
  nf-core create
  ```

- Lint the pipeline code:

  ```
  nf-core lint {{path/to/directory}}
  ```

- Bump software versions in pipeline recipe:

  ```
  nf-core bump-version {{path/to/directory}} {{new_version}}
  ```

- Launch an nf-core pipeline:

  ```
  nf-core launch {{pipeline_name}}
  ```

- Download an nf-core pipeline for offline use:

  ```
  nf-core download {{pipeline_name}}
  ```

ng

Command-line Interface (CLI) for creating and managing Angular applications.

More information: https://angular.io/cli.

- Create a new Angular application inside a directory:

  ```
  ng new {{project_name}}
  ```

- Add a new component to one's application:

  ```
  ng generate component {{component_name}}
  ```

- Add a new class to one's application:

  ```
  ng generate class {{class_name}}
  ```

- Add a new directive to one's application:

  ```
  ng generate directive {{directive_name}}
  ```

- Run the application with the following command in its root directory:

  ```
  ng serve
  ```

- Build the application:

  ```
  ng build
  ```

- Run unit tests:

  ```
  ng test
  ```

- Check the version of your current Angular installation:

  ```
  ng version
  ```

nginx

N ginx web server.

More information: https://nginx.org/en/.

- S tart server with the default config file:

  ```
  ng inx
  ```

- S tart server with a custom config file:

  ```
  ng inx - c {{config _file}}
  ```

- S tart server with a prefix for all relative paths in the config file:

  ```
  ng inx - c {{config _file}} - p {{prefix/for/relativ e/paths}}
  ```

- T est the configuration without affecting the running server:

  ```
  ng inx - t
  ```

- R eload the configuration by sending a signal with no downtime:

  ```
  ng inx - s reload
  ```

ngrep

Filter network traffic packets using regular expressions.

More information: https://github.com/jpr5/ngrep.

- Capture traffic of all interfaces:

```
ngrep -d any
```

- Capture traffic of a specific interface:

```
ngrep -d {{eth0}}
```

- Capture traffic crossing port 22 of interface eth0:

```
ngrep -d {{eth0}} port {{22}}
```

- Capture traffic from or to a host:

```
ngrep host {{www.example.com}}
```

- Filter keyword 'User-Agent:' of interface eth0:

```
ngrep -d {{eth0}} '{{User-Agent:}}'
```

ngrok

Reverse proxy that creates a secure tunnel from a public endpoint to a locally running web service.

More information: https://ngrok.com.

- Expose a local HTTP service on a given port:

  ```
  ngrok http {{80}}
  ```

- Expose a local HTTP service on a specific host:

  ```
  ngrok http {{foo.dev}}:{{80}}
  ```

- Expose a local HTTPS server:

  ```
  ngrok http https://localhost
  ```

- Expose TCP traffic on a given port:

  ```
  ngrok tcp {{22}}
  ```

- Expose TLS traffic for a specific host and port:

  ```
  ngrok tls -hostname={{foo.com}} {{443}}
  ```

nice

Execute a program with a custom scheduling priority (niceness).

Niceness values range from -20 (the highest priority) to 19 (the lowest).

More information: https://www.gnu.org/software/coreutils/nice.

- Launch a program with altered priority:

```
nice -n {{niceness_value}} {{command}}
```

nikto

Web server scanner which performs tests against web servers for multiple items.

More information: https://cirt.net/N ikto2

- P erform a basic N ikto scan against a target host:

```
perl nik to. pl - h {{1 9 2 . 1 6 8 . 0 . 1 }}
```

- S pecify the port number when performing a basic scan:

```
perl nik to. pl - h {{1 9 2 . 1 6 8 . 0 . 1 }} - p {{4 4 3 }}
```

- S can ports and protocols with full U R L syntax:

```
perl nik to. pl - h {{https: //1 9 2 . 1 6 8 . 0 . 1 : 4 4 3 /}}
```

- S can multiple ports in the same scanning session:

```
perl nik to. pl - h {{1 9 2 . 1 6 8 . 0 . 1 }} - p {{8 0 , 8 8 , 4 4 3 }}
```

- U pdate to the latest plugins and databases:

```
perl nik to. pl - u pdate
```

nim

The Nim compiler.

Processes, compiles and links Nim language source files.

More information: https://nim-lang.org.

- Compile a source file:

```
nim compile {{file.nim}}
```

- Compile and run a source file:

```
nim compile -r {{file.nim}}
```

- Compile a source file with release optimizations enabled:

```
nim compile -d:release {{file.nim}}
```

- Build a release binary optimized for low file size:

```
nim compile -d:release --opt:size {{file.nim}}
```

- Generate HTML documentation for a module (output will be placed in the current directory):

```
nim doc {{file.nim}}
```

nimble

Package manager for the Nim programming language.

Manage Nim projects and their dependencies.

More information: https://github.com/nim-lang/nimble.

- Search for packages:

  ```
  nimble search {{search_string}}
  ```

- Install a package:

  ```
  nimble install {{package_name}}
  ```

- List installed packages:

  ```
  nimble list -i
  ```

- Create a new Nimble package in the current directory:

  ```
  nimble init
  ```

- Build a Nimble package:

  ```
  nimble build
  ```

- Install a Nimble package:

  ```
  nimble install
  ```

ninja

A build system designed to be fast.

More information: https://ninja-build.org/manual.html

- Build in the current directory:

  `ninja`

- Build a program in a given directory:

  `ninja -C {{path/to/directory}}`

- Show targets (e.g. `install` and `uninstall`):

  `ninja -t targets`

- Show help:

  `ninja -h`

nix-build

Build a Nix expression.

More information: https://nixos.org/releases/nix/latest/manual#_sec-nix-build

- Build a Nix expression:

```
nix-build --attr {{expression_name}}
```

- Build a sandboxed Nix expression (on non-NixOS):

```
nix-build --attr {{expression_name}} --option sandbox true
```

nix-collect-garbage

Delete unused and unreachable nix store paths.

Generations can be listed using `nix-env --list-generations`.

More information: https://nixos.org/releases/nix/latest/manual/#sec-nix-collect-garbage.

- Delete all store paths unused by current generations of each profile:

```
sudo nix-collect-garbage --delete-old
```

- Simulate the deletion of old store paths:

```
sudo nix-collect-garbage --delete-old --dry-run
```

- Delete all store paths older than 30 days:

```
sudo nix-collect-garbage --delete-older-than {{30d}}
```

nix-env

Manipulate or q uery N ix user environments.

More information: https://nixos.org/manual/nix/stable/#_sec-nix-env

- List all installed packages:

```
nix- env  - q
```

- Q uery installed packages:

```
nix- env  - q  {{search_term }}
```

- Q uery available packages:

```
nix- env  - q a {{search_term }}
```

- Install package:

```
nix- env  - iA  nixpk g s. {{pk g _nam e}}
```

- Install a package from a U R L:

```
nix- env  - i {{pk g _nam e}} - - file {{exam ple. com }}
```

- U ninstall package:

```
nix- env  - e {{pk g _nam e}}
```

- U pgrade one package:

```
nix- env  - u  {{pk g _nam e}}
```

- U pgrade all packages:

```
nix- env  - u
```

nix-shell

Start an interactive shell based on a Nix expression.

More information: https://nixos.org/manual/nix/stable/#sec-nix-shell

- Start with nix expression in `shell.nix` or `default.nix` in the current directory:

  ```
  nix-shell
  ```

- Run shell command in non-interactive shell and exit:

  ```
  nix-shell --run "{{command}} {{command_arguments}}"
  ```

- Start with expression in `default.nix` in the current directory:

  ```
  nix-shell {{default.nix}}
  ```

- Start with packages loaded from nixpkgs:

  ```
  nix-shell --packages {{package_name_1}} {{package_name_2}}
  ```

- Start with packages loaded from specific nixpkgs revision:

  ```
  nix-shell --packages {{package_names}} -I nixpkgs={{https://github.com/NixOS/nixpkgs/archive/nixpkgs_revision.tar.gz}}
  ```

- Evaluate rest of file in specific interpreter, for use in `#!-scripts` (see https://nixos.org/manual/nix/stable/#use-as-a-interpreter):

  ```
  nix-shell -i {{interpreter}} --packages {{package_names}}
  ```

nix

Utilities for the Nix language and store.

More information: https://nixos.org.

- Search for a package via its name or description:

  ```
  nix search {{search_term}}
  ```

- Start a Nix shell with the specified packages available:

  ```
  nix run {{nixpkgs.pkg1 nixpkgs.pkg2 nixpkgs.pkg3 ...}}
  ```

- Optimise Nix store disk usage by combining duplicate files:

  ```
  nix optimise-store
  ```

- Start an interactive environment for evaluating Nix expressions:

  ```
  nix repl
  ```

- Upgrade Nix to the latest stable version:

  ```
  nix upgrade-nix
  ```

nkf

Network kanji filter.

Converts kanji code from one encoding to another.

- Convert to UTF-8 encoding:

  ```
  nkf -w {{path/to/file.txt}}
  ```

- Convert to SHIFT_JIS encoding:

  ```
  nkf -s {{path/to/file.txt}}
  ```

- Convert to UTF-8 encoding and overwrite the file:

  ```
  nkf -w --overwrite {{path/to/file.txt}}
  ```

- Set new line code to LF and overwrite (unix type):

  ```
  nkf -d --overwrite {{path/to/file.txt}}
  ```

- Set new line code to CRLF and overwrite (windows type):

  ```
  nkf -c --overwrite {{path/to/file.txt}}
  ```

- Decrypt mime file and overwrite:

  ```
  nkf -m --overwrite {{path/to/file.txt}}
  ```

nl

A utility for numbering lines, either from a file, or from standard input.

More information: https://www.gnu.org/software/coreutils/nl.

- Number non-blank lines in a file:

```
nl {{file}}
```

- Read from standard output:

```
cat {{file}} | nl {{options}} -
```

- Number only the lines with printable text:

```
nl -t {{file}}
```

- Number all lines including blank lines:

```
nl -b a {{file}}
```

- Number only the body lines that match a basic regular expression (BRE) pattern:

```
nl -b p'FooBar[0-9]' {{file}}
```

nload

A tool for visualiz ing network usage in the terminal.

More information: https://github.com/rolandriegel/nload.

- V iew all network traffic (use the arrow keys to switch interfaces):

  ```
  nload
  ```

- V iew network traffic on specific interfaces (use the arrow keys to switch interfaces):

  ```
  nload dev ice {{interface_one}} {{interface_tw o}}
  ```

nmap

Network exploration tool and security / port scanner.

Some features only activate when Nmap is run with privileges.

More information: https://nmap.org.

- Check if an IP address is up, and guess the remote host's operating system:

  `nmap -O {{ip_or_hostname}}`

- Try to determine whether the specified hosts are up and what their names are:

  `nmap -sn {{ip_or_hostname}} {{optional_another_address}}`

- Like above, but also run a default 1000-port TCP scan if host seems up:

  `nmap {{ip_or_hostname}} {{optional_another_address}}`

- Also enable scripts, service detection, OS fingerprinting and traceroute:

  `nmap -A {{address_or_addresses}}`

- Assume good network connection and speed up execution:

  `nmap -T4 {{address_or_addresses}}`

- Scan a specific list of ports (use `-p-` for all ports `1-65535`):

  `nmap -p {{port1,port2,…,portN}} {{address_or_addresses}}`

- Perform TCP and UDP scanning (use `-sU` for UDP only, `-sZ` for SCTP, `-sO` for IP):

  `nmap -sSU {{address_or_addresses}}`

- Perform full port, service, version detection scan with all default NSE scripts active against a host to determine weaknesses and info:

  `nmap -sC -sV {{address_or_addresses}}`

nms

Command-line tool that recreates the famous data decryption effect seen in the 1 9 9 2 movie S neakers from stdin.

More information: https://github.com/bartobri/no-more-secrets.

- Decrypt text after a keystroke:

```
echo " {{H ello,  W orld! }}"  |  nm s
```

- Decrypt output immediately, without waiting for a keystroke:

```
{{ls - la}} |  nm s - a
```

- Decrypt the content of a file, with a custom output color:

```
cat {{path/to/file}} |  nm s - a - f {{b lu e| w hite| yellow | b lack |
m ag enta| g reen| red}}
```

- Clear the screen before decrypting:

```
{{com m and}} |  nm s - a - c
```

node

Server-side JavaScript platform (Node.js).

More information: https://nodejs.org

- Run a JavaScript file:

  ```
  node {{path/to/file}}
  ```

- Start a REPL (interactive shell):

  ```
  node
  ```

- Evaluate JavaScript code by passing it as an argument:

  ```
  node -e "{{code}}"
  ```

- Evaluate and print result, useful to see node's dependencies versions:

  ```
  node -p "{{process.versions}}"
  ```

- Activate inspector, pausing execution until a debugger is connected once source code is fully parsed:

  ```
  node --no-lazy --inspect-brk {{path/to/file}}
  ```

nodemon

Watch files and automatically restart a node application when changes are detected.

More information: https://nodemon.io.

- Execute the specified file and watch a specific file for changes:

  ```
  nodemon --inspect {{path/to/file.js}}
  ```

- Manually restart nodemon (note nodemon must already be active for this to work):

  ```
  rs
  ```

- Ignore specific files:

  ```
  nodemon --ignore {{path/to/file_or_directory}}
  ```

- Pass arguments to the node application:

  ```
  nodemon {{path/to/file.js}} {{arguments}}
  ```

- Run non-node scripts:

  ```
  nodemon --exec "{{python --verbose}}" {{path/to/file.py}}
  ```

nodenv

A tool to manage Node.js versions.

More information: https://github.com/nodenv/nodenv.

- Install a specific version of Node.js:

  ```
  nodenv install {{version}}
  ```

- Display a list of available versions:

  ```
  nodenv install --list
  ```

- Use a specific version of Node.js across the whole system:

  ```
  nodenv global {{version}}
  ```

- Use a specific version of Node.js with a directory:

  ```
  nodenv local {{version}}
  ```

- Display the Node.js version for the current directory:

  ```
  nodenv version
  ```

- Display the location of a Node.js installed command (e.g. `npm`):

  ```
  nodenv which {{command}}
  ```

nohup

Allows for a process to live when the terminal gets killed.

More information: https://www.gnu.org/software/coreutils/nohup.

- Run process that can live beyond the terminal:

```
nohup {{command}} {{command_options}}
```

nokogiri

An HTML, XML, SAX and Reader parser.

More information: https://nokogiri.org.

- Parse the contents of a URL or file:

  ```
  nokogiri {{url|path/to/file}}
  ```

- Parse as a specific type:

  ```
  nokogiri {{url|path/to/file}} --type {{xml|html}}
  ```

- Load a specific initialisation file before parsing:

  ```
  nokogiri {{url|path/to/file}} -C {{path/to/config_file}}
  ```

- Parse using a specific encoding:

  ```
  nokogiri {{url|path/to/file}} --encoding {{encoding}}
  ```

- Validate using a RELAX NG file:

  ```
  nokogiri {{url|path/to/file}} --rng {{url|path/to/file}}
  ```

nomad

Distributed, highly available, datacenter-aware scheduler.

More information: https://www.nomadproj ect.io/docs/commands/

- S how the status of nodes in the cluster:

  ```
  nom ad node statu s
  ```

- V alidate a j ob file:

  ```
  nom ad j ob v alidate {{path/to/file. nom ad}}
  ```

- P lan a j ob for execution on the cluster:

  ```
  nom ad j ob plan {{path/to/file. nom ad}}
  ```

- R un a j ob on the cluster:

  ```
  nom ad j ob ru n {{path/to/file. nom ad}}
  ```

- S how the status of j obs currently running on the cluster:

  ```
  nom ad j ob statu s
  ```

- S how the detailed status information about a specific j ob:

  ```
  nom ad j ob statu s {{j ob _nam e}}
  ```

- F ollow the logs of a specific allocation:

  ```
  nom ad alloc log s {{alloc_id}}
  ```

- S how the status of storage volumes:

  ```
  nom ad v olu m e statu s
  ```

noti

Monitor a process and trigger a banner notification.

More information: https://github.com/variadico/noti.

- Display a notification when tar finishes compressing files:

  ```
  noti {{tar -cjf example.tar.bz2 example/}}
  ```

- Display a notification even when you put it after the command to watch:

  ```
  {{command_to_watch}}; noti
  ```

- Monitor a process by PID and trigger a notification when the PID disappears:

  ```
  noti -w {{process_id}}
  ```

notmuch

Command-line based program for indexing, searching, reading, and tagging large collections of email messages.

More information: https://notmuchmail.org/manpages/.

- Configure for first use:

```
notmuch setup
```

- Add a tag for all messages matching a search term:

```
notmuch tag +{{custom_tag}} "{{search_term}}"
```

- Remove a tag for all messages matching a search term:

```
notmuch tag -{{custom_tag}} "{{search_term}}"
```

- Count messages matching the given search term:

```
notmuch count --output={{messages|threads}} "{{search_term}}"
```

- Search for messages matching the given search term:

```
notmuch search --format={{json|text}} --output={{summary|
threads|messages|files|tags}} "{{search_term}}"
```

- Limit the number of search results to X:

```
notmuch search --format={{json|text}} --output={{summary|
threads|messages|files|tags}} --limit={{X}} "{{search_term}}"
```

- Create a reply template for a set of messages:

```
notmuch reply --format={{default|headers-only}} --reply-
to={{sender|all}} "{{search_term}}"
```

now

Cloud platform for serverless deployment.

This command is deprecated. Se `vercel`, the updated version of this tool.

More information: https://zeit.co/now

- Deploy the current directory:

  `now`

- Display a list of deployments:

  `now list`

- Display information related to a deployment:

  `now inspect {{deployment_url}}`

- Remove a deployment:

  `now remove {{deployment_id}}`

- Log in into an account or create a new one:

  `now login`

- Initialize an example project (a new directory will be created):

  `now init`

npm-check

Check for outdated, incorrect, and unused npm package dependencies.

More information: https://www.npmj s.com/package/npm-check/

- Display a report of outdated, incorrect, and unused dependencies:

  `npm - check`

- Interactively update out-of-date packages:

  `npm - check - - u pdate`

- U pdate everything without prompting:

  `npm - check - - u pdate- all`

- Don' t check for unused packages:

  `npm - check - - sk ip- u nu sed`

npm-why

Identifies why an npm package is installed.

More information: https://www.npmj s.com/package/npm-why

- S how why an npm package is installed:

  ```
  npm - w hy {{pack ag e_nam e}}
  ```

npm

JavaScript and Node.js package manager.

Manage Node.js projects and their module dependencies.

More information: https://www.npmjs.com/

- Interactively create a `package.json` file:

  ```
  npm init
  ```

- Download all the packages listed as dependencies in package.json:

  ```
  npm install
  ```

- Download a specific version of a package and add it to the list of dependencies in `package.json`

  ```
  npm install {{module_name}}@{{version}}
  ```

- Download a package and add it to the list of dev dependencies in `package.json`

  ```
  npm install {{module_name}} --save-dev
  ```

- Download a package and install it globally:

  ```
  npm install --global {{module_name}}
  ```

- Uninstall a package and remove it from the list of dependencies in `package.json`

  ```
  npm uninstall {{module_name}}
  ```

- Print a tree of locally-installed dependencies:

  ```
  npm list
  ```

- List top-level globally installed modules:

  ```
  npm list --global --depth={{0}}
  ```

nproc

Print the number of processing units (normally CPUs) available.

More information: https://www.gnu.org/software/coreutils/nproc.

- Display the number of available processing units:

  ```
  nproc
  ```

- Display the number of installed processing units, including any inactive ones:

  ```
  nproc --all
  ```

- If possible, subtract a given number of units from the returned value:

  ```
  nproc --ignore {{count}}
  ```

npx

Execute binaries from `npm` packages.

More information: https://www.npmjs.com/package/npx

- Execute the binary from a given npm module:

  ```
  npx {{module_name}}
  ```

- In case a package has multiple binaries, specify the package name along with the binary:

  ```
  npx -p {{package_name}} {{module_name}}
  ```

- View help contents:

  ```
  npx --help
  ```

nrm

N P M registry manager.

H elps to easily switch between different npm registries.

More information: https://github.com/P ana/nrm

- List all registries:

  ```
  nrm ls
  ```

- Change to a particular registry:

  ```
  nrm u se {{reg istry}}
  ```

- S how the response time for all registries:

  ```
  nrm test
  ```

- Add a custom registry:

  ```
  nrm add {{reg istry}} {{u rl}}
  ```

- Delete a registry:

  ```
  nrm del {{reg istry}}
  ```

nslookup

Query name server(s) for various domain records.

- Query your system's default name server for an IP address (A record) of the domain:

  ```
  nslookup {{example.com}}
  ```

- Query a given name server for a NS record of the domain:

  ```
  nslookup -type=NS {{example.com}} {{8.8.8.8}}
  ```

- Query for a reverse lookup (PTR record) of an IP address:

  ```
  nslookup -type=PTR {{54.240.162.118}}
  ```

- Query for ANY available records using TCP protocol:

  ```
  nslookup -vc -type=ANY {{example.com}}
  ```

- Query a given name server for the whole zone file (zone transfer) of the domain using TCP protocol:

  ```
  nslookup -vc -type=AXFR {{example.com}} {{name_server}}
  ```

- Query for a mail server (MX record) of the domain, showing details of the transaction:

  ```
  nslookup -type=MX -debug {{example.com}}
  ```

- Query a given name server on a specific port number for a TXT record of the domain:

  ```
  nslookup -port={{port_number}} -type=TXT {{example.com}} {{name_server}}
  ```

nu

Nushell ("a new type of shell") takes a modern, structured approach to your command-line.

More information: https://www.nushell.sh.

- Start an interactive shell session:

```
nu
```

- Execute a command and then exit:

```
nu --commands "{{command}}"
```

- Execute a script:

```
nu {{path/to/script.nu}}
```

- Execute a script with logging:

```
nu --loglevel {{error|warn|info|debug|trace}} {{path/to/script.nu}}
```

- Print the Nushell version:

```
nu --version
```

nudoku

Sudoku game in terminal.

More information: https://jubalh.github.io/nudoku/

- Start a sudoku game:

  ```
  nudoku
  ```

- Choose the difficulty of the game:

  ```
  nudoku -d {{easy|normal|hard}}
  ```

- Navigate the board:

  ```
  {{h|j|k|l}} OR {{Left|Down|Up|Right}} arrow key
  ```

- Delete a number:

  ```
  {{Backspace|x}}
  ```

- Get a hint:

  ```
  H
  ```

- See the complete solution:

  ```
  S
  ```

- Create a new puzzle:

  ```
  N
  ```

- Quit the game:

  ```
  Q
  ```

numfmt

Convert numbers to and from human-readable strings.

More information: https://www.gnu.org/software/coreutils/numfmt.

- Convert 1.5K (SI Units) to 1500:

```
numfmt --from=si {{1.5K}}
```

- Convert 5th field (1-indexed) to IEC Units without converting header:

```
ls -l | numfmt --header={{1}} --field={{5}} --to={{iec}}
```

- Convert to IEC units, pad with 5 characters, left aligned:

```
du -s * | numfmt --to={{iec}} --format="{{%-5f}}"
```

nvidia-smi

Aid the management and monitoring of NVIDIA GPU devices.

More information: https://developer.nvidia.com/nvidia-system-management-interface.

- Display information on all available GPUs and processes using them:

  ```
  nvidia-smi
  ```

- Display more detailed GPU information:

  ```
  nvidia-smi --query
  ```

- Monitor overall GPU usage with 1-second update interval:

  ```
  nvidia-smi dmon
  ```

nvim

Neovim, a programmer's text editor based on Vim, provides several modes for different kinds of text manipulation.

Pressing `i` enters edit mode. `<Esc>` goes back to normal mode, which doesn't allow regular text insertion.

More information: https://neovim.io.

- Open a file:

  ```
  nvim {{file}}
  ```

- Enter text editing mode (insert mode):

  ```
  <Esc> i
  ```

- Copy ("yank") or cut ("delete") the current line (paste it with `P`):

  ```
  <Esc> {{yy|dd}}
  ```

- Undo the last operation:

  ```
  <Esc> u
  ```

- Search for a pattern in the file (press `n`/`N` to go to next/previous match):

  ```
  <Esc> /{{search_pattern}}<Enter>
  ```

- Perform a regular expression substitution in the whole file:

  ```
  <Esc> :%s/{{regular_expression}}/{{replacement}}/g <Enter>
  ```

- Save (write) the file, and quit:

  ```
  <Esc> :wq <Enter>
  ```

- Quit without saving:

  ```
  <Esc> :q! <Enter>
  ```

nvm

Install, uninstall or switch between Node.js versions.

Supports version numbers like "0.12" or "v4.2", and labels like "stable", "system", etc.

More information: https://github.com/creationix/nvm.

- Install a specific version of Node.js:

  ```
  nvm install {{node_version}}
  ```

- Use a specific version of Node.js in the current shell:

  ```
  nvm use {{node_version}}
  ```

- Set the default Node.js version:

  ```
  nvm alias default {{node_version}}
  ```

- List all available Node.js versions and highlight the default one:

  ```
  nvm list
  ```

- Uninstall a given Node.js version:

  ```
  nvm uninstall {{node_version}}
  ```

- Launch the REPL of a specific version of Node.js:

  ```
  nvm run {{node_version}} --version
  ```

- Execute a script in a specific version of Node.js:

  ```
  nvm exec {{node_version}} node {{app.js}}
  ```

obj dump

View information about object files.

- Display the file header information:

  ```
  objdump -f {{binary}}
  ```

- Display the dis-assembled output of executable sections:

  ```
  objdump -d {{binary}}
  ```

- Display a complete binary hex dump of all sections:

  ```
  objdump -s {{binary}}
  ```

obs

Open Broadcaster Software.

Video recording and livestreaming program.

- Launch OBS:

  ```
  obs
  ```

- Launch OBS in portable mode:

  ```
  obs --portable
  ```

- Automatically start recording a video on launch:

  ```
  obs --startrecording
  ```

- Automatically start the replay buffer on launch:

  ```
  obs --startreplaybuffer
  ```

- Automatically start streaming on launch:

  ```
  obs --startstreaming
  ```

- Minimise to the system tray on launch:

  ```
  obs --minimize-to-tray
  ```

- Make the log more verbose (for debugging):

  ```
  obs --verbose
  ```

oc

The OpenShift Container Platform CLI.

Allows for application and container management.

More information: https://docs.openshift.com/container-platform/3.11/cli_reference/get_started_cli.html

- Log in to the OpenShift Container Platform server:

  ```
  oc login
  ```

- Create a new project:

  ```
  oc new-project {{project_name}}
  ```

- Switch to an existing project:

  ```
  oc project {{project_name}}
  ```

- Add a new application to a project:

  ```
  oc new-app {{repo_url}} --name {{application}}
  ```

- Open a remote shell session to a container:

  ```
  oc rsh {{pod_name}}
  ```

- List pods in a project:

  ```
  oc get pods
  ```

- Log out from the current session:

  ```
  oc logout
  ```

ocaml

The OCaml repl (read-evaluate-print-loop).

Interprets OCaml commands.

More information: https://ocaml.org.

- Read OCaml commands from the user and execute them:

  `ocaml`

- Read OCaml commands from a file and execute them:

  `ocaml {{path/to/file.ml}}`

ocamlc

T he O Caml bytecode compiler.

P roduces executables runnable by the O Caml interpreter.

More information: https://ocaml.org.

- Create a binary from a source file:

  `ocam lc {{path/to/sou rce_file. m l}}`

- Create a named binary from a source file:

  `ocam lc - o {{path/to/b inary}} {{path/to/sou rce_file. m l}}`

- Automatically generate a module signature (interface) file:

  `ocam lc - i {{path/to/sou rce_file. m l}}`

ocamlfind

T he findlib package manager for O Caml.

S implifies linking executables with external libraries.

More information: http://proj ects.camlcity.org/proj ects/findlib.html

- Compile a source file to a native binary and link with packages:

  ```
  ocam lfind ocam lopt - pack ag e {{pack ag e1 }}, {{pack ag e2 }} -
  link pk g - o {{execu tab le}} {{sou rce_file. m l}}
  ```

- Compile a source file to a bytecode binary and link with packages:

  ```
  ocam lfind ocam lc - pack ag e {{pack ag e1 }}, {{pack ag e2 }} - link pk g
  - o {{execu tab le}} {{sou rce_file. m l}}
  ```

- Cross-compile for a different platform:

  ```
  ocam lfind - toolchain {{cross- toolchain}} ocam lopt - o
  {{execu tab le}} {{sou rce_file. m l}}
  ```

ocamlopt

The OCaml native code compiler.

Produces native executables, e.g. ELF on Linux.

More information: https://ocaml.org.

- Compile a source file:

  ```
  ocamlopt -o {{path/to/binary}} {{path/to/source_file.ml}}
  ```

- Compile with debugging enabled:

  ```
  ocamlopt -g -o {{path/to/binary}} {{path/to/source_file.ml}}
  ```

ocrmypdf

Generate a searchable PDF or PDF/A from a scanned PDF or an image of text.

More information: https://ocrmypdf.readthedocs.io/en/latest/cookbook.html.

- Create a new searchable PDF/A file from a scanned PDF or image file:

```
ocrmypdf {{path/to/input_file}} {{path/to/output.pdf}}
```

- Replace a scanned PDF file with a searchable PDF file:

```
ocrmypdf {{path/to/input.pdf}}
```

- Skip pages of a mixed-format input PDF file that already contain text:

```
ocrmypdf --skip-text {{path/to/input.pdf}} {{path/to/
output.pdf}}
```

- Clean, de-skew, and rotate pages of a poor scan:

```
ocrmypdf --clean --deskew --rotate-pages {{path/to/
input_file}} {{path/to/output.pdf}}
```

- Set the metadata of the searchable PDF file:

```
ocrmypdf --title "{{title}}" --author "{{author}}" --subject
"{{subject}}" --keywords "{{keyword; keyphrase;
...}}" {{path/to/input_file}} {{path/to/output.pdf}}
```

- Display help:

```
ocrmypdf --help
```

od

Display file contents in octal, decimal or hexadecimal format.

Optionally display the byte offsets and/or printable representation for each line.

More information: https://www.gnu.org/software/coreutils/od.

- Display file using default settings: octal format, 8 bytes per line, byte offsets in octal, and duplicate lines replaced with *:

```
od {{path/to/file}}
```

- Display file in verbose mode, i.e. without replacing duplicate lines with *:

```
od -v {{path/to/file}}
```

- Display file in hexadecimal format (2-byte units), with byte offsets in decimal format:

```
od --format={{x}} --address-radix={{d}} -v {{path/to/file}}
```

- Display file in hexadecimal format (1-byte units), and 4 bytes per line:

```
od --format={{x1}} --width={{4}} -v {{path/to/file}}
```

- Display file in hexadecimal format along with its character representation, and do not print byte offsets:

```
od --format={{xz}} --address-radix={{n}} -v {{path/to/file}}
```

- Read only 100 bytes of a file starting from the 500th byte:

```
od --read-bytes={{100}} --skip-bytes={{500}} -v {{path/to/file}}
```

odps auth

User authorities in ODPS (Open Data Processing Service).

See also **odps**.

More information: https://www.alibabacloud.com/help/doc-detail/27971.htm

- Add a user to the current project:

  ```
  add user {{username}};
  ```

- Grant a set of authorities to a user:

  ```
  grant {{action_list}} on {{object_type}} {{object_name}} to
  user {{username}};
  ```

- Show authorities of a user:

  ```
  show grants for {{username}};
  ```

- Create a user role:

  ```
  create role {{role_name}};
  ```

- Grant a set of authorities to a role:

  ```
  grant {{action_list}} on {{object_type}} {{object_name}} to
  role {{role_name}};
  ```

- Describe authorities of a role:

  ```
  desc role {{role_name}};
  ```

- Grant a role to a user:

  ```
  grant {{role_name}} to {{username}};
  ```

odps func

Manage functions in ODPS (Open Data Processing Service).

See also **`odps`**.

More information: https://www.alibabacloud.com/help/doc-detail/27971.htm

- Show functions in the current project:

  ```
  list functions;
  ```

- Create a Java function using a `.jar` resource:

  ```
  create function {{func_name}} as {{path.to.package.Func}}
  using '{{package.jar}}';
  ```

- Create a Python function using a `.py` resource:

  ```
  create function {{func_name}} as {{script.Func}} using
  '{{script.py}}';
  ```

- Delete a function:

  ```
  drop function {{func_name}};
  ```

odps inst

Manage instances in ODPS (Open Data Processing Service).

See also **`odps`**.

More information: https://www.alibabacloud.com/help/doc-detail/27971.htm

- Show instances created by current user:

  ```
  show instances;
  ```

- Describe the details of an instance:

  ```
  desc instance {{instance_id}};
  ```

- Check the status of an instance:

  ```
  status {{instance_id}};
  ```

- Wait on the termination of an instance, printing log and progress information until then:

  ```
  wait {{instance_id}};
  ```

- Kill an instance:

  ```
  kill {{instance_id}};
  ```

odps resource

Manage resources in ODPS (Open Data Processing Service).

See also **`odps`**.

More information: https://www.alibabacloud.com/help/doc-detail/27971.htm

- Show resources in the current project:

  ```
  list resources;
  ```

- Add file resource:

  ```
  add file {{filename}} as {{alias}};
  ```

- Add archive resource:

  ```
  add archive {{archive.tar.gz}} as {{alias}};
  ```

- Add .jar resource:

  ```
  add jar {{package.jar}};
  ```

- Add .py resource:

  ```
  add py {{script.py}};
  ```

- Delete resource:

  ```
  drop resource {{resource_name}};
  ```

odps table

Create and modify tables in ODPS (Open Data Processing Service).

See also **`odps`**.

More information: https://www.alibabacloud.com/help/doc-detail/27971.htm

- Create a table with partition and lifecycle:

  ```
  create table {{table_name}} ({{col}} {{type}}) partitioned by
  ({{col}} {{type}}) lifecycle {{days}};
  ```

- Create a table based on the definition of another table:

  ```
  create table {{table_name}} like {{another_table}};
  ```

- Add partition to a table:

  ```
  alter table {{table_name}} add partition
  ({{partition_spec}});
  ```

- Delete partition from a table:

  ```
  alter table {{table_name}} drop partition
  ({{partition_spec}});
  ```

- Delete table:

  ```
  drop table {{table_name}};
  ```

odps tunnel

Data tunnel in ODPS (Open Data Processing Service).

See also **odps**.

More information: https://www.alibabacloud.com/help/doc-detail/27971.htm

- Download table to local file:

  ```
  tunnel download {{table_name}} {{file}};
  ```

- Upload local file to a table partition:

  ```
  tunnel upload {{file}} {{table_name}}/{{partition_spec}};
  ```

- Upload table specifying field and record delimiters:

  ```
  tunnel upload {{file}} {{table_name}} -fd {{field_delim}} -rd
  {{record_delim}};
  ```

- Upload table using multiple threads:

  ```
  tunnel upload {{file}} {{table_name}} -threads {{num}};
  ```

odps

Aliyun ODPS (Open Data Processing Service) command-line tool.

More information: https://www.alibabacloud.com/help/doc-detail/27971.htm

- Start the command-line with a custom configuration file:

  ```
  odpscmd --config={{odps_config.ini}}
  ```

- Switch current project:

  ```
  use {{project_name}};
  ```

- Show tables in the current project:

  ```
  show tables;
  ```

- Describe a table:

  ```
  desc {{table_name}};
  ```

- Show table partitions:

  ```
  show partitions {{table_name}};
  ```

- Describe a partition:

  ```
  desc {{table_name}} partition ({{partition_spec}});
  ```

offl ineimap

S ynchroniz e a remote IMAP server with local Maildir folders.

More information: http://www.offl ineimap.org

- S ynchroniz e once, without enabling autorefresh:

  ```
  offlineim ap - o
  ```

- S ynchroniz e a specific account:

  ```
  offlineim ap - a {{accou nt}}
  ```

- S ynchroniz e a specific folder:

  ```
  offlineim ap - f {{folder}}
  ```

ogr2ogr

Convert Simple Features data between file formats.

More information: https://gdal.org/programs/ogr2_ogr.html#_ogr2_ogr

- Convert a Shapefile into a GeoPackage:

```
ogr2ogr -f GPKG {{output}}.gpkg {{input}}.shp
```

- Change coordinate reference system of a GeoPackage from EPSG:4326 to EPSG:3857:

```
ogr2ogr -s_srs {{EPSG:4326}} -t_srs {{EPSG:3857}} -f GPKG {{output}}.gpkg {{input}}.gpkg
```

- Convert a CSV file into a GeoPackage, specifying the names of the coordinate columns and assigning a coordinate reference system:

```
ogr2ogr -f GPKG {{output}}.gpkg {{input}}.csv -oo X_POSSIBLE_NAMES={{longitude}} -oo Y_POSSIBLE_NAMES={{latitude}} -a_srs {{EPSG:4326}}
```

- Load a GeoPackage into a PostgIS database:

```
ogr2ogr -f "PostgreSQL" PG:dbname="{{database_name}}" {{input}}.gpkg
```

- Clip layers of a GeoPackage file to the given bounding box:

```
ogr2ogr -spat {{min_x}} {{min_y}} {{max_x}} {{max_y}} -f GPKG {{output}}.gpkg {{input}}.gpkg
```

ogrinfo

List information about an O G R -supported data source.

More information: https://gdal.org/programs/ogrinfo.html.

- List layers of a G eoP ackage:

  ```
  og rinfo {{inpu t}}. g pk g
  ```

- G et detailed information about a specific layer of a G eoP ackage:

  ```
  og rinfo {{inpu t}}. g pk g  {{layer_nam e}}
  ```

- O nly show summary information about a specific layer of a G eoP ackage:

  ```
  og rinfo - so {{inpu t}}. g pk g  {{layer_nam e}}
  ```

ohdear-cli

An unofficial O h Dear CLI written with Laravel Z ero.

More information: https://github.com/nunomaduro/ohdear-cli.

- Display details about the currently authenticated user:

  ```
  ohdear- cli m e
  ```

- Add a new site to O h Dear:

  ```
  ohdear- cli sites: add {{u rl}}
  ```

- Display a list of sites and their current status:

  ```
  ohdear- cli sites: list
  ```

- Display details about a specific site:

  ```
  ohdear- cli sites: show  {{site_id}}
  ```

okular

A universal document viewer.

More information: https://okular.kde.org/.

- Launch O kular:

  ```
  ok u lar
  ```

- O pen specific documents in O kular:

  ```
  ok u lar {{path/to/file1  path/to/file2  . . . }}
  ```

- O pen a document at a specific page:

  ```
  ok u lar - - pag e {{pag e_nu m b er}} {{path/to/file}}
  ```

- O pen a document in presentation mode:

  ```
  ok u lar - - presentation {{path/to/file}}
  ```

- O pen a document and start the print dialog:

  ```
  ok u lar - - print {{path/to/file}}
  ```

- O pen a document and search for a specific string:

  ```
  ok u lar - - find {{search_string }} {{path/to/file}}
  ```

omf

O h My F ish, the F ishshell F ramework.

Install packages to extend and modify the fish shell.

More information: https://github.com/oh-my-fish/oh-my-fish.

- Install one or more packages:

  ```
  om f install {{nam e}}
  ```

- List installed packages:

  ```
  om f list
  ```

- List available themes:

  ```
  om f them e
  ```

- Apply a theme:

  ```
  om f them e {{nam e}}
  ```

- R emove a theme or package:

  ```
  om f rem ov e {{nam e}}
  ```

- U ninstall O h My F ish:

  ```
  om f destroy
  ```

opam

O Caml P ackage Manager.

Manage O Caml compilers, tools and libraries.

More information: https://opam.ocaml.org/.

- Initializ e opam for first use:

  ```
  opam init
  ```

- S earch for packages:

  ```
  opam search {{pack ag e_nam e}}
  ```

- Install a package and all of its dependencies:

  ```
  opam install {{pack ag e_nam e}}
  ```

- Display detailed information about a package:

  ```
  opam show {{pack ag e_nam e}}
  ```

- List all installed packages:

  ```
  opam list
  ```

- U pdate the local package database:

  ```
  opam u pdate
  ```

- U pgrade all installed packages:

  ```
  opam u pg rade
  ```

- Display all commands:

  ```
  opam help
  ```

openconnect

A V P N client, for Cisco AnyConnect V P N s and others.

More information: https://www.infradead.org/openconnect/manual.html.

- Connect to a server:

  ```
  openconnect {{v pn. exam ple. org }}
  ```

- Connect to a server, forking into the background:

  ```
  openconnect - - b ack g rou nd {{v pn. exam ple. org }}
  ```

- T erminate the connection that is running in the background:

  ```
  k illall - S I G I N T  openconnect
  ```

- Connect to a server, reading options from a config file:

  ```
  openconnect - - config = {{path/to/file}} {{v pn. exam ple. org }}
  ```

- Connect to a server and authenticate with a specific S S L client certificate:

  ```
  openconnect - - certificate= {{path/to/file}}
  {{v pn. exam ple. org }}
  ```

openscad

Software for creating solid 3D CAD objects.

More information: https://openscad.org.

- Open a file:

  ```
  openscad {{path/to/button.scad}}
  ```

- Convert a file to STL:

  ```
  openscad -o {{path/to/button.stl}} {{path/to/button.scad}}
  ```

- Render a file to PNG in a specific colorscheme:

  ```
  openscad -o {{path/to/button.png}} --colorscheme {{Sunset}}
  {{path/to/button.scad}}
  ```

openssl dgst

OpenSSL command to generate digest values and perform signature operations.

More information: https://www.openssl.org/docs/manmaster/man1/openssl-dgst.html

- Calculate the SHA256 digest for a file, saving the result to a specific file:

```
openssl dgst -sha256 -binary -out {{output_file}} {{input_file}}
```

- Sign a file using an RSA key, saving the result to a specific file:

```
openssl dgst -sign {{private_key_file}} -sha256 -sigopt rsa_padding_mode:pss -out {{output_file}} {{input_file}}
```

- Verify an RSA signature:

```
openssl dgst -verify {{public_key_file}} -signature {{signature_file}} -sigopt rsa_padding_mode:pss {{signature_message_file}}
```

- Sign a file using and ECDSA key:

```
openssl dgst -sign {{private_key_file}} -sha256 -out {{output_file}} {{input_file}}
```

- Verify an ECDSA signature:

```
openssl dgst -verify {{public_key_file}} -signature {{signature_file}} {{signature_message_file}}
```

openssl genpkey

OpenSSL command to generate asymmetric key pairs.

More information: https://www.openssl.org/docs/manmaster/man1/openssl-genpkey.html.

- Generate an RSA private key of 2048 bits, saving it to a specific file:

```
openssl genpkey -algorithm rsa -pkeyopt rsa_keygen_bits:{{2048}} -out {{filename.key}}
```

- Generate an elliptic curve private key using the curve `prime256v1`, saving it to a specific file:

```
openssl genpkey -algorithm EC -pkeyopt ec_paramgen_curve:{{prime256v1}} -out {{filename.key}}
```

- Generate an `ED25519` elliptic curve private key, saving it to a specific file:

```
openssl genpkey -algorithm {{ED25519}} -out {{filename.key}
```

openssl genrsa

OpenSSL command to generate RSA private keys.

More information: https://www.openssl.org/docs/manmaster/man1/openssl-genrsa.html.

- Generate an RSA private key of 2048 bits to stdout:

```
openssl genrsa
```

- Save an RSA private key of an arbitrary number of bits to the output file:

```
openssl genrsa -out {{output_file.key}} {{1234}}
```

- Generate an RSA private key and encrypt it with AES256 (you will be prompted for a passphrase):

```
openssl genrsa {{-aes256}}
```

openssl prime

OpenSSL command to compute prime numbers.

More information: https://www.openssl.org/docs/manmaster/man1/openssl-prime.html.

- Generate a 2048 bit prime number and display it in hexadecimal:

```
openssl prime -generate -bits 2048 -hex
```

- Check if a given number is prime:

```
openssl prime {{number}}
```

openssl req

OpenSSL command to manage PKCS#10 Certificate Signing Requests.

More information: https://www.openssl.org/docs/manmaster/man1/openssl-req.html

- Generate a certificate signing request to be sent to a certificate authority:

```
openssl req -new -sha256 -key {{filename.key}} -out {{filename.csr}}
```

- Generate a self-signed certificate and a corresponding key-pair, storing both in a file:

```
openssl req -new -x509 -newkey {{rsa}}:{{4096}} -keyout {{filename.key}} -out {{filename.cert}} -subj "{{/CN=foobar}}" -days {{365}}
```

openssl s_client

OpenSSL command to create TLS client connections.

More information: https://www.openssl.org/docs/manmaster/man1/openssl-s_client.html

- Display the start and expiry dates for a domain's certificate:

```
openssl s_client -connect {{host}}:{{port}} 2>/dev/null |
openssl x509 -noout -dates
```

- Display the certificate presented by an SSL/TLS server:

```
openssl s_client -connect {{host}}:{{port}} </dev/null
```

- Display the complete certificate chain of an HTTPS server:

```
openssl s_client -connect {{host}}:443 -showcerts </dev/null
```

openssl ts

OpenSSL command to generate and verify timestamps.

More information: https://www.openssl.org/docs/manmaster/man1/openssl-ts.html.

- Generate a SHA-512 timestamp request of a specific file and output it to `file.tsq`:

  ```
  openssl ts -query -data {{path/to/file}} -sha512 -out {{path/to/file.tsq}}
  ```

- Check the date and metadata of a specific timestamp response file:

  ```
  openssl ts -reply -in {{path/to/file.tsr}} -text
  ```

- Verify a timestamp request file and a timestamp response file from the server with an SSL certificate file:

  ```
  openssl ts -verify -in {{path/to/file.tsr}} -queryfile {{path/to/file.tsq}} -partial_chain -CAfile {{path/to/cert.pem}}
  ```

- Create a timestamp response for request using key and signing certificate and output it to `file.tsr`:

  ```
  openssl ts -reply -queryfile {{path/to/file.tsq}} -inkey {{path/to/tsakey.pem}} -signer tsacert.pem -out {{path/to/file.tsr}}
  ```

openssl x509

OpenSSL command to manage X.509 certificates.

More information: https://www.openssl.org/docs/manmaster/man1/openssl-x509.html

- Display certificate information:

```
openssl x509 -in {{filename.crt}} -noout -text
```

- Display a certificate's expiration date:

```
openssl x509 -enddate -noout -in {{filename.pem}}
```

- Convert a certificate between binary DER encoding and textual PEM encoding:

```
openssl x509 -inform {{der}} -outform {{pem}} -in
{{original_certificate_file}} -out
{{converted_certificate_file}}
```

- Store a certificate's public key in a file:

```
openssl x509 -in {{certificate_file}} -noout -pubkey -out
{{output_file}}
```

openssl

OpenSSL cryptographic toolkit.

More information: https://www.openssl.org.

- Print a list of available subcommands:

  ```
  openssl help
  ```

- Print options for a specific subcommand:

  ```
  openssl help {{x509}}
  ```

- Print the version of OpenSSL:

  ```
  openssl version
  ```

openttd

Open source clone of the Microprose game "Transport Tycoon Deluxe".

More information: https://www.openttd.org.

- Start a new game:

  ```
  openttd -g
  ```

- Load save game at start:

  ```
  openttd -g {{path/to/file}}
  ```

- Start with the specified window resolution:

  ```
  openttd -r {{1920x1080}}
  ```

- Start with a custom configuration file:

  ```
  openttd -c {{path/to/file}}
  ```

- Start with selected video, sound, and music drivers:

  ```
  openttd -v {{video_driver}} -s {{sound_driver}} -m {{music_driver}}
  ```

- Start a dedicated server, forked in the background:

  ```
  openttd -f -D {{host}}:{{port}}
  ```

- Join a server with a password:

  ```
  openttd -n {{host}}:{{port}}#{{player_name}} -p {{password}}
  ```

openvpn

OpenVPN client and daemon binary.

More information: https://openvpn.net/.

- Connect to server using a config file:

```
sudo openvpn {{path/to/client.conf}}
```

- Try to set up an insecure peer-to-peer tunnel on bob.example.com host:

```
sudo openvpn --remote {{alice.example.com}} --dev {{tun1}} -
ifconfig {{10.4.0.1}} {{10.4.0.2}}
```

- Connect to the awaiting bob.example.com host without encryption:

```
sudo openvpn --remote {{bob.example.com}} --dev {{tun1}} -
ifconfig {{10.4.0.2}} {{10.4.0.1}}
```

- Create a cryptographic key and save it to file:

```
openvpn --genkey --secret {{path/to/key}}
```

- Try to set up a peer-to-peer tunnel on bob.example.com host with a static key:

```
sudo openvpn --remote {{alice.example.com}} --dev {{tun1}} -
ifconfig {{10.4.0.1}} {{10.4.0.2}} --secret {{path/to/key
```

- Connect to the awaiting bob.example.com host with the same static key as on bob.example.com:

```
sudo openvpn --remote {{bob.example.com}} --dev {{tun1}} -
ifconfig {{10.4.0.2}} {{10.4.0.1}} --secret {{path/to/key
```

opt

A tool that takes LLVM source files and runs specified optimizations and/or analyses on them.

More information: https://llvm.org/docs/CommandGuide/opt.html

- Run an optimization or analysis on a bitcode file:

```
opt -{{passname}} {{path/to/file.bc}} -S -o {{file_opt.bc}}
```

- Output the Control Flow Graph of a function to a `dot` file:

```
opt {{-dot-cfg}} -S {{path/to/file.bc}} -disable-output
```

- Optimize the program at level 2 and output the result to another file:

```
opt -O2 {{path/to/file.bc}} -S -o {{path/to/output_file.bc}}
```

optipng

PNG image file optimization utility.

More information: http://optipng.sourceforge.net.

- Compress a PNG with default settings:

  ```
  optipng {{path/to/file.png}}
  ```

- Compress a PNG with best compression:

  ```
  optipng -o{{7}} {{path/to/file.png}}
  ```

- Compress a PNG with fastest compression:

  ```
  optipng -o{{0}} {{path/to/file.png}}
  ```

- Compress a PNG and add interlacing:

  ```
  optipng -i {{1}} {{path/to/file.png}}
  ```

- Compress a PNG and preserve all metadata (including file timestamps):

  ```
  optipng -preserve {{path/to/file.png}}
  ```

- Compress a PNG and remove all metadata:

  ```
  optipng -strip all {{path/to/file.png}}
  ```

opusenc

Convert WAV or FLAC audio to Opus.

More information: https://opus-codec.org/docs/opus-tools/opusenc.html.

- Convert WAV to Opus using default options:

```
opusenc {{path/to/input.wav}} {{path/to/output.opus}}
```

- Convert stereo audio at the highest quality level:

```
opusenc --bitrate {{512}} {{path/to/input.wav}} {{path/to/output.opus}}
```

- Convert 5.1 surround sound audio at the highest quality level:

```
opusenc --bitrate {{1536}} {{path/to/input.flac}} {{path/to/output.opus}}
```

- Convert speech audio at the lowest quality level:

```
opusenc {{path/to/input.wav}} --downmix-mono --bitrate {{6}} {{path/to/output.opus}}
```

osage

Render an image of a **clustered** network graph from a **graphviz** file.

Layouts: **dot**, **neato**, **twopi**, **circo**, **fdp**, **sfdp**, **osage** & **patchwork**

More information: https://graphviz.org/doc/info/command.html

- Render a `png` image with a filename based on the input filename and output format (uppercase -O):

  ```
  osage -T {{png}} -O {{path/to/input.gv}}
  ```

- Render a `svg` image with the specified output filename (lowercase -o):

  ```
  osage -T {{svg}} -o {{path/to/image.svg}} {{path/to/input.gv}}
  ```

- Render the output in `ps`, `pdf`, `svgz`, `fig`, `png`, `gif`, `jpg`, `jpeg`, or `dot` format:

  ```
  osage -T {{format}} -O {{path/to/input.gv}}
  ```

- Render a `gif` image using stdin and stdout:

  ```
  echo "{{digraph {this->that}}}" | osage -T {{gif}} > {{path/to/image.gif}}
  ```

- Display help:

  ```
  osage -?
  ```

p4

Perforce Version Control System.

More information: https://www.perforce.com/manuals/cmdref.

- Log in to the Perforce service:

```
p4 login -a
```

- Create a client:

```
p4 client
```

- Copy files from depot into the client workspace:

```
p4 sync
```

- Create or edit changelist description:

```
p4 change
```

- Open a file to edit:

```
p4 edit -c {{changelist_number}} {{filename}}
```

- Open a new file to add it to the depot:

```
p4 add
```

- Display list of files modified by changelist:

```
p4 describe -c {{changelist_number}}
```

- Submit a changelist to the depot:

```
p4 submit -c {{changelist_number}}
```

p5

Template builder and sketch manager for p5.js.

More information: https://www.npmjs.com/package/p5-manager

- Create a new p5 collection:

  ```
  p5 new {{collection_name}}
  ```

- Generate a new p5 project (should be run from collection directory):

  ```
  p5 generate {{project_name}}
  ```

- Run the p5 manager server:

  ```
  p5 server
  ```

- Update libraries to their latest versions:

  ```
  p5 update
  ```

p7zip

Wrapper of 7-Zip file archiver with high compression ratio.

Internally executes either 7z a or 7z r command.

More information: http://p7zip.sourceforge.net

- Archive a file, replacing it with a 7zipped compressed version:

  ```
  p7zip {{path/to/file}}
  ```

- Archive a file keeping the input file:

  ```
  p7zip -k {{path/to/file}}
  ```

- Decompress a file, replacing it with the original uncompressed version:

  ```
  p7zip -d {{compressed.ext}}.7z
  ```

- Decompress a file keeping the input file:

  ```
  p7zip -d -k {{compressed.ext}}.7z
  ```

- Skip some checks and force compression or decompression:

  ```
  p7zip -f {{path/to/file}}
  ```

paci

A package manager for bash scripts.

- U pdate the list of available packages and versions (it' s recommended to run this before other `paci` commands):

  ```
  paci refresh
  ```

- Configure its behaviour:

  ```
  paci config u re
  ```

- S earch for a given package:

  ```
  paci search {{pack ag e}}
  ```

- Install a package:

  ```
  paci install {{pack ag e}}
  ```

- U pdate a package:

  ```
  paci u pdate {{pack ag e}}
  ```

packer

Build automated machine images.

More information: https://www.packer.io/.

- Build an image:

  ```
  packer build {{path/to/config.json}}
  ```

- Check the syntax of a Packer image config:

  ```
  packer validate {{path/to/config.json}}
  ```

packtpub

Download freely offered books from packtpub.com.

More information: https://github.com/vladimyr/packtpub-cli.

- Download the daily offer book into the current directory with the specified book format (defaults to `pdf`):

  ```
  pack tpu b  dow nload - - type {{pdf| eb u p| m ob i}}
  ```

- Download the daily offer book into the specified directory:

  ```
  pack tpu b  dow nload - - dir {{path/to/directory}}
  ```

- S tart an interactive login to packtpub.com:

  ```
  pack tpu b  log in
  ```

- Log out from packtpub.com:

  ```
  pack tpu b  log ou t
  ```

- Display the daily offer:

  ```
  pack tpu b  v iew - offer
  ```

- O pen the daily offer in the default web browser:

  ```
  pack tpu b  v iew - offer
  ```

- Display the currently logged-in user:

  ```
  pack tpu b  w hoam i
  ```

pactl

Control a running PulseAudio sound server.

- List all sinks (or other types - sinks are outputs and sink-inputs are active audio streams):

  ```
  pactl list {{sinks}} short
  ```

- Change the default sink (output) to 1 (the number can be retrieved via the `list` subcommand):

  ```
  pactl set-default-sink {{1}}
  ```

- Move sink-input 627 to sink 1:

  ```
  pactl move-sink-input {{627}} {{1}}
  ```

- Set the volume of sink 1 to 75%:

  ```
  pactl set-sink-volume {{1}} {{0.75}}
  ```

- Toggle mute on the default sink (using the special name `@DEFAULT_SINK@`):

  ```
  pactl set-sink-mute {{@DEFAULT_SINK@}} toggle
  ```

pageres

Capture screenshots of websites in various resolutions.

More information: https://github.com/sindresorhus/pageres-cli.

- Take multiple screenshots of multiple URLs at different resolutions:

```
pageres {{https://example.com/}} {{https://example2.com/}} {{1366x768}} {{1600x900}}
```

- Provide specific options for a URL, overriding global options:

```
pageres [{{https://example.com/}} {{1366x768}} --no-crop] [{{https://example2.com/}} {{1024x768}}] --crop
```

- Provide a custom filename template:

```
pageres {{https://example.com/}} {{1024x768}} --filename={{'<%= date %> - <%= url %>'}}
```

- Capture a specific element on a page:

```
pageres {{https://example.com/}} {{1366x768}} --selector='{{.page-header}}'
```

- Hide a specific element:

```
pageres {{https://example.com/}} {{1366x768}} --hide='{{.page-header}}'
```

- Capture a screenshot of a local file:

```
pageres {{local_file_path.html}} {{1366x768}}
```

pamixer

A simple command-line mixer for PulseAudio.

More information: https://github.com/cdemoulins/pamixer.

- List all sinks and sources with their corresponding IDs:

  ```
  pamixer --list-sinks --list-sources
  ```

- Set the volume to 75% on the default sink:

  ```
  pamixer --set-volume {{75}}
  ```

- Toggle mute on a sink other than the default:

  ```
  pamixer --toggle-mute --sink {{ID}}
  ```

- Increase the volume on default sink by 5%:

  ```
  pamixer --increase {{5}}
  ```

- Decrease the volume on a source by 5%:

  ```
  pamixer --decrease {{5}} --source {{ID}}
  ```

- Use the allow boost option to increase, decrease, or set the volume above 100%:

  ```
  pamixer --set-volume {{105}} --allow-boost
  ```

- Mute the default sink (use `--unmute` instead to unmute):

  ```
  pamixer --mute
  ```

pandoc

Convert documents between various formats.

More information: https://pandoc.org.

- Convert file to pdf (the output format is determined by file extension):

```
pandoc {{input.md}} -o {{output.pdf}}
```

- Force conversion to use a specific format:

```
pandoc {{input.docx}} --to {{gfm}} -o {{output.md}}
```

- Convert to a standalone file with the appropriate headers/footers (for LaTeX, HTML, etc.):

```
pandoc {{input.md}} -s -o {{output.tex}}
```

- List all supported input formats:

```
pandoc --list-input-formats
```

- List all supported output formats:

```
pandoc --list-output-formats
```

parallel-lint

A tool to check the syntax of PHP files in parallel.

More information: https://github.com/JakubOnderka/PHP-Parallel-Lint

- Lint a specific directory:

```
parallel-lint {{path/to/directory}}
```

- Lint a directory using the specified number of parallel processes:

```
parallel-lint -j {{processes}} {{path/to/directory}}
```

- Lint a directory, excluding the specified directory:

```
parallel-lint --exclude {{path/to/excluded_directory}} {{path/to/directory}}
```

- Lint a directory of files using a comma-separated list of extension(s):

```
parallel-lint -e {{php,html,phpt}} {{path/to/directory}}
```

- Lint a directory and output the results as JSON:

```
parallel-lint --json {{path/to/directory}}
```

- Lint a directory and show GitBlame results for rows containing errors:

```
parallel-lint --blame {{path/to/directory}}
```

parallel

Run commands on multiple CPU cores.

More information: https://www.gnu.org/software/parallel.

- Gzip several files at once, using all cores:

```
parallel gzip ::: {{file1}} {{file2}} {{file3}}
```

- Read arguments from stdin, run 4 jobs at once:

```
ls *.txt | parallel -j 4 gzip
```

- Convert JPG images to PNG using replacement strings:

```
parallel convert {} {.}.png ::: *.jpg
```

- Parallel xargs, cram as many args as possible onto one command:

```
{{args}} | parallel -X {{command}}
```

- Break stdin into ~1M blocks, feed each block to stdin of new command:

```
cat {{big_file.txt}} | parallel --pipe --block 1M {{command}}
```

- Run on multiple machines via SSH:

```
parallel -S {{machine1}},{{machine2}} {{command}} ::: {{arg1}} {{arg2}}
```

parq uet-tools

A tool to show, inspect and manipulate P arq uet file.

More information: [https://github.com/apache/parq uet-mr/tree/master/parq uet-tools-deprecated](https://github.com/apache/parquet-mr/tree/master/parquet-tools-deprecated).

- Display the content of a P arq uet file:

  ```
  parq u et- tools cat {{path/to/parq u et}}
  ```

- Display the first few lines of a P arq uet file:

  ```
  parq u et- tools head {{path/to/parq u et}}
  ```

- P rint the schema of a P arq uet file:

  ```
  parq u et- tools schem a {{path/to/parq u et}}
  ```

- P rint the metadata of a P arq uet file:

  ```
  parq u et- tools m eta {{path/to/parq u et}}
  ```

- P rint the content and metadata of a P arq uet file:

  ```
  parq u et- tools du m p {{path/to/parq u et}}
  ```

- Concatenate several P arq uet files into the target one:

  ```
  parq u et- tools m erg e {{path/to/parq u et1 }} {{path/to/parq u et2 }} {{path/to/targ et_parq u et}}
  ```

- P rint the count of rows in a P arq uet file:

  ```
  parq u et- tools row cou nt {{path/to/parq u et}}
  ```

- P rint the column and offset indexes of a P arq uet file:

  ```
  parq u et- tools colu m n- index {{path/to/parq u et}}
  ```

particle

A command-line tool for interacting with Particle devices.

More information: https://docs.particle.io/tutorials/developer-tools/cli.

- Log in or create an account for the Particle CLI:

  ```
  particle setup
  ```

- Display a list of devices:

  ```
  particle list
  ```

- Create a new Particle project interactively:

  ```
  particle project create
  ```

- Compile a Particle project:

  ```
  particle compile {{device_type}} {{path/to/source_code.ino}}
  ```

- Update a device to use a specific app remotely:

  ```
  particle flash {{device_name}} {{path/to/program.bin}}
  ```

- Update a device to use the latest firmware via serial:

  ```
  particle flash --serial {{path/to/firmware.bin}}
  ```

- Execute a function on a device:

  ```
  particle call {{device_name}} {{function_name}}
  {{function_arguments}}
  ```

pass otp

A pass extension for managing one-time-password (O T P) tokens.

More information: https://github.com/tadfisher/pass-otp#_readme

- P rompt for an otpauth U R I token and create a new pass file:

  ```
  pass otp insert {{path/to/pass}}
  ```

- P rompt for an otpauth U R I token and append to an existing pass file:

  ```
  pass otp append {{path/to/pass}}
  ```

- P rint a 2 F A code using the O T P token in a pass file:

  ```
  pass otp {{path/to/pass}}
  ```

- Copy and don' t print a 2 F A code using the O T P token in a pass file:

  ```
  pass otp - - clip {{path/to/pass}}
  ```

- Display a Q R code using the O T P token stored in a pass file:

  ```
  pass otp u ri - - q rcode {{path/to/pass}}
  ```

- P rompt for an O T P secret value specifying issuer and account (at least one must be specified) and append to existing pass file:

  ```
  pass otp append - - secret - - issu er {{issu er_nam e}} - - accou nt
  {{accou nt_nam e}} {{path/to/pass}}
  ```

pass

Tool for storing and reading passwords or other sensitive data.

All data is GPG-encrypted, and managed with a Git repository.

More information: https://www.passwordstore.org.

- Initialize (or re-encrypt) the storage using one or more GPG IDs:

  ```
  pass init {{gpg_id_1}} {{gpg_id_2}}
  ```

- Save a new password and additional information (press Ctrl+D on a new line to complete):

  ```
  pass insert --multiline {{path/to/data}}
  ```

- Edit an entry:

  ```
  pass edit {{path/to/data}}
  ```

- Copy a password (first line of the data file) to the clipboard:

  ```
  pass -c {{path/to/data}}
  ```

- List the whole store tree:

  ```
  pass
  ```

- Generate a new random password with a given length, and copy it to the clipboard:

  ```
  pass generate -c {{path/to/data}} {{num}}
  ```

- Initialize a new Git repository (any changes done by pass will be committed automatically):

  ```
  pass git init
  ```

passwd

P asswd is a tool used to change a user' s password.

- Change the password of the current user interactively:

  ```
  passw d
  ```

- Change the password of a specific user:

  ```
  passw d {{u sernam e}}
  ```

- G et the current status of the user:

  ```
  passw d - S
  ```

- Make the password of the account blank (it will set the named account passwordless):

  ```
  passw d - d
  ```

paste

Merge lines of files.

More information: https://www.gnu.org/software/coreutils/paste.

- Join all the lines into a single line, using TAB as delimiter:

  ```
  paste -s {{file}}
  ```

- Join all the lines into a single line, using the specified delimiter:

  ```
  paste -s -d {{delim iter}} {{file}}
  ```

- Merge two files side by side, each in its column, using TAB as delimiter:

  ```
  paste {{file1 }} {{file2 }}
  ```

- Merge two files side by side, each in its column, using the specified delimiter:

  ```
  paste -d {{delim iter}} {{file1 }} {{file2 }}
  ```

- Merge two files, with lines added alternatively:

  ```
  paste -d '\n' {{file1 }} {{file2 }}
  ```

pastel

Generate, analyze, convert and manipulate colors.

More information: https://github.com/sharkdp/pastel.

- Convert colors from one format to another. Here from RGB to HSL:

  ```
  pastel format {{hsl}} {{ff8000}}
  ```

- Show and analyze colors on the terminal:

  ```
  pastel color "{{rgb(255,50,127)}}"
  ```

- Pick a color from somewhere on the screen:

  ```
  pastel pick
  ```

- Generate a set of N visually distinct colors:

  ```
  pastel distinct {{8}}
  ```

- Get a list of all X11/CSS color names:

  ```
  pastel list
  ```

patch

Patch a file (or files) with a diff file.

Note that diff files should be generated by the `diff` command.

- Apply a patch using a diff file (filenames must be included in the diff file):

  ```
  patch < {{patch.diff}}
  ```

- Apply a patch to a specific file:

  ```
  patch {{path/to/file}} < {{patch.diff}}
  ```

- Patch a file writing the result to a different file:

  ```
  patch {{path/to/input_file}} -o {{path/to/output_file}} <
  {{patch.diff}}
  ```

- Apply a patch to the current directory:

  ```
  patch -p1 < {{patch.diff}}
  ```

- Apply the reverse of a patch:

  ```
  patch -R < {{patch.diff}}
  ```

patchwork

Render an image of a **squarified treemap** graph from a **graphviz** file.

Layouts: **dot**, **neato**, **twopi**, **circo**, **fdp**, **sfdp**, **osage** & **patchwork**

More information: https://graphviz.org/doc/info/command.html

- Render a `png` image with a filename based on the input filename and output format (uppercase -O):

  ```
  patchwork -T {{png}} -O {{path/to/input.gv}}
  ```

- Render a `svg` image with the specified output filename (lowercase -o):

  ```
  patchwork -T {{svg}} -o {{path/to/image.svg}} {{path/to/input.gv}}
  ```

- Render the output in `ps`, `pdf`, `svgz`, `fig`, `png`, `gif`, `jpg`, `json`, or `dot` format:

  ```
  patchwork -T {{format}} -O {{path/to/input.gv}}
  ```

- Render a `gif` image using stdin and stdout:

  ```
  echo "{{digraph {this -> that}}}" | patchwork -T {{gif}} > {{path/to/image.gif}}
  ```

- Display help:

  ```
  patchwork -?
  ```

pathchk

Check the validity and portability of one or more pathnames.

More information: https://www.gnu.org/software/coreutils/pathchk.

- Check pathames for validity in the current system:

```
pathchk {{path1 path2 … }}
```

- Check pathnames for validity on a wider range of P O S IX compliant systems:

```
pathchk -p {{path1 path2 … }}
```

- Check pathnames for validity on all P O S IX compliant systems:

```
pathchk --portability {{path1 path2 … }}
```

- O nly check for empty pathnames or leading dashes (-):

```
pathchk -P {{path1 path2 … }}
```

pax

Archiving and copying utility.

- List the contents of an archive:

```
pax -f {{archive.tar}}
```

- List the contents of a gzipped archive:

```
pax -zf {{archive.tar.gz}}
```

- Create an archive from files:

```
pax -wf {{target.tar}} {{path/to/file1}} {{path/to/file2}} {{path/to/file3}}
```

- Create an archive from files, using output redirection:

```
pax -w {{path/to/file1}} {{path/to/file2}} {{path/to/file3}} > {{target.tar}}
```

- Extract an archive into the current directory:

```
pax -rf {{source.tar}}
```

- Copy to a directory, while keeping the original metadata; `target/` must exist:

```
pax -rw {{path/to/file1}} {{path/to/directory1}} {{path/to/directory2}} {{target/}}
```

pdffonts

Portable Document Format (PDF) file fonts information viewer.

More information: https://www.xpdfreader.com/pdffonts-man.html.

- Print PDF file fonts information:

  ```
  pdffonts {{path/to/file.pdf}}
  ```

- Specify user password for PDF file to bypass security restrictions:

  ```
  pdffonts -upw {{password}} {{path/to/file.pdf}}
  ```

- Specify owner password for PDF file to bypass security restrictions:

  ```
  pdffonts -opw {{password}} {{path/to/file.pdf}}
  ```

- Print additional information on location of the font that will be used when the PDF file is rasterized:

  ```
  pdffonts -loc {{path/to/file.pdf}}
  ```

- Print additional information on location of the font that will be used when the PDF file is converted to PostScript:

  ```
  pdffonts -locPS {{path/to/file.pdf}}
  ```

pdfimages

Utility for extracting images from PDFs.

- Extract all images from a PDF file and save them as PNGs:

  ```
  pdfimages -png {{path/to/file.pdf}} {{filename_prefix}}
  ```

- Extract images from pages 3 to 5:

  ```
  pdfimages -f {{3}} -l {{5}} {{path/to/file.pdf}} {{filename_prefix}}
  ```

- Extract images from a PDF file and include the page number in the output filenames:

  ```
  pdfimages -p {{path/to/file.pdf}} {{filename_prefix}}
  ```

- List information about all the images in a PDF file:

  ```
  pdfimages -list {{path/to/file.pdf}}
  ```

pdfinfo

Portable Document Format (PDF) file information viewer.

More information: https://www.xpdfreader.com/pdfinfo-man.html.

- Print PDF file information:

  ```
  pdfinfo {{path/to/file.pdf}}
  ```

- Specify user password for PDF file to bypass security restrictions:

  ```
  pdfinfo -upw {{password}} {{path/to/file.pdf}}
  ```

- Specify owner password for PDF file to bypass security restrictions:

  ```
  pdfinfo -opw {{password}} {{path/to/file.pdf}}
  ```

pdfj am

S hell frontend for the LaT eX pdfpages package for mingling P DF s.

More information: https://github.com/rrthomas/pdfj am

- Merge two (or more) P DF s:

```
pdfj am {{path/to/file1 . pdf}} {{path/to/file2 . pdf}} - - ou tfile
{{path/to/ou tpu t_file. pdf}}
```

- Merge the first page of each file together:

```
pdfj am {{files. . . }} 1 - - ou tfile {{path/to/ou tpu t_file. pdf}}
```

- Merge subranges from two P DF s:

```
pdfj am {{path/to/file1 . pdf 3 - 5 , 1 }} {{path/to/file2 . pdf 4 - 6 }}
- - ou tfile {{path/to/ou tpu t_file. pdf}}
```

- S ign an A4 page (adj ust delta to height for other formats) with a scanned signature by overlaying them:

```
pdfj am {{path/to/file. pdf}} {{path/to/sig natu re}} - - fitpaper
tru e - - ou tfile {{path/to/sig ned. pdf}} - - nu p " {{1 x2 }}" - - delt
" {{0 - 8 4 2 pt}}"
```

- Arrange the pages from the input file into a fancy 2 x2 grid:

```
pdfj am {{path/to/file. pdf}} - - nu p {{2 x2 }} - - su ffix {{4 u p}} - -
pream b le ' {{\ u sepack ag e{fancyhdr} \ pag estyle{fancy}}}'
```

- R everse the order of pages within each given file and concatenate them:

```
pdfj am {{files. . . }} {{last- 1 }} - - su ffix {{rev ersed}}
```

pdfj oin

P DF merging utility based on pdfj am.

More information: https://github.com/rrthomas/pdfj am-extras

- Merge two P DF s into one with the default suffix " j oined" :

```
pdfj oin {{path/to/file1 . pdf}} {{path/to/file2 . pdf}}
```

- Merge the first page of each given file together:

```
pdfj oin {{path/to/file1 . pdf path/to/file2 . pdf . . . }} {{1 }} - -
ou tfile {{ou tpu t_file}}
```

- S ave pages 3 to 5 followed by page 1 to a new P DF with custom suffix:

```
pdfj oin {{path/to/file. pdf}} {{3 - 5 , 1 }} - - su ffix
{{rearrang ed}}
```

- Merge page subranges from two P DF s:

```
pdfj oin {/path/to/file1 . pdf}} {{2 - }} {{file2 }} {{last- 3 }} - -
ou tfile {{ou tpu t_file}}
```

pdflatex

Compile a PDF document from LaTeX source files.

More information: https://manned.org/pdflatex

- Compile a PDF document:

```
pdflatex {{source.tex}}
```

- Compile a PDF document specifying an output directory:

```
pdflatex -output-directory={{path/to/directory}}
{{source.tex}}
```

- Compile a PDF document, halting on each error:

```
pdflatex -halt-on-error {{source.tex}}
```

pdfposter

Convert a large-sheeted pdf into multiple A4 pages for printing.

More information: https://pdfposter.readthedocs.io.

- Convert an A2 poster into 4 A4 pages:

  ```
  pdfposter - - poster- size a2 {{inpu t_file. pdf}}
  {{ou tpu t_file. pdf}}
  ```

- S cale an A4 poster to A3 and then generate 2 A4 pages:

  ```
  pdfposter - - scale 2 {{inpu t_file. pdf}} {{ou tpu t_file. pdf}}
  ```

pdfseparate

Portable Document Format (PDF) file page extractor.

More information: https://manpages.debian.org/unstable/poppler-utils/pdfseparate.1.en.html

- Extract pages from PDF file and make a separate PDF file for each page:

```
pdfseparate {{path/to/source_filename.pdf}} {{path/to/destination_filename-%d.pdf}}
```

- Specify the first/start page for extraction:

```
pdfseparate -f {{3}} {{path/to/source_filename.pdf}} {{path/to/destination_filename-%d.pdf}}
```

- Specify the last page for extraction:

```
pdfseparate -l {{10}} {{path/to/source_filename.pdf}} {{path/to/destination_filename-%d.pdf}}
```

pdftex

Compile a PDF document from TeX source files.

More information: https://www.tug.org/applications/pdftex/.

- Compile a PDF document:

```
pdftex {{source.tex}}
```

- Compile a PDF document, specifying an output directory:

```
pdftex -output-directory={{path/to/directory}} {{source.tex}}
```

- Compile a PDF document, halting on each error:

```
pdftex -halt-on-error {{source.tex}}
```

pdftk

PDF toolkit.

More information: https://www.pdflabs.com/tools/pdftk-the-pdf-toolkit

- Extract pages 1-3, 5 and 6-10 from a PDF file and save them as another one:

```
pdftk {{input.pdf}} cat {{1-3 5 6-10}} output {{output.pdf}}
```

- Merge (concatenate) a list of PDF files and save the result as another one:

```
pdftk {{file1.pdf file2.pdf ...}} cat output {{output.pdf}}
```

- Split each page of a PDF file into a separate file, with a given filename output pattern:

```
pdftk {{input.pdf}} burst output {{out_%d.pdf}}
```

- Rotate all pages by 180 degrees clockwise:

```
pdftk {{input.pdf}} cat {{1-endsouth}} output {{output.pdf}}
```

- Rotate third page by 90 degrees clockwise and leave others unchanged:

```
pdftk {{input.pdf}} cat {{1-2 3east 4-end}} output
{{output.pdf}}
```

pdftocairo

Converts PDF files to PNG/JPEG/TIFF/PDF/PS/EPS/SVG using cairo.

More information: https://poppler.freedesktop.org.

- Convert a PDF file to JPEG:

```
pdftocairo {{path/to/file.pdf}} -jpeg
```

- Convert to PDF expanding the output to fill the paper:

```
pdftocairo {{path/to/file.pdf}} {{output.pdf}} -pdf -expand
```

- Convert to SVG specifying the first/last page to convert:

```
pdftocairo {{path/to/file.pdf}} {{output.svg}} -svg -f {{first_page}} -l {{last_page}}
```

- Convert to PNG with 200 ppi resolution:

```
pdftocairo {{path/to/file.pdf}} {{output.png}} -png -r 200
```

- Convert to grayscale TIFF setting paper size to A3:

```
pdftocairo {{path/to/file.pdf}} -tiff -gray -paper A3
```

- Convert to PNG cropping x and y pixels from the top left corner:

```
pdftocairo {{path/to/file.pdf}} -png -x {{x_pixels}} -y {{y_pixels}}
```

pdftotext

Convert PDF files to plain text format.

- Convert `filename.pdf` to plain text and print it to standard output:

  ```
  pdftotext {{filename.pdf}} -
  ```

- Convert `filename.pdf` to plain text and save it as `filename.txt`:

  ```
  pdftotext {{filename.pdf}}
  ```

- Convert `filename.pdf` to plain text and preserve the layout:

  ```
  pdftotext -layout {{filename.pdf}}
  ```

- Convert `input.pdf` to plain text and save it as `output.txt`:

  ```
  pdftotext {{input.pdf}} {{output.txt}}
  ```

- Convert pages 2, 3, and 4 of `input.pdf` to plain text and save them as `output.txt`:

  ```
  pdftotext -f {{2}} -l {{4}} {{input.pdf}} {{output.txt}}
  ```

pdfunite

PDF merging utility.

More information: https://github.com/mtgrosser/pdfunite.

- Merge 2 PDFs into a single PDF:

```
pdfunite {{path/to/fileA.pdf}} {{path/to/fileB.pdf}} {{path/to/merged_output.pdf}}
```

- Merge a directory of PDFs into a single PDF:

```
pdfunite {{path/to/directory/*.pdf}} {{path/to/merged_output.pdf}}
```

peco

Interactive filtering tool.

More information: https://github.com/peco/peco.

- Start peco on all files in the specified directory:

  ```
  find {{path/to/directory}} -type f | peco
  ```

- Start peco for running processes:

  ```
  ps aux | peco
  ```

- Start peco with a specified query:

  ```
  peco --query "{{query}}"
  ```

peerflix

Stream video- or audio-based torrents to a media player.

More information: https://github.com/mafintosh/peerflix

- Stream the largest media file in a torrent:

  ```
  peerflix "{{torrent_url|magnet_link}}"
  ```

- List all streamable files contained in a torrent (given as a magnet link):

  ```
  peerflix "{{magnet:?xt=urn:btih:
  0123456789abcdef0123456789abcdef0123456789}}" --li
  ```

- Stream the largest file in a torrent, given as a torrent URL, to VLC:

  ```
  peerflix "{{http://example.net/music.torrent}}" --vlc
  ```

- Stream the largest file in a torrent to MPlayer, with subtitles:

  ```
  peerflix "{{torrent_url|magnet_link}}" --mplayer --subtitles
  {{subtitle-file.srt}}
  ```

- Stream all files from a torrent to Airplay:

  ```
  peerflix "{{torrent_url|magnet_link}}" --all --airplay
  ```

peludna-prognoza

Fetch pollen measurement data for Croatian cities from your terminal using Pliva's allergies data API.

More information: https://github.com/vladimyr/peludna-prognoza

- Start an interactive search for a city and fetch data for it:

  ```
  peludna-prognoza
  ```

- Fetch data for a city:

  ```
  peludna-prognoza "{{city}}"
  ```

- Display data in a machine-readable format:

  ```
  peludna-prognoza "{{city}}" --{{json|xml}}
  ```

- Display the pollen measurement page for a city at https://plivazdravljenje.hr in the default web browser:

  ```
  peludna-prognoza "{{city}}" --web
  ```

perl

The Perl 5 language interpreter.

More information: https://www.perl.org.

- Parse and execute a Perl script:

```
perl {{script.pl}}
```

- Check syntax errors on a Perl script:

```
perl -c {{script.pl}}
```

- Parse and execute a Perl statement:

```
perl -e {{perl_statement}}
```

- Run a Perl script in debug mode, using `perldebug`:

```
perl -d {{script.pl}}
```

- Loop over all lines of a file, editing them in-place using a find/replace expression:

```
perl -p -i -e 's/{{find}}/{{replace}}/g' {{filename}}
```

- Run a find/replace expression on a file, saving the original file with a given extension:

```
perl -p -i'.old' -e 's/{{find}}/{{replace}}/g' {{filename}}
```

- Run a multiline find/replace expression on a file, and save the result in another file:

```
perl -p0e 's/{{foo\nbar}}/{{foobar}}/g' {{input_file}} > {{output_file}}
```

- Run a regular expression on stdin, printing out the first capture group for each line:

```
cat {{path/to/input_file}} | perl -nle 'if (/.*({{foo}}).*/) {print "$1"; last;}'
```

pest

A P H P testing framework with a focus on simplicity.

More information: https://pestphp.com.

- Initialise a standard P est configuration in the current directory:

  ```
  pest - - init
  ```

- R un tests in the current directory:

  ```
  pest
  ```

- R un tests annotated with the given group:

  ```
  pest - - g rou p {{nam e}}
  ```

- R un tests and print the coverage report to stdout:

  ```
  pest - - cov erag e
  ```

- R un tests with coverage and fail if the coverage is less than the minimum percentage:

  ```
  pest - - cov erag e - - m in= {{8 0 }}
  ```

pg_ctl

Utility for controlling a PostgreSQL server and database cluster.

More information: https://www.postgresql.org/docs/current/app-pg-ctl.html

- Initialize a new PostgreSQL database cluster:

  ```
  pg_ctl -D {{data_directory}} init
  ```

- Start a PostgreSQL server:

  ```
  pg_ctl -D {{data_directory}} start
  ```

- Stop a PostgreSQL server:

  ```
  pg_ctl -D {{data_directory}} stop
  ```

- Restart a PostgreSQL server:

  ```
  pg_ctl -D {{data_directory}} restart
  ```

- Reload the PostgreSQL server configuration:

  ```
  pg_ctl -D {{data_directory}} reload
  ```

pg_dump

Extract a PostgreSQL database into a script file or other archive file.

More information: https://www.postgresql.org/docs/current/app-pgdump.html

- Dump database into a SQL-script file:

```
pg_dump {{database}} > {{output_file.sql}}
```

- Same as above, customize username:

```
pg_dump -U {{username}} {{database}} > {{output_file.sql}}
```

- Same as above, customize host and port:

```
pg_dump -h {{host}} -p {{port}} {{database}} >
{{output_file.sql}}
```

- Dump a database into a custom-format archive file:

```
pg_dump -Fc {{database}} > {{output_file.dump}}
```

- Dump only database data into an SQL-script file:

```
pg_dump -a {{database}} > {{path/to/output_file.sql}}
```

- Dump only schema (data definitions) into an SQL-script file:

```
pg_dump -s {{database}} > {{path/to/output_file.sql}}
```

pg_restore

Restore a PostgreSQL database from an archive file created by pg_dump.

More information: https://www.postgresql.org/docs/current/app-pgrestore.html

- Restore an archive into an existing database:

```
pg_restore -d {{database}} {{archive_file.dump}}
```

- Same as above, customize username:

```
pg_restore -U {{username}} -d {{database}}
{{archive_file.dump}}
```

- Same as above, customize host and port:

```
pg_restore -h {{host}} -p {{port}} -d {{database}}
{{archive_file.dump}}
```

- List database objects included in the archive:

```
pg_restore --list {{archive_file.dump}}
```

- Clean database objects before creating them:

```
pg_restore --clean -d {{database}} {{archive_file.dump}}
```

- Use multiple jobs to do the restoring:

```
pg_restore -j {{2}} -d {{database}} {{archive_file.dump}}
```

pgbench

Run a benchmark test on PostgreSQL.

More information: https://www.postgresql.org/docs/10/pgbench.html

- Initialize a database with a scale factor of 50 times the default size:

  ```
  pgbench --initialize --scale={{50}} {{database_name}}
  ```

- Benchmark a database with 10 clients, 2 worker threads, and 10,000 transactions per client:

  ```
  pgbench --client={{10}} --jobs={{2}} --transactions={{10000}} {{database_name}}
  ```

pgrep

Find or signal processes by name.

More information: https://www.man7.org/linux/man-pages/man1/pkill.1.html

- Return PIDs of any running processes with a matching command string:

```
pgrep {{process_name}}
```

- Search for processes including their command-line options:

```
pgrep --full "{{process_name}} {{parameter}}"
```

- Search for processes run by a specific user:

```
pgrep --euid root {{process_name}}
```

phan

A static analysis tool for PHP.

More information: https://github.com/phan/phan.

- Generate a `.phan/config.php` in the current directory:

  ```
  phan --init
  ```

- Generate a Phan configuration file using a specific level (1 being strictest to 5 being the least strict):

  ```
  phan --init --init-level {{level}}
  ```

- Analyse the current directory:

  ```
  phan
  ```

- Analyse one or more directories:

  ```
  phan --directory {{path/to/directory}} --directory {{path/to/another_directory}}
  ```

- Specify a config file (defaults to `.phan/config.php`):

  ```
  phan --config-file {{path/to/config.php}}
  ```

- Specify the output mode:

  ```
  phan --output-mode {{text|verbose|json|csv|codeclimate|checkstyle|pylint|html}}
  ```

- Specify the number of parallel processes:

  ```
  phan --processes {{number_of_processes}}
  ```

phing

A P H P build tool based on Apache Ant.

More information: https://www.phing.info.

- P erform the default task in the `b u ild. xml` file:

  ```
  phing
  ```

- Initialise a new build file:

  ```
  phing - i {{path/to/b u ild. xm l}}
  ```

- P erform a specific task:

  ```
  phing {{task _nam e}}
  ```

- S pecify a custom build file path:

  ```
  phing - f {{path/to/b u ild. xm l}} {{task _nam e}}
  ```

- S pecify a log file to output to:

  ```
  phing - b {{path/to/log _file}} {{task _nam e}}
  ```

- S pecify custom properties to use in the build:

  ```
  phing - D {{property}}= {{v alu e}} {{task _nam e}}
  ```

- S pecify a custom listener class:

  ```
  phing - listener {{class_nam e}} {{task _nam e}}
  ```

- B uild using verbose output:

  ```
  phing - v erb ose {{task _nam e}}
  ```

phive

The PHAR Installation and Verification Environment for secure PHP application deployment.

More information: https://phar.io.

- Display a list of available aliased PHARs:

  ```
  phive list
  ```

- Install a specified PHAR to the local directory:

  ```
  phive install {{alias|url}}
  ```

- Install a specified PHAR globally:

  ```
  phive install {{alias|url}} --global
  ```

- Install a specified PHAR to a target directory:

  ```
  phive install {{alias|url}} --target {{path/to/directory}}
  ```

- Update all PHAR files to the latest version:

  ```
  phive update
  ```

- Remove a specified PHAR file:

  ```
  phive remove {{alias|url}}
  ```

- Remove unused PHAR files:

  ```
  phive purge
  ```

- List all available commands:

  ```
  phive help
  ```

php artisan

Laravel's Artisan command-line interface.

More information: https://laravel.com/docs/artisan.

- Start PHP's built-in web server for the current Laravel application:

  ```
  php artisan serve
  ```

- Start an interactive PHP command-line interface:

  ```
  php artisan tinker
  ```

- Generate a new Eloquent model class with a migration, factory and resource controller:

  ```
  php artisan make:model {{ModelName}} --all
  ```

- Display a list of all available commands:

  ```
  php artisan help
  ```

php-coveralls

A P H P client for Coveralls.

More information: https://php-coveralls.github.io/php-coveralls.

- S end coverage information to Coveralls:

  ```
  php- cov eralls
  ```

- S end coverage information to Coveralls for a specific directory:

  ```
  php- cov eralls - - root_dir {{path/to/directory}}
  ```

- S end coverage information to Coveralls with a specific config:

  ```
  php- cov eralls - - config {{path/to/. cov eralls. ym l}}
  ```

- S end coverage information to Coveralls with verbose output:

  ```
  php- cov eralls - - v erb ose
  ```

- S end coverage information to Coveralls excluding source files with no executable statements:

  ```
  php- cov eralls - - exclu de- no- stm t
  ```

- S end coverage information to Coveralls with a specific environment name:

  ```
  php- cov eralls - - env {{test| dev | prod}}
  ```

- S pecify multiple Coverage Clover X ML files to upload:

  ```
  php- cov eralls - - cov erag e_clov er {{path/to/first_clov er. xm l}}
  - - cov erag e_clov er {{path/to/second_clov er. xm l}}
  ```

- O utput the J S O N that will be sent to Coveralls to a specific file:

  ```
  php- cov eralls - - j son_path {{path/to/cov eralls- u pload. j son}}
  ```

php yii

Yii Framework's command-line interface.

More information: https://yiiframework.com.

- Display a list of all available commands:

  ```
  php yii {{help}}
  ```

- Start PHP's built-in web server for the current Yii application:

  ```
  php yii {{serve}}
  ```

- Generate a controller, views and related files for the CRUD actions on the specified model class:

  ```
  php yii {{gii/crud}} --modelClass={{ModelName}} --controllerClass={{ControllerName}}
  ```

php

PHP command-line interface.

More information: https://php.net.

- Parse and execute a php script:

  ```
  php {{file}}
  ```

- Check syntax on (i.e. lint) a PHP script:

  ```
  php -l {{file}}
  ```

- Run PHP interactively:

  ```
  php -a
  ```

- Run PHP code (Notes: Don't use <??> tags; escape double quotes with backslash):

  ```
  php -r "{{code}}"
  ```

- Start a PHP built-in web server in the current directory:

  ```
  php -S {{host:port}}
  ```

- Get a list of installed PHP extensions:

  ```
  php -m
  ```

- Display information about the current PHP configuration:

  ```
  php -i
  ```

phpbu

A backup utility framework for PHP.

More information: https://phpbu.de.

- Run backups using the default `phpbu.xml` configuration file:

  ```
  phpbu
  ```

- Run backups using a specific configuration file:

  ```
  phpbu --configuration={{path/to/configuration_file.xml}}
  ```

- Only run the specified backups:

  ```
  phpbu --limit={{backup_task_name}}
  ```

- Simulate the actions that would have been performed:

  ```
  phpbu --simulate
  ```

phpcbf

Fix violations detected by phpcs.

More information: https://github.com/squiz/labs/PHP_CodeSniffer

- Fix issues in the specified directory (defaults to the PEAR standard):

  ```
  phpcbf {{path/to/directory}}
  ```

- Display a list of installed coding standards:

  ```
  phpcbf -i
  ```

- Specify a coding standard to validate against:

  ```
  phpcbf {{path/to/directory}} --standard {{standard}}
  ```

- Specify comma-separated file extensions to include when sniffing:

  ```
  phpcbf {{path/to/directory}} --extensions
  {{file_extension(s)}}
  ```

- A comma-separated list of files to load before processing:

  ```
  phpcbf {{path/to/directory}} --bootstrap {{file(s)}}
  ```

- Don't recurse into subdirectories:

  ```
  phpcbf {{path/to/directory}} -l
  ```

phpcpd

A copy and paste detector for PHP code.

More information: https://github.com/sebastianbergmann/phpcpd.

- Analyse duplicated code for a specific file or directory:

  ```
  phpcpd {{path/to/file_or_directory}}
  ```

- Analyse using fuzzy matching for variable names:

  ```
  phpcpd --fuzzy {{path/to/file_or_directory}}
  ```

- Specify a minimum number of identical lines (defaults to 5):

  ```
  phpcpd --min-lines {{number_of_lines}} {{path/to/file_or_directory}}
  ```

- Specify a minimum number of identical tokens (defaults to 70):

  ```
  phpcpd --min-tokens {{number_of_tokens}} {{path/to/file_or_directory}}
  ```

- Exclude a directory from analysis (must be relative to the source):

  ```
  phpcpd --exclude {{path/to/excluded_directory}} {{path/to/file_or_directory}}
  ```

- Output the results to a PHP-CPD XML file:

  ```
  phpcpd --log-pmd {{path/to/log_file}} {{path/to/file_or_directory}}
  ```

phpcs

Tokenize PHP, JavaScript and CSS files to detect violations of a defined set of coding standards.

More information: https://github.com/squizlabs/PHP_CodeSniffer

- Sniff the specified directory for issues (defaults to the PEAR standard):

```
phpcs {{path/to/directory}}
```

- Display a list of installed coding standards:

```
phpcs -i
```

- Specify a coding standard to validate against:

```
phpcs {{path/to/directory}} --standard {{standard}}
```

- Specify comma-separated file extensions to include when sniffing:

```
phpcs {{path/to/directory}} --extensions
{{file_extension(s)}}
```

- Specify the format of the output report (e.g.: `full`, `xml`, `json`, `summary`):

```
phpcs {{path/to/directory}} --report {{format}}
```

- Set config variables to be used during the process:

```
phpcs {{path/to/directory}} --config-set {{key}} {{value}}
```

- A comma-separated list of files to load before processing:

```
phpcs {{path/to/directory}} --bootstrap {{file(s)}}
```

- Don't recurse into subdirectories:

```
phpcs {{path/to/directory}} -l
```

phpdox

A P H P documentation generator.

More information: https://phpdox.net.

- Display an annotated skeleton configuration X ML file:

  `phpdox - - sk el`

- G enerate documentation for the current working directory:

  `phpdox`

- G enerate documentation using a specific configuration file:

  `phpdox - - file {{path/to/phpdox. xm l}}`

- O nly run the metadata collection process:

  `phpdox - - collector`

- O nly run the documentation generator process:

  `phpdox - - g enerator`

phpenv

A PHP version manager for development purposes.

More information: https://github.com/phpenv/phpenv.

- Install a PHP version globally:

  `phpenv install {{version}}`

- Refresh shim files for all PHP binaries known to `phpenv`:

  `phpenv rehash`

- List all installed PHP versions:

  `phpenv versions`

- Display the currently active PHP version:

  `phpenv version`

- Set the global PHP version:

  `phpenv global {{version}}`

- Set the local PHP version, which overrides the global version:

  `phpenv local {{version}}`

- Unset the local PHP version:

  `phpenv local --unset`

phpize

Prepare a PHP extension for compiling.

More information: https://www.php.net/manual/install.pecl.phpize

- Prepare the PHP extension in the current directory for compiling:

  ```
  phpize
  ```

- Delete files previously created by phpize:

  ```
  phpize --clean
  ```

phploc

A tool for quickly measuring the size and analyzing the structure of a PHP project.

More information: https://github.com/sebastianbergmann/phploc.

- Analyse a directory and print the result:

  ```
  phploc {{path/to/directory}}
  ```

- Include only specific files from a comma-separated list (globs are allowed):

  ```
  phploc {{path/to/directory}} --names {{files}}
  ```

- Exclude specific files from a comma-separated list (globs are allowed):

  ```
  phploc {{path/to/directory}} --names-exclude {{files}}
  ```

- Exclude a specific directory from analysis:

  ```
  phploc {{path/to/directory}} --exclude {{path/to/
  exclude_directory}}
  ```

- Log the results to a specific CSV file:

  ```
  phploc {{path/to/directory}} --log-csv {{path/to/file}}
  ```

- Log the results to a specific XML file:

  ```
  phploc {{path/to/directory}} --log-xml {{path/to/file}}
  ```

- Count PHPUnit test case classes and test methods:

  ```
  phploc {{path/to/directory}} --count-tests
  ```

phpmd

A PHP mess detector that checks for common potential problems.

More information: https://github.com/phpmd/phpmd.

- Display a list of available rulesets and formats:

```
phpmd
```

- Scan a file or directory for problems using comma-separated rulesets:

```
phpmd {{path/to/file_or_directory}} {{xml|text|html}} {{rulesets}}
```

- Specify the minimum priority threshold for rules:

```
phpmd {{path/to/file_or_directory}} {{xml|text|html}} {{rulesets}} --minimumpriority {{priority}}
```

- Include only the specified extensions in analysis:

```
phpmd {{path/to/file_or_directory}} {{xml|text|html}} {{rulesets}} --suffixes {{extensions}}
```

- Exclude the specified comma-separated directories:

```
phpmd {{path/to/file_or_directory}} {{xml|text|html}} {{rulesets}} --exclude {{directory_patterns}}
```

- Output the results to a file instead of stdout:

```
phpmd {{path/to/file_or_directory}} {{xml|text|html}} {{rulesets}} --reportfile {{path/to/report_file}}
```

- Ignore the use of warning-suppressive PHPDoc comments:

```
phpmd {{path/to/file_or_directory}} {{xml|text|html}} {{rulesets}} --strict
```

phpspec

A B ehaviour Driven Development tool for P H P .

More information: https://phpspec.net.

- Create a specification for a class:

  ```
  phpspec describ e {{class_nam e}}
  ```

- R un all specifications in the " spec" directory:

  ```
  phpspec ru n
  ```

- R un a single specification:

  ```
  phpspec ru n {{path/to/class_specification_file}}
  ```

- R un specifications using a specific configuration file:

  ```
  phpspec ru n - c {{path/to/config u ration_file}}
  ```

- R un specifications using a specific bootstrap file:

  ```
  phpspec ru n - b {{path/to/b ootstrap_file}}
  ```

- Disable code generation prompts:

  ```
  phpspec ru n - - no- code- g eneration
  ```

- E nable fake return values:

  ```
  phpspec ru n - - fak e
  ```

phpstan

A P H P static analysis tool to discover bugs in code.

More information: https://github.com/phpstan/phpstan.

- Display available options for analysis:

  ```
  phpstan analyse - - help
  ```

- Analyse the specified space-separated directories:

  ```
  phpstan analyse {{path/to/directory}}
  ```

- Analyse a directory using a configuration file:

  ```
  phpstan analyse {{path/to/directory}} - - config u ration {{path/
  to/config }}
  ```

- Analyse using a specific rule level (0 -7 , higher is stricter):

  ```
  phpstan analyse {{path/to/directory}} - - lev el {{lev el}}
  ```

- S pecify an autoload file to load before analysing:

  ```
  phpstan analyse {{path/to/directory}} - - au toload- file {{path/
  to/au toload_file}}
  ```

- S pecify a memory limit during analysis:

  ```
  phpstan analyse {{path/to/directory}} - - m em ory- lim it
  {{m em ory_lim it}}
  ```

phpstorm

A cross-platform IDE for PHP based on the JetBrains IntelliJ platform.

More information: https://jetbrains.com/phpstorm

- Open a specific directory:

  ```
  phpstorm {{path/to/directory}}
  ```

- Open a file:

  ```
  phpstorm {{path/to/file}}
  ```

- Open a file at a specific line:

  ```
  phpstorm --line {{line_number}} {{path/to/file}}
  ```

- View the differences between two files:

  ```
  phpstorm diff {{path/to/left_file}} {{path/to/right_file}}
  ```

phpunit

PHPUnit command-line test runner.

More information: https://phpunit.de.

- Run tests in the current directory. Note: Expects you to have a 'phpunit.xml':

  ```
  phpunit
  ```

- Run tests in a specific file:

  ```
  phpunit {{path/to/TestFile.php}}
  ```

- Run tests annotated with the given group:

  ```
  phpunit --group {{name}}
  ```

- Run tests and generate a coverage report in HTML:

  ```
  phpunit --coverage-html {{directory}}
  ```

piactl

The command-line tool for Private Internet Access, a commercial VPN provider.

More information: https://privateinternetaccess.com/helpdesk/kb/articles/pia-desktop-command-line-interface.

- Log in to Private Internet Access:

  ```
  piactl login {{path/to/login_file}}
  ```

- Connect to Private Internet Access:

  ```
  piactl connect
  ```

- Disconnect from Private Internet Access:

  ```
  piactl disconnect
  ```

- Enable or disable the Private Internet Access daemon in the background:

  ```
  piactl background {{enable|disable}}
  ```

- List all available VPN regions:

  ```
  piactl get regions
  ```

- Display the current VPN region:

  ```
  piactl get region
  ```

- Set your VPN region:

  ```
  piactl set region {{region}}
  ```

- Log out of Private Internet Access:

  ```
  piactl logout
  ```

picard

Next generation MusicBrainz tagging application.

More information: https://picard.musicbrainz.org/

- Start Picard:

  ```
  picard
  ```

- Open a set of files:

  ```
  picard {{path/to/file1.mp3}} {{path/to/file2.mp3}}
  ```

- Display the version of Picard installed:

  ```
  picard --longversion
  ```

pickle

A P H P extension installer based on Composer.

More information: https://github.com/F riendsO fP H P /pickle

- Install a specific P H P extension:

  ```
  pick le install {{extension_nam e}}
  ```

- Convert an existing P E CL extension configuration to a P ickle configuration file:

  ```
  pick le conv ert {{path/to/directory}}
  ```

- V alidate a P E CL extension:

  ```
  pick le v alidate {{path/to/directory}}
  ```

- P ackage a P E CL extension for release:

  ```
  pick le release {{path/to/directory}}
  ```

pigz

Multithreaded zlib compression utility.

More information: https://github.com/madler/pigz.

- Compress a file with default options:

```
pigz {{filename}}
```

- Compress a file using the best compression method:

```
pigz -9 {{filename}}
```

- Compress a file using no compression and 4 processors:

```
pigz -0 -p{{4}} {{filename}}
```

- Compress a directory using tar:

```
tar cf - {{path/to/directory}} | pigz > {{filename}}.tar.gz
```

- Decompress a file:

```
pigz -d {{archive.gz}}
```

- List the contents of an archive:

```
pigz -l {{archive.tar.gz}}
```

ping

Send ICMP ECHO_REQUEST packets to network hosts.

- Ping host:

```
ping {{host}}
```

- Ping a host only a specific number of times:

```
ping -c {{count}} {{host}}
```

- Ping host, specifying the interval in seconds between requests (default is 1 second):

```
ping -i {{seconds}} {{host}}
```

- Ping host without trying to lookup symbolic names for addresses:

```
ping -n {{host}}
```

- Ping host and ring the bell when a packet is received (if your terminal supports it):

```
ping -a {{host}}
```

- Also display a message if no response was received:

```
ping -O {{host}}
```

ping6

Send ICMP ECHO_REQUEST packets to network hosts via IPv6 address.

- Ping a host:

  ```
  ping6 {{host}}
  ```

- Ping a host only a specific number of times:

  ```
  ping6 -c {{count}} {{host}}
  ```

- Ping a host, specifying the interval in seconds between requests (default is 1 second):

  ```
  ping6 -i {{seconds}} {{host}}
  ```

- Ping a host without trying to lookup symbolic names for addresses:

  ```
  ping6 -n {{host}}
  ```

- Ping a host and ring the bell when a packet is received (if your terminal supports it):

  ```
  ping6 -a {{host}}
  ```

pinky

Print user information using the `finger` protocol.

- Display details about the current user:

  ```
  pinky
  ```

- Display details for a specific user:

  ```
  pinky {{user}}
  ```

- Display details in the long format:

  ```
  pinky {{user}} -l
  ```

- Omit the user's home directory and shell in long format:

  ```
  pinky {{user}} -lb
  ```

- Omit the user's project file in long format:

  ```
  pinky {{user}} -lh
  ```

- Omit the column headings in short format:

  ```
  pinky {{user}} -f
  ```

pio access

Set the access level on published resources (packages) in the registry.

More information: https://docs.platformio.org/en/latest/core/userguide/access/.

- Grant a user access to a resource:

  ```
  pio access grant {{guest|maintainer|admin}} {{username}} {{resource_urn}}
  ```

- Remove a user's access to a resource:

  ```
  pio access revoke {{username}} {{resource_urn}}
  ```

- Show all resources that a user or team has access to and the access level:

  ```
  pio access list {{username}}
  ```

- Restrict access to a resource to specific users or team members:

  ```
  pio access private {{resource_urn}}
  ```

- Allow all users access to a resource:

  ```
  pio access public {{resource_urn}}
  ```

pio account

Manage your PlatformIO account in the command-line.

More information: https://docs.platformio.org/en/latest/core/userguide/account/.

- Register a new PlatformIO account:

  ```
  pio account register --username {{username}} --email {{email}} --password {{password}} --firstname {{firstname}} --lastname {{lastname}}
  ```

- Permanently delete your PlatformIO account and related data:

  ```
  pio account destroy
  ```

- Log in to your PlatformIO account:

  ```
  pio account login --username {{username}} --password {{password}}
  ```

- Log out of your PlatformIO account:

  ```
  pio account logout
  ```

- Update your PlatformIO profile:

  ```
  pio account update --username {{username}} --email {{email}} --firstname {{firstname}} --lastname {{lastname}} --current-password {{password}}
  ```

- Show detailed information about your PlatformIO account:

  ```
  pio account show
  ```

- Reset your password using your username or email:

  ```
  pio account forgot --username {{username_or_email}}
  ```

pio boards

List pre-configured embedded boards available in P latformIO .

More information: [https://docs.platformio.org/en/latest/core/userguide/cmd_ boards.html](https://docs.platformio.org/en/latest/core/userguide/cmd_boards.html)

- List all available boards:

  ```
  pio b oards
  ```

- List only boards from installed platforms:

  ```
  pio b oards - - installed
  ```

pio check

Perform a static analysis check on a PlatformIO project.

More information: https://docs.platformio.org/en/latest/core/userguide/cmd_check.html

- Perform a basic analysis check on the current project:

  ```
  pio check
  ```

- Perform a basic analysis check on a specific project:

  ```
  pio check --project-dir {{project_dir}}
  ```

- Perform an analysis check for a specific environment:

  ```
  pio check --environment {{environment}}
  ```

- Perform an analysis check and only report a specified defect severity type:

  ```
  pio check --severity {{low|medium|high}}
  ```

- Perform an analysis check and show detailed information when processing environments:

  ```
  pio check --verbose
  ```

pio ci

Build PlatformIO projects with an arbitrary source code structure.

This will create a new temporary project which the source code will be copied into.

More information: https://docs.platformio.org/en/latest/core/userguide/cmd_ci.html

- Build a PlatformIO project in the default system temporary directory and delete it afterwards:

  ```
  pio ci {{path/to/project}}
  ```

- Build a PlatformIO project and specify specific libraries:

  ```
  pio ci --lib {{path/to/library_directory}} {{path/to/project}}
  ```

- Build a PlatformIO project and specify a specific board (`pio boards` lists all of them):

  ```
  pio ci --board {{board}} {{path/to/project}}
  ```

- Build a PlatformIO project in a specific directory:

  ```
  pio ci --build-dir {{path/to/build_directory}} {{path/to/project}}
  ```

- Build a PlatformIO project and don't delete the build directory:

  ```
  pio ci --keep-build-dir {{path/to/project}}
  ```

- Build a PlatformIO project using a specific configuration file:

  ```
  pio ci --project-conf {{path/to/platformio.ini}}
  ```

pio debug

Debug PlatformIO projects.

More information: https://docs.platformio.org/en/latest/core/userguide/cmd_debug.html

- Debug the PlatformIO project in the current directory:

  `pio debug`

- Debug a specific PlatformIO project:

  `pio debug --project-dir {{path/to/platformio_project}}`

- Debug a specific environment:

  `pio debug --environment {{environment}}`

- Debug a PlatformIO project using a specific configuration file:

  `pio debug --project-conf {{path/to/platformio.ini}}`

- Debug a PlatformIO project using the `gdb` debugger:

  `pio debug --interface={{gdb}} {{gdb_options}}`

pio device

Manage and monitor PlatformIO devices.

More information: https://docs.platformio.org/en/latest/core/userguide/device/.

- List all available serial ports:

  ```
  pio device list
  ```

- List all available logical devices:

  ```
  pio device list --logical
  ```

- Start an interactive device monitor:

  ```
  pio device monitor
  ```

- Start an interactive device monitor and listen to a specific port:

  ```
  pio device monitor --port {{/dev/ttyUSB X}}
  ```

- Start an interactive device monitor and set a specific baud rate (defaults to 9600):

  ```
  pio device monitor --baud {{57600}}
  ```

- Start an interactive device monitor and set a specific EOL character (defaults to CRLF):

  ```
  pio device monitor --eol {{CRLF|CR|LF}}
  ```

- Go to the menu of the interactive device monitor:

  ```
  Ctrl + T
  ```

pio home

Launch the PlatformIO Home web server.

More information: https://docs.platformio.org/en/latest/core/userguide/cmd_home.html

- Open PlatformIO Home in the default web browser:

  ```
  pio home
  ```

- Use a specific HTTP port (defaults to 8008):

  ```
  pio home --port {{port}}
  ```

- Bind to a specific IP address (defaults to 127.0.0.1):

  ```
  pio home --host {{ip_address}}
  ```

- Do not automatically open PlatformIO Home in the default web browser:

  ```
  pio home --no-open
  ```

- Automatically shutdown the server on timeout (in seconds) when no clients are connected:

  ```
  pio home --shutdown-timeout {{time}}
  ```

- Specify a unique session identifier to keep PlatformIO Home isolated from other instances and protected from 3rd party access:

  ```
  pio home --session-id {{id}}
  ```

pio init

This command is an alias of `pio project init`.

- View documentation for the original command:

  `tldr pio project`

pio lib

Manage PlatformIO libraries.

More information: https://docs.platformio.org/en/latest/core/userguide/lib/.

- List installed libraries:

  ```
  pio lib list
  ```

- List built-in libraries based on installed development platforms and their frameworks:

  ```
  pio lib builtin
  ```

- Search for existing libraries:

  ```
  pio lib search {{keyword}}
  ```

- Show details about a library:

  ```
  pio lib show {{library}}
  ```

- Install a library:

  ```
  pio lib install {{library}}
  ```

- Update installed libraries:

  ```
  pio lib update
  ```

- Uninstall a library:

  ```
  pio lib uninstall {{library}}
  ```

- Show PlatformIO library registry statistics:

  ```
  pio lib stats
  ```

pio org

Manage PlatformIO organizations and their owners.

More information: https://docs.platformio.org/en/latest/core/userguide/org/.

- Create a new organization:

  ```
  pio org create {{organization_name}}
  ```

- Delete an organization:

  ```
  pio org destroy {{organization_name}}
  ```

- Add a user to an organization:

  ```
  pio org add {{organization_name}} {{username}}
  ```

- Remove a user from an organization:

  ```
  pio org remove {{organization_name}} {{username}}
  ```

- List all organizations the current user is a member of and their owners:

  ```
  pio org list
  ```

- Update the name, email or display name of an organization:

  ```
  pio org update --orgname {{new_organization_name}} --email
  {{new_email}} --displayname {{new_display_name}}
  {{organization_name}}
  ```

pio package

Manage packages in the registry.

Packages can only be removed within 72 hours (3 days) from the date that they are published.

More information: https://docs.platformio.org/en/latest/core/userguide/package/.

- Create a package tarball from the current directory:

```
pio package --output {{path/to/package.tar.gz}}
```

- Create and publish a package tarball from the current directory:

```
pio package publish
```

- Publish the current directory and restrict public access to it:

```
pio package publish --private
```

- Publish a package:

```
pio package publish {{path/to/package.tar.gz}}
```

- Publish a package with a custom release date (UTC):

```
pio package publish {{path/to/package.tar.gz}} --released-at
" {{2021-04-08 21:15:38}}"
```

- Remove all versions of a published package from the registry:

```
pio package unpublish {{package_name}}
```

- Remove a specific version of a published package from the registry:

```
pio package unpublish {{package_name}}@{{version}}
```

- Undo the removal, putting all versions or a specific version of the package back into the registry:

```
pio package unpublish --undo {{package_name}}@{{version}}
```

pio platform

Manage PlatformIO development platforms.

More information: https://docs.platformio.org/en/latest/core/userguide/platforms/.

- List all installed development platforms:

  ```
  pio platform list
  ```

- Search for existing development platforms:

  ```
  pio platform search {{platform}}
  ```

- Show details about a development platform:

  ```
  pio platform show {{platform}}
  ```

- Install a development platform:

  ```
  pio platform install {{platform}}
  ```

- Update installed development platforms:

  ```
  pio platfom update
  ```

- Uninstall a development platform:

  ```
  pio platform uninstall {{platform}}
  ```

- List all supported frameworks:

  ```
  pio platform frameworks
  ```

pio project

Tool to manage PlatformIO projects.

More information: https://docs.platformio.org/en/latest/core/userguide/project/

- Initialize a new PlatformIO project:

  ```
  pio project init
  ```

- Initialize a new PlatformIO project in a specific directory:

  ```
  pio project init --project-dir {{path/to/project_directory}}
  ```

- Initialize a new PlatformIO project, specifying a board ID:

  ```
  pio project init --board {{ATmega328P|uno|...}}
  ```

- Initialize a new PlatformIO based project, specifying one or more project options:

  ```
  pio project init --project-option="{{option}}={{value}}" --
  project-option="{{option}}={{value}}"
  ```

- Print the configuration of a project:

  ```
  pio project config
  ```

pio remote

Helper command for PlatformIO Remote Development.

`pio remote [command]` takes the same arguments as its locally executing counterpart `pio [command]`.

More information: https://docs.platformio.org/en/latest/core/userguide/remote/index.html.

- List all active Remote Agents:

  ```
  pio remote agent list
  ```

- Start a new Remote Agent with a specific name and share it with friends:

  ```
  pio remote agent start --name {{agent_name}} --share {{example1@example.com}} --share {{example2@example.com}}
  ```

- List devices from specified Agents (omit `--agent` to specify all Agents):

  ```
  pio remote --agent {{agent_name1}} --agent {{agent_name2}} device list
  ```

- Connect to the serial port of a remote device:

  ```
  pio remote --agent {{agent_name}} device monitor
  ```

- Run all targets on a specified Agent:

  ```
  pio remote --agent {{agent_name}} run
  ```

- Update installed core packages, development platforms and global libraries on a specific Agent:

  ```
  pio remote --agent {{agent_name}} update
  ```

- Run all tests in all environments on a specific Agent:

  ```
  pio remote --agent {{agent_name}} test
  ```

pio run

Run PlatformIO project targets.

More information: https://docs.platformio.org/en/latest/core/userguide/cmd_run.html

- List all available project targets:

  ```
  pio run --list-targets
  ```

- List all available project targets of a specific environment:

  ```
  pio run --list-targets --environment {{environment}}
  ```

- Run all targets:

  ```
  pio run
  ```

- Run all targets of specified environments:

  ```
  pio run --environment {{environment1}} --environment {{environment2}}
  ```

- Run specified targets:

  ```
  pio run --target {{target1}} --target {{target2}}
  ```

- Run the targets of a specified configuration file:

  ```
  pio run --project-conf {{path/to/platformio.ini}}
  ```

pio settings

View and modify PlatformIO settings.

More information: https://docs.platformio.org/en/latest/core/userguide/cmd_settings.html

- Display the names, values and descriptions of all PlatformIO settings:

  ```
  pio settings get
  ```

- Display the name, value and description of a specific PlatformIO setting:

  ```
  pio settings get {{setting}}
  ```

- Set a specific setting value:

  ```
  pio settings set {{setting}} {{value}}
  ```

- Reset the values of all modified settings to their factory defaults:

  ```
  pio settings reset
  ```

pio system

Miscellaneous system commands for PlatformIO.

More information: https://docs.platformio.org/en/latest/core/userguide/system/.

- Install shell completion for the current shell (supports bash, fish, zsh and powershell):

  ```
  pio system completion install
  ```

- Uninstall shell completion for the current shell:

  ```
  pio system completion uninstall
  ```

- Display system-wide PlatformIO information:

  ```
  pio system info
  ```

- Remove unused PlatformIO data:

  ```
  pio system prune
  ```

- Remove only cached data:

  ```
  pio system prune --cache
  ```

- List unused PlatformIO data that would be removed but do not actually remove it:

  ```
  pio system prune --dry-run
  ```

pio team

Manage PlatformIO teams.

More information: https://docs.platformio.org/en/latest/core/userguide/team/.

- Create a new team with the specified description:

  ```
  pio team create --description {{description}}
  {{organization_name}}:{{team_name}}
  ```

- Delete a team:

  ```
  pio team destroy {{organization_name}}:{{team_name}}
  ```

- Add a new user to a team:

  ```
  pio team add {{organization_name}}:{{team_name}} {{username}}
  ```

- Remove a user from a team:

  ```
  pio team remove {{organization_name}}:{{team_name}}
  {{username}}
  ```

- List all teams that the user is part of and their members:

  ```
  pio team list
  ```

- List all teams in an organization:

  ```
  pio team list {{organization_name}}
  ```

- Rename a team:

  ```
  pio team update --name {{new_team_name}}
  {{organization_name}}:{{team_name}}
  ```

- Change the description of a team:

  ```
  pio team update --description {{new_description}}
  {{organization_name}}:{{team_name}}
  ```

pio test

Run local tests on a PlatformIO project.

More information: https://docs.platformio.org/en/latest/core/userguide/cmd_test.html

- Run all tests in all environments of the current PlatformIO project:

  ```
  pio test
  ```

- Test only specific environments:

  ```
  pio test --environment {{environment1}} --environment {{environment2}}
  ```

- Run only tests whose name matches a specific glob pattern:

  ```
  pio test --filter "{{pattern}}"
  ```

- Ignore tests whose name matches a specific glob pattern:

  ```
  pio test --ignore "{{pattern}}"
  ```

- Specify a port for firmware uploading:

  ```
  pio test --upload-port {{upload_port}}
  ```

- Specify a custom configuration file for running the tests:

  ```
  pio test --project-conf {{path/to/platformio.ini}}
  ```

pio update

Update installed PlatformIO Core packages, development platforms and global libraries.

See also `pio platform update`, `pio lib update`

More information: https://docs.platformio.org/en/latest/core/userguide/cmd_update.html

- Perform a full update of all packages, development platforms and global libraries:

  ```
  pio update
  ```

- Update core packages only (skips platforms and libraries):

  ```
  pio update --core-packages
  ```

- Check for new versions of packages, platforms and libraries but do not actually update them:

  ```
  pio update --dry-run
  ```

pio upgrade

Update PlatformIO to the latest version.

More information: https://docs.platformio.org/en/latest/core/userguide/cmd_upgrade.html

- Update PlatformIO to the latest version:

  ```
  pio upgrade
  ```

- Update PlatformIO to the latest development (unstable) version:

  ```
  pio upgrade --dev
  ```

pio

Development environment for embedded boards.

More information: https://docs.platformio.org/en/latest/core/userguide/.

- S how help and list subcommands:

  ```
  pio - - help
  ```

- P rint the version number and exit:

  ```
  pio - - v ersion
  ```

- S how help for a specific subcommand:

  ```
  pio {{su b com m and}} - - help
  ```

piodebuggdb

This command is an alias of `pio debug --interface=gdb`.

- View documentation for the original command:

  `tldr pio debug`

pip install

Install P ython packages.

More information: https://pip.pypa.io.

- Install a package:

  ```
  pip install {{pack ag e_nam e}}
  ```

- Install a specific version of a package:

  ```
  pip install {{pack ag e_nam e}}= = {{pack ag e_v ersion}}
  ```

- Install packages listed in a file:

  ```
  pip install - r {{req u irem ents. txt}}
  ```

- Install the local package in the current directory in develop (editable) mode:

  ```
  pip install - e .
  ```

pip

Python package manager.

More information: https://pip.pypa.io.

- Install a package (see `pip install` for more install examples):

  `pip install {{package_name}}`

- Upgrade a package:

  `pip install -U {{package_name}}`

- Uninstall a package:

  `pip uninstall {{package_name}}`

- Save installed packages to file:

  `pip freeze > {{requirements.txt}}`

- Show installed package info:

  `pip show {{package_name}}`

pip3

Python package manager.

More information: https://pip.pypa.io.

- Find available packages:

  ```
  pip3 search {{package name}}
  ```

- Install a package:

  ```
  pip3 install {{package name}}
  ```

- Install a specific version of a package:

  ```
  pip3 install {{package name}}==={{package version}}
  ```

- Upgrade a package:

  ```
  pip3 install --upgrade {{package name}}
  ```

- Uninstall a package:

  ```
  pip3 uninstall {{package name}}
  ```

- Save the list of installed packages to a file:

  ```
  pip3 freeze > {{requirements.txt}}
  ```

- Install packages from a file:

  ```
  pip3 install --requirement {{requirements.txt}}
  ```

- Show installed package info:

  ```
  pip3 show {{package name}}
  ```

pipenv

Simple and unified Python development workflow.

Manages packages and the virtual environment for a project.

More information: https://pypi.org/project/pipenv

- Create a new project:

  `pipenv`

- Create a new project using Python 3:

  `pipenv --three`

- Install a package:

  `pipenv install {{package_name}}`

- Install all the dependencies for a project:

  `pipenv install`

- Install all the dependencies for a project (including dev packages):

  `pipenv install --dev`

- Uninstall a package:

  `pipenv uninstall {{package_name}}`

- Start a shell within the created virtual environment:

  `pipenv shell`

- Generate a `requirements.txt` (list of dependencies) for a project:

  `pipenv lock --requirements`

pipx

Install and run python applications in isolated environments.

More information: https://github.com/pipxproj ect/pipx

- R un an app in a temporary virtual environment:

  ```
  pipx ru n {{pycow say}} {{m oo}}
  ```

- Install a package in a virtual environment and add entry points to path:

  ```
  pipx install {{pack ag e}}
  ```

- List installed packages:

  ```
  pipx list
  ```

- R un an app in a temporary virtual environment with a package name different from the executable:

  ```
  pipx ru n - - spec {{httpx- cli}} {{httpx}} {{http: //
  w w w . g ithu b . com }}
  ```

pixiecore

Tool to manage the network booting of machines.

More information: https://github.com/danderson/netboot/tree/master/pixiecore.

- Start a PXE boot server which provides a `netboot.xyz` boot image:

  `pixiecore {{quick}} xyz --dhcp-no-bind`

- Start a new PXE boot server which provides an Ubuntu boot image:

  `pixiecore {{quick}} ubuntu --dhcp-no-bind`

- Get a list of all available boot images for quick mode:

  `pixiecore quick --help`

pkill

Signal process by name.

Mostly used for stopping processes.

More information: https://www.man7.org/linux/man-pages/man1/pkill.1.html

- Kill all processes which match:

  ```
  pkill -9 "{{process_name}}"
  ```

- Kill all processes which match their full command instead of just the process name:

  ```
  pkill -9 --full "{{command_name}}"
  ```

- Send SIGUSR1 signal to processes which match:

  ```
  pkill -USR1 "{{process_name}}"
  ```

- Kill the main `firefox` process to close the browser:

  ```
  pkill --oldest "{{firefox}}"
  ```

plantuml

Create UML diagrams from a plain text language and render them in different formats.

More information: https://plantuml.com/en/command-line.

- Render diagrams to default format (PNG):

  ```
  plantuml {{diagram1.puml}} {{diagram2.puml}}
  ```

- Render a diagram in given format (e.g. `png`, `pdf`, `svg` `txt`):

  ```
  plantuml -t {{format}} {{diagram.puml}}
  ```

- Render all diagrams of a directory:

  ```
  plantuml {{path/to/diagrams}}
  ```

- Render a diagram to the output directory:

  ```
  plantuml -o {{path/to/output}} {{diagram.puml}}
  ```

- Render a diagram with the configuration file:

  ```
  plantuml -config {{config.cfg}} {{diagram.puml}}
  ```

- Display help:

  ```
  plantuml -help
  ```

platformio

This command is an alias of **`pio`**

- View documentation for the original command:

  `tldr pio`

play

Audio player of S oX - S ound eX change.

P lays any audio from the command-line, with audio formats identified by the extension.

More information: http://sox.sourceforge.net.

- P lay the given audio file:

  ```
  play {{au diofile}}
  ```

- P lay the given audio files:

  ```
  play {{au diofile1 }} {{au diofile2 }}
  ```

- P lay the given audio at twice the speed:

  ```
  play {{au diofile}} speed 2 . 0
  ```

- P lay the given audio in reverse:

  ```
  play {{au diofile}} rev erse
  ```

plesk

Plesk hosting control panel CLI interface.

More information: https://docs.plesk.com.

- Generate an auto login link for the admin user and print it:

  ```
  plesk log in
  ```

- Show product version information:

  ```
  plesk version
  ```

- List all hosted domains:

  ```
  plesk bin domain --list
  ```

- Start watching for changes in the `panel.log` file:

  ```
  plesk log {{panel.log}}
  ```

- Start the interactive MySQL console:

  ```
  plesk db
  ```

- Open the Plesk main configuration file in the default editor:

  ```
  plesk conf {{panel.ini}}
  ```

pm2

Process manager for Node.js.

Used for log management, monitoring and configuring processes.

More information: https://pm2.keymetrics.io

- Start a process with a name that can be used for later operations:

  ```
  pm2 start {{app.js}} --name {{myapp}}
  ```

- List processes:

  ```
  pm2 list
  ```

- Monitor all processes:

  ```
  pm2 monit
  ```

- Stop a process:

  ```
  pm2 stop {{myapp}}
  ```

- Restart a process:

  ```
  pm2 restart {{myapp}}
  ```

- Dump all processes for resurrecting them later:

  ```
  pm2 save
  ```

- Resurrect previously dumped processes:

  ```
  pm2 resurrect
  ```

pngcheck

P rint detailed information about and verify P N G , J N G , and MN G files.

More information: http://www.libpng.org/pub/png/apps/pngcheck.html.

- P rint a summary for an image (width, height, and color depth):

  ```
  png check {{im ag e. png }}
  ```

- P rint information for an image with [c] oloriz ed output:

  ```
  png check - c {{im ag e. png }}
  ```

- P rint [v] erbose information for an image:

  ```
  png check - cv t {{im ag e. png }}
  ```

- R eceive an image from stdin and display detailed information:

  ```
  cat {{path/to/im ag e. png }} | png check - cv t
  ```

- [s] earch for P N G s within a specific file and display information about them:

  ```
  png check - s {{im ag e. png }}
  ```

- S earch for P N G s within another file and e[x] tract them:

  ```
  png check - x {{im ag e. png }}
  ```

pngcrush

P N G image compression utility.

More information: https://pmt.sourceforge.io/pngcrush.

- Compress a P N G file:

```
png cru sh {{in. png }} {{ou t. png }}
```

- Compress all P N G s and output to directory:

```
png cru sh - d {{path/to/ou tpu t}} * . png
```

- Compress P N G file with all 1 1 4 available algorithms and pick the best result:

```
png cru sh - rem allb - b ru te - redu ce {{in. png }} {{ou t. png }}
```

pnpm

Fast, disk space efficient package manager for Node.js.

Manage Node.js projects and their module dependencies.

More information: https://pnpm.io.

- Interactively create a `package.json` file:

  ```
  pnpm init
  ```

- Download all the packages listed as dependencies in `package.json`

  ```
  pnpm install
  ```

- Download a specific version of a package and add it to the list of dependencies in `package.json`

  ```
  pnpm install {{module_name}}@{{version}}
  ```

- Download a package and add it to the list of dev dependencies in `package.json`

  ```
  pnpm install --dev {{module_name}}
  ```

- Download a package and install it globally:

  ```
  pnpm install -g {{module_name}}
  ```

- Uninstall a package and remove it from the list of dependencies in `package.json`

  ```
  pnpm uninstall {{module_name}}
  ```

- Print a tree of locally installed modules:

  ```
  pnpm list
  ```

- List top-level [g]lobally installed modules:

  ```
  pnpm list -g --depth={{0}}
  ```

pnpx

Directly execute binaries from npm packages, using **`pnpm`** instead of **`npm`**

More information: https://pnpm.io/pnpx-cli.

- Execute the binary from a given npm module:

  ```
  pnpx {{module_name}}
  ```

- Execute a specific binary from a given npm module, in case the module has multiple binaries:

  ```
  pnpx --package {{package_name}} {{module_name}}
  ```

- Display help:

  ```
  pnpx --help
  ```

podman

Simple management tool for pods, containers and images.

Podman provides a Docker-CLI comparable command-line. Simply put `alias docker=podman`.

More information: https://github.com/containers/podman/blob/main/commands-demo.md.

- Print out information about containers:

  ```
  podman ps
  ```

- List all containers (both running and stopped):

  ```
  podman ps --all
  ```

- Start one or more containers:

  ```
  podman start {{container_name}} {{container_id}}
  ```

- Stop one or more running containers:

  ```
  podman stop {{container_name}} {{container_id}}
  ```

- Pull an image from a registry (defaults to the Docker Hub):

  ```
  podman pull {{image_name}}:{{image_tag}}
  ```

- Open a shell inside of an already running container:

  ```
  podman exec --interactive --tty {{container_name}} {{sh}}
  ```

- Remove one or more stopped containers:

  ```
  podman rm {{container_name}} {{container_id}}
  ```

- Display the logs of one or more containers and follow log output:

  ```
  podman logs --follow {{container_name}} {{container_id}}
  ```

poetry

Manage Python packages and dependencies.

More information: https://python-poetry.org/docs.

- Create a new Poetry project in the directory with a specific name:

  `poetry new {{project_name}}`

- Install a dependency and its subdependencies:

  `poetry add {{dependency}}`

- Interactively initialize the current directory as a new Poetry project:

  `poetry init`

- Get the latest version of all dependencies and update `poetry.lock`:

  `poetry update`

- Execute a command inside the project's virtual environment:

  `poetry run {{command}}`

popd

Remove a directory placed on the directory stack via the pushd shell built-in.

See also **pushd** to place a directory on the stack and **dirs** to display directory stack contents.

- Remove the top directory from the stack and cd to it:

  ```
  popd
  ```

- Remove the N th directory (starting from zero to the left from the list printed with `dirs`):

  ```
  popd +N
  ```

- Remove the N th directory (starting from zero to the right from the list printed with `dirs`):

  ```
  popd -N
  ```

popeye

Utility that reports potential issues with Kubernetes deployment manifests.

More information: https://github.com/derailed/popeye.

- Scan the current Kubernetes cluster:

  ```
  popeye
  ```

- Scan a specific namespace:

  ```
  popeye -n {{namespace}}
  ```

- Scan specific Kubernetes context:

  ```
  popeye --context={{context}}
  ```

- Use a spinach configuration file for scanning:

  ```
  popeye -f {{spinach.yaml}}
  ```

postcss

PostCSS is a tool for transforming styles with JS plugins.

More information: https://postcss.org.

- Parse and transform a CSS file:

  ```
  postcss {{path/to/file}}
  ```

- Parse and transform a CSS file and output to a specific file:

  ```
  postcss {{path/to/file}} --output {{path/to/file}}
  ```

- Parse and transform a CSS file and output to a specific directory:

  ```
  postcss {{path/to/file}} --dir {{path/to/directory}}
  ```

- Parse and transform a CSS file in-place:

  ```
  postcss {{path/to/file}} --replace
  ```

- Specify a custom PostCSS parser:

  ```
  postcss {{path/to/file}} --parser {{parser}}
  ```

- Specify a custom PostCSS syntax:

  ```
  postcss {{path/to/file}} --syntax {{syntax}}
  ```

- Watch for changes to a CSS file:

  ```
  postcss {{path/to/file}} --watch
  ```

- Display available options and examples:

  ```
  postcss --help
  ```

powerstat

Measures the power consumption of a computer that has a battery power source or supports the RAPL interface.

More information: http://manpages.ubuntu.com/manpages/bionic/man8/powerstat.8.html

- Measure power with the default of 10 samples with an interval of 10 seconds:

```
powerstat
```

- Measure power with custom number of samples and interval duration:

```
powerstat {{interval}} {{number_of_samples}}
```

- Measure power using Intel's RAPL interface:

```
powerstat -R {{interval}} {{number_of_samples}}
```

- Show an histogram of the power measurements:

```
powerstat -H {{interval}} {{number_of_samples}}
```

- Enable all statistics gathering options:

```
powerstat -a {{interval}} {{number_of_samples}}
```

pprof

Command-line tool for visualiz ation and analysis of profile data.

More information: https://github.com/google/pprof.

- G enerate a text report from a specific profiling file, on fibbo binary:

  `pprof - top {{. /fib b o}} {{. /fib b o- profile. pb . g z}}`

- G enerate a graph and open it on a web browser:

  `pprof - sv g {{. /fib b o}} {{. /fib b o- profile. pb . g z}}`

- R un pprof in interactive mode to be able to manually launch `pprof` on a file:

  `pprof {{. /fib b o}} {{. /fib b o- profile. pb . g z}}`

- R un a web server that serves a web interface on top of `pprof`:

  `pprof - http= {{localhost: 8 0 8 0 }} {{. /fib b o}} {{. /fib b o- profile. pb . g z}}`

- F etch a profile from an H T T P server and generate a report:

  `pprof {{http: //localhost: 8 0 8 0 /deb u g /pprof}}`

pr

Paginate or columnate files for printing.

More information: https://www.gnu.org/software/coreutils/pr.

- Print multiple files with a default header and footer:

```
pr {{file1}} {{file2}} {{file3}}
```

- Print with a custom centered header:

```
pr -h "{{header}}" {{file1}} {{file2}} {{file3}}
```

- Print with numbered lines and a custom date format:

```
pr -n -D "{{format}}" {{file1}} {{file2}} {{file3}}
```

- Print all files together, one in each column, without a header or footer:

```
pr -m -T {{file1}} {{file2}} {{file3}}
```

- Print, beginning at page 2 up to page 5, with a given page length (including header and footer):

```
pr +{{2}}:{{5}} -l {{page_length}} {{file1}} {{file2}} {{file3}}
```

- Print with an offset for each line and a truncating custom page width:

```
pr -o {{offset}} -W {{width}} {{file1}} {{file2}} {{file3}}
```

pre-commit

Create G it hooks that get run before a commit.

More information: https://pre-commit.com.

- Install pre-commit into your G it hooks:

  ```
  pre- com m it install
  ```

- R un pre-commit hooks on all staged files:

  ```
  pre- com m it ru n
  ```

- R un pre-commit hooks on all files, staged or unstaged:

  ```
  pre- com m it ru n - - all- files
  ```

- Clean pre-commit cache:

  ```
  pre- com m it clean
  ```

prettier

An opinionated code formatter for JavaScript, JSON, CSS, YAML, and more.

More information: https://prettier.io/.

- Format a file and print the result to stdout:

  ```
  prettier {{path/to/file}}
  ```

- Check if a specific file has been formatted:

  ```
  prettier --check {{path/to/file}}
  ```

- Run with a specific configuration file:

  ```
  prettier --config {{path/to/config_file}} {{path/to/file}}
  ```

- Format a file or directory, replacing the original:

  ```
  prettier --write {{path/to/file_or_directory}}
  ```

- Format files or directories recursively using single quotes and no trailing commas:

  ```
  prettier --single-quote --trailing-comma {{none}} --write
  {{path/to/file_or_directory}}
  ```

pretty-bytes

Convert bytes to a human readable string.

More information: https://github.com/sindresorhus/pretty-bytes-cli.

- Convert numeric bytes value to a human readable string:

  ```
  pretty-bytes {{1337}}
  ```

- Convert numeric bytes value from stdin to a human readable string:

  ```
  echo {{1337}} | pretty-bytes
  ```

- Display help and usage information:

  ```
  pretty-bytes --help
  ```

printenv

Print values of all or specific environment variables.

More information: https://www.gnu.org/software/coreutils/printenv.

- Display key-value pairs of all environment variables:

  ```
  printenv
  ```

- Display the value of a specific variable:

  ```
  printenv {{HOME}}
  ```

- Display the value of a variable and end with NUL instead of newline:

  ```
  printenv --null {{HOME}}
  ```

printf

Format and print text.

More information: https://www.gnu.org/software/coreutils/printf.

- Print a text message:

```
printf "{{%s\n}}" "{{Hello world}}"
```

- Print an integer in bold blue:

```
printf "{{\e[1;34m%.3d\e[0m\n}}" {{42}}
```

- Print a float number with the Unicode Euro sign:

```
printf "{{\u20AC%.2f\n}}" {{123.4}}
```

- Print a text message composed with environment variables:

```
printf "{{var1: %s\tvar2: %s\n}}" "{{$VAR1}}" "{{$VAR2
```

- Store a formatted message in a variable (does not work on zsh):

```
printf -v {{myvar}} {{"This is %s = %d\n" "a year" 2016}}
```

progpilot

A P H P static analysis tool for detecting security vulnerabilities.

More information: https://github.com/designsecurity/progpilot.

- Analyse the current directory:

  ```
  prog pilot
  ```

- Analyse a specific file or directory:

  ```
  prog pilot {{path/to/file_or_directory}}
  ```

- S pecify a custom configuration file:

  ```
  prog pilot - - config u ration {{path/to/config u ration. ym l}}
  ```

promtool

Tooling for the Prometheus monitoring system.

More information: https://prometheus.io/docs/prometheus/latest/getting_started/

- Check if the config files are valid or not (if present report errors):

  ```
  promtool check config {{config_file.yml}}
  ```

- Check if the rule files are valid or not (if present report errors):

  ```
  promtool check rules {{rules_file.yml}}
  ```

- Pass Prometheus metrics over stdin to check them for consistency and correctness:

  ```
  curl --silent {{http://example.com:9090/metrics/}} | promtool check metrics
  ```

- Unit tests for rules config:

  ```
  promtool test rules {{test_file.yml}}
  ```

prosodyctl

The control tool for the Prosody XMPP server.

More information: https://prosody.im/doc/prosodyctl.

- Show the status of the Prosody server:

  ```
  sudo prosodyctl status
  ```

- Reload the server's configuration files:

  ```
  sudo prosodyctl reload
  ```

- Add a user to the Prosody XMPP server:

  ```
  sudo prosodyctl adduser {{user@example.com}}
  ```

- Set a user's password:

  ```
  sudo prosodyctl passwd {{user@example.com}}
  ```

- Permanently delete a user:

  ```
  sudo prosodyctl deluser {{user@example.com}}
  ```

protector

Protect or unprotect branches on GitHub repositories.

More information: https://github.com/jacgay/protector

- Protect branches of a GitHub repository (create branch protection rules):

  ```
  protector {{branches_regex}} -repos {{organization/repository}}
  ```

- Use the dry run to see what would be protected (can also be used for freeing):

  ```
  protector -dry-run {{branches_regex}} -repos {{organization/repository}}
  ```

- Free branches of a GitHub repository (delete branch protection rules):

  ```
  protector -free {{branches_regex}} -repos {{organization/repository}}
  ```

protoc

Parse Google Protobuf `protos` files and generate output in the specified language.

More information: https://developers.google.com/protocol-buffers.

- Generate Python code from a `proto` file:

  ```
  protoc --python_out={{path/to/output_directory}}
  {{input_file.proto}}
  ```

- Generate Java code from a `proto` file that imports other `.proto` files:

  ```
  protoc --java_out={{path/to/output_directory}} --
  proto_path={{path/to/import_search_path}}
  {{input_file.proto}}
  ```

- Generate code for multiple languages:

  ```
  protoc --csharp_out={{path/to/c#_output_directory}} --
  js_out={{path/to/js_output_directory}} {{input_file.proto}}
  ```

ps

Information about running processes.

- List all running processes:

```
ps au x
```

- List all running processes including the full command string:

```
ps au xw w
```

- S earch for a process that matches a string:

```
ps au x |  g rep {{string }}
```

- List all processes of the current user in extra full format:

```
ps - - u ser $ ( id - u )  - F
```

- List all processes of the current user as a tree:

```
ps - - u ser $ ( id - u )  f
```

- G et the parent pid of a process:

```
ps - o ppid=  - p {{pid}}
```

- S ort processes by memory consumption:

```
ps - - sort size
```

psalm

A static analysis tool for finding errors in PHP applications.

More information: https://psalm.dev.

- Generate a Psalm configuration:

  ```
  psalm --init
  ```

- Analyse the current working directory:

  ```
  psalm
  ```

- Analyse a specific directory or file:

  ```
  psalm {{path/to/file_or_directory}}
  ```

- Analyse a project with a specific configuration file:

  ```
  psalm --config {{path/to/psalm.xml}}
  ```

- Include informational findings in the output:

  ```
  psalm --show-info
  ```

- Analyse a project and display statistics:

  ```
  psalm --stats
  ```

- Analyse a project in parallel with 4 threads:

  ```
  psalm --threads {{4}}
  ```

psgrep

Search running processes with **`pgrep`**

More information: https://jvz.github.io/psgrep

- Find process lines containing a specific string:

  `psgrep {{process_name}}`

- Find process lines containing a specific string, excluding headers:

  `psgrep -n {{process_name}}`

- Search using a simplified format (PID, user, command):

  `psgrep -s {{process_name}}`

psq l

P ostgreS Q L command-line client.

More information: https://www.postgresq l.org/docs/current/app-psq l.html

- Connect to database. It connects to localhost using default port 5 4 3 2 with default user as currently logged in user:

```
psq l {{datab ase}}
```

- Connect to database on given server host running on given port with given username, without a password prompt:

```
psq l - h {{host}} - p {{port}} - U {{u sernam e}} {{datab ase}}
```

- Connect to database; user will be prompted for password:

```
psq l - h {{host}} - p {{port}} - U {{u sernam e}} - W {{datab ase}}
```

- E xecute a single S Q L q uery or P ostgreS Q L command on the given database (useful in shell scripts):

```
psq l - c ' {{q u ery}}' {{datab ase}}
```

- E xecute commands from a file on the given database:

```
psq l {{datab ase}} - f {{file. sq l}}
```

pssh

Parallel SSH program.

- Run a command on two hosts, and print its output on each server inline:

```
pssh -i -H "{{host1}} {{host2}}" {{hostname -i}}
```

- Run a command and save the output to separate files:

```
pssh -H {{host1}} -H {{host2}} -o {{path/to/output_dir}}
{{hostname -i}}
```

- Run a command on multiple hosts, specified in a new-line separated file:

```
pssh -i -h {{path/to/hosts_file}} {{hostname -i}}
```

- Run a command as root (this asks for the root password):

```
pssh -i -h {{path/to/hosts_file}} -A -l {{root_username}}
{{hostname -i}}
```

- Run a command with extra SSH arguments:

```
pssh -i -h {{path/to/hosts_file}} -x "{{-O
VisualHostKey=yes}}" {{hostname -i}}
```

- Run a command limiting the number of parallel connections to 10:

```
pssh -i -h {{path/to/hosts_file}} -p {{10}} '{{cd dir; ./
script.sh; exit}}'
```

psysh

A runtime developer console, interactive debugger and R E P L for P H P .

More information: https://psysh.org.

- O pen a shell in the current directory:

  ```
  psysh
  ```

- O pen a shell in a specific directory:

  ```
  psysh - - cw d {{path/to/directory}}
  ```

- U se a specific configuration file:

  ```
  psysh - - config  {{path/to/file}}
  ```

pt

Platinum Searcher.

A code search tool similar to `ag`

More information: https://github.com/monochromegane/the_platinum_searcher

- Find files containing "foo" and print the files with highlighted matches:

  ```
  pt {{foo}}
  ```

- Find files containing "foo" and display count of matches in each file:

  ```
  pt -c {{foo}}
  ```

- Find files containing "foo" as a whole word and ignore its case:

  ```
  pt -wi {{foo}}
  ```

- Find "foo" in files with a given extension using a regular expression:

  ```
  pt -G='{{\.bar$}}' {{foo}}
  ```

- Find files whose contents match the regular expression, up to 2 directories deep:

  ```
  pt --depth={{2}} -e '{{^ba[rz]*$}}'
  ```

pueue add

Enqueue a task for execution.

More information: https://github.com/Nukesor/pueue

- Add any command to the default queue:

```
pueue add {{command}}
```

- Pass a list of flags or arguments to a command when enqueuing:

```
pueue add -- {{command --arg -f}}
```

- Add a command but do not start it if it's the first in a queue:

```
pueue add --stashed -- {{rsync --archive --compress /local/directory /remote/directory}}
```

- Add a command to a group and start it immediately, see `pueue group` to manage groups:

```
pueue add --immediate --group "{{CPU_intensive}}" -- {{ffmpeg -i input.mp4 frame_%d.png}}
```

- Add a command and start it after commands 9 and 12 finish successfully:

```
pueue add --after {{9}} {{12}} --group "{{torrents}}" -- {{transmission-cli torrent_file.torrent}}
```

- Add a command with a label after some delay has passed, see `pueue enqueue --help` for valid datetime formats:

```
pueue add --label "{{compressing large file}}" --delay "{{wednesday 10:30pm}}" -- "{{7za compressed_file.7z large_file.xml}}"
```

pueue clean

Remove all finished tasks from the list and clear logs.

More information: https://github.com/Nukesor/pueue

- Remove finished tasks and clear logs:

  ```
  pueue clean
  ```

- Only clean commands that finished successfully:

  ```
  pueue clean --successful-only
  ```

pueue completions

G enerates shell completion files for bash, elvish, fish, powershell, and z sh.

More information: https://github.com/N ukesor/pueue

- G enerate completions for bash:

  ```
  su do  pu eu e  com pletions  b ash  {{/u sr/share/b ash- com pletion/
  com pletions/pu eu e. b ash}}
  ```

- G enerate completions for z sh:

  ```
  su do  pu eu e  com pletions  zsh  {{/u sr/share/zsh/site- fu nctions}}
  ```

- G enerate completions for fish:

  ```
  su do  pu eu e  com pletions  fish  {{/u sr/share/fish/com pletions}}
  ```

pueue edit

Edit the command or path of a stashed or queued task.

More information: https://github.com/Nukesor/pueue

- Edit a task, see `pueue status` to get the task ID:

  `pueue edit {{task_id}}`

- Edit the path from which a task is executed:

  `pueue edit {{task_id}} --path`

- Edit a command with the specified editor:

  `EDITOR={{nano}} pueue edit {{task_id}}`

pueue enq ueue

Enq ueue stashed tasks.

See also: `pueue stash`

More information: https://github.com/Nukesor/pueue

- Enq ueue multiple stashed tasks at once:

  ```
  pueue enqueue {{task_id}} {{task_id}}
  ```

- Enq ueue a stashed task after 60 seconds:

  ```
  pueue enqueue --delay {{60}} {{task_id}}
  ```

- Enq ueue a stashed task next Wednesday:

  ```
  pueue enqueue --delay {{wednesday}} {{task_id}}
  ```

- Enq ueue a stashed task after four months:

  ```
  pueue enqueue --delay "4 months" {{task_id}}
  ```

- Enq ueue a stashed task on 2021-02-19:

  ```
  pueue enqueue --delay {{2021-02-19}} {{task_id}}
  ```

- List all available date/time formats:

  ```
  pueue enqueue --help
  ```

pueue follow

Follow the output of a currently running task.

See also: `pueue log`

More information: https://github.com/Nukesor/pueue.

- Follow the output of a task (stdout + stderr):

  `pueue follow {{task_id}}`

- Follow the stderr of a task:

  `pueue follow --err {{task_id}}`

pueue group

Display, add or remove groups.

More information: https://github.com/N ukesor/pueue

- S how all groups with their statuses and number of parallel j obs:

  ```
  pu eu e g rou p
  ```

- Add a custom group:

  ```
  pu eu e g rou p - - add " {{g rou p_nam e}}"
  ```

- R emove a group and move its tasks to the default group:

  ```
  pu eu e g rou p - - rem ov e " {{g rou p_nam e}}"
  ```

pueue help

Display help for subcommands.

More information: https://github.com/N ukesor/pueue

- S how all available subcommands and fl ags:

  ```
  pu eu e help
  ```

- S how help for a specific subcommand:

  ```
  pu eu e help {{su b com m and}}
  ```

pueue kill

Kill running tasks or whole groups.

More information: https://github.com/N ukesor/pueue

- Kill all tasks in the default group:

  ```
  pu eu e  k ill
  ```

- Kill a specific task:

  ```
  pu eu e  k ill {{task _id}}
  ```

- Kill a task and terminate all its child processes:

  ```
  pu eu e  k ill - - children {{task _id}}
  ```

- Kill all tasks in a group and pause the group:

  ```
  pu eu e  k ill - - g rou p {{g rou p_nam e}}
  ```

- Kill all tasks across all groups and pause all groups:

  ```
  pu eu e  k ill - - all
  ```

pueue log

Display the log output of 1 or more tasks.

See also `pueue status`

More information: https://github.com/Nukesor/pueue

- Show the last few lines of output from all tasks:

  ```
  pueue log
  ```

- Show the full output of a task:

  ```
  pueue log {{task_id}}
  ```

- Show the last few lines of output from several tasks:

  ```
  pueue log {{task_id}} {{task_id}}
  ```

- Print a specific number of lines from the tail of output:

  ```
  pueue log --lines {{number_of_lines}} {{task_id}}
  ```

pueue parallel

S et the amount of allowed parallel tasks.

More information: https://github.com/N ukesor/pueue

- S et the maximum number of tasks allowed to run in parallel, in the default group:

  ```
  pu eu e parallel {{m ax_nu m b er_of_parallel_task s}}
  ```

- S et the maximum number of tasks allowed to run in parallel, in a specific group:

  ```
  pu eu e parallel - - g rou p {{g rou p_nam e}}
  {{m axim u m _nu m b er_of_parallel_task s}}
  ```

pueue pause

Pause running tasks or groups.

See also: `pueue start`.

More information: https://github.com/Nukesor/pueue.

- Pause all tasks in the default group:

  ```
  pueue pause
  ```

- Pause a running task:

  ```
  pueue pause {{task_id}}
  ```

- Pause a running task and stop all its direct children:

  ```
  pueue pause --children {{task_id}}
  ```

- Pause all tasks in a group and prevent it from starting new tasks:

  ```
  pueue pause --group {{group_name}}
  ```

- Pause all tasks and prevent all groups from starting new tasks:

  ```
  pueue pause --all
  ```

pueue remove

Remove tasks from the list. Running or paused tasks need to be killed first.

More information: https://github.com/Nukesor/pueue

- Remove a killed or finished task:

```
pueue remove {{task_id}}
```

- Remove multiple tasks at once:

```
pueue remove {{task_id}} {{task_id}}
```

pueue reset

Kill everything and reset.

More information: https://github.com/N ukesor/pueue

- Kill all tasks and remove everything (logs, status, groups, task IDs):

  ```
  pu eu e reset
  ```

- Kill all tasks, terminate their children, and reset everything:

  ```
  pu eu e reset - - children
  ```

- R eset without asking for confirmation:

  ```
  pu eu e reset - - force
  ```

pueue restart

Restart tasks.

More information: https://github.com/Nukesor/pueue

- Restart a specific task:

```
pueue restart {{task_id}}
```

- Restart multiple tasks at once, and start them immediately (do not queue):

```
pueue restart --start-immediately {{task_id}} {{task_id}}
```

- Restart a specific task from a different path:

```
pueue restart --edit-path {{task_id}}
```

- Edit a command before restarting:

```
pueue restart --edit {{task_id}}
```

- Restart a task in-place (without queuing as a separate task):

```
pueue restart --in-place {{task_id}}
```

- Restart all failed tasks and stash them:

```
pueue restart --all-failed --stashed
```

pueue send

Send input to a task.

More information: https://github.com/Nukesor/pueue

- Send input to a running command:

  ```
  pueue send {{task_id}} "{{input}}"
  ```

- Send confirmation to a task expecting y/N (e.g. apt, cp):

  ```
  pueue send {{task_id}} {{y}}
  ```

pueue shutdown

Remotely shut down the daemon.

Only use this subcommand if the daemon isn't started by a service manager.

More information: https://github.com/Nukesor/pueue

- Shutdown the daemon without a service manager:

  ```
  pueue shutdown
  ```

pueue start

Resume operation of specific tasks or groups of tasks.

See also: **`pueue pause`**

More information: https://github.com/Nukesor/pueue

- Resume all tasks in the default group:

  ```
  pueue start
  ```

- Resume a specific task:

  ```
  pueue start {{task_id}}
  ```

- Resume multiple tasks at once:

  ```
  pueue start {{task_id}} {{task_id}}
  ```

- Resume all tasks and start their children:

  ```
  pueue start --all --children
  ```

- Resume all tasks in a specific group:

  ```
  pueue start group {{group_name}}
  ```

pueue stash

Stash tasks to prevent them starting automatically.

See also `pueue start` and `pueue enqueue`.

More information: https://github.com/Nukesor/pueue.

- Stash an enqueued task:

  ```
  pueue stash {{task_id}}
  ```

- Stash multiple tasks at once:

  ```
  pueue stash {{task_id}} {{task_id}}
  ```

- Start a stashed task immediately:

  ```
  pueue start {{task_id}}
  ```

- Enqueue a task to be executed when preceding tasks finish:

  ```
  pueue enqueue {{task_id}}
  ```

pueue status

Display the current status of all tasks.

More information: https://github.com/N ukesor/pueue

- S how the status of all tasks:

  ```
  pu eu e statu s
  ```

- S how the status of a specific group:

  ```
  pu eu e statu s - - g rou p {{g rou p_nam e}}
  ```

pueue switch

Switches the queue position of two enqueued or stashed commands.

More information: https://github.com/Nukesor/pueue

- Switch the priority of two tasks:

```
pueue switch {{task_id1}} {{task_id2}}
```

pueue

Pueue is a command-line task management tool for sequential and parallel execution of long-running tasks.

More information: https://github.com/Nukesor/pueue

- Show general help and available subcommands:

  ```
  pueue --help
  ```

- Check the version of pueue:

  ```
  pueue --version
  ```

- Execute a pueue subcommand:

  ```
  pueue {{subcommand}}
  ```

pup

Command-line HTML parsing tool.

More information: https://github.com/ericchiang/pup.

- Transform a raw HTML file into a cleaned, indented, and colored format:

  ```
  cat {{index.html}} | pup --color
  ```

- Filter HTML by element tag name:

  ```
  cat {{index.html}} | pup '{{tag}}'
  ```

- Filter HTML by id:

  ```
  cat {{index.html}} | pup '{{div#id}}'
  ```

- Filter HTML by attribute value:

  ```
  cat {{index.html}} | pup '{{input[type="text"]}}'
  ```

- Print all text from the filtered HTML elements and their children:

  ```
  cat {{index.html}} | pup '{{div}} text{}'
  ```

- Print HTML as JSON:

  ```
  cat {{index.html}} | pup '{{div}} json{}'
  ```

pushd

Place a directory on a stack so it can be accessed later.

See also **popd** to switch back to original directory and **dirs** to display directory stack contents.

- Switch to directory and push it on the stack:

  ```
  pushd {{directory}}
  ```

- Switch first and second directories on the stack:

  ```
  pushd
  ```

- Rotate stack by making the 5th element the top of the stack:

  ```
  pushd +4
  ```

pv

Monitor the progress of data through a pipe.

- Print the contents of the file and display a progress bar:

```
pv {{file}}
```

- Measure the speed and amount of data flow between pipes (`-s` is optional):

```
command1 | pv -s {{expected_amount_of_data_for_eta}} | command2
```

- Filter a file, see both progress and amount of output data:

```
pv -cN in {{big_text_file}} | grep {{pattern}} | pv -cN out > {{filtered_file}}
```

- Attach to an already running process and see its file reading progress:

```
pv -d {{PID}}
```

- Read an erroneous file, skip errors as `dd conv=sync,noerror` would:

```
pv -EE {{path/to/faulty_media}} > image.img
```

- Stop reading after reading specified amount of data, rate limit to 1 K/s:

```
pv -L 1K -S {{maximum_file_size_to_be_read}}
```

pwd

Print name of current/working directory.

More information: https://www.gnu.org/software/coreutils/pwd.

- Print the current directory:

  ```
  pwd
  ```

- Print the current directory, and resolve all symlinks (i.e. show the "physical" path):

  ```
  pwd -P
  ```

pwsh

PowerShell Core is a cross-platform automation and configuration tool/framework.

More information: https://docs.microsoft.com/powershell/.

- Start an instance of PowerShell:

  ```
  pwsh
  ```

- Execute a script and then exit:

  ```
  pwsh -File {{path/to/file.ps1}}
  ```

- Set the execution policy for the current session:

  ```
  pwsh -ExecutionPolicy {{AllSigned|Bypass|Default|RemoteSigned|Restricted|Undefined|Unrestricted}}
  ```

- Execute a command and then exit:

  ```
  pwsh -Command {{command}}
  ```

pycodestyle

A tool to check Python code against PEP 8 style conventions.

More information: https://pycodestyle.readthedocs.io.

- Check the style of a single file:

  ```
  pycodestyle {{file.py}}
  ```

- Check the style of multiple files:

  ```
  pycodestyle {{file1.py}} {{file2.py}} {{file3.py}}
  ```

- Show only the first occurrence of an error:

  ```
  pycodestyle --first {{file.py}}
  ```

- Show the source code for each error:

  ```
  pycodestyle --show-source {{file.py}}
  ```

- Show the specific PEP 8 text for each error:

  ```
  pycodestyle --show-pep8 {{file.py}}
  ```

pyenv virtualenv

Create virtual environments based on one's installed Python distributions.

More information: https://github.com/pyenv/pyenv-virtualenv.

- Create a new Python 3.6.6 virtual environment:

```
pyenv virtualenv {{3.6.6}} {{virtualenv_name}}
```

- List all existing virtual environments:

```
pyenv virtualenvs
```

- Activate a virtual environment:

```
pyenv activate {{virtualenv_name}}
```

- Deactivate the virtual environment:

```
pyenv deactivate
```

pyenv

Switch between multiple versions of Python easily.

More information: https://github.com/pyenv/pyenv.

- List all available commands:

  ```
  pyenv commands
  ```

- List all Python versions under the `${PYENV_ROOT}/versions` directory:

  ```
  pyenv versions
  ```

- Install a Python version under the `${PYENV_ROOT}/versions` directory:

  ```
  pyenv install {{2.7.10}}
  ```

- Uninstall a Python version under the `${PYENV_ROOT}/versions` directory:

  ```
  pyenv uninstall {{2.7.10}}
  ```

- Set Python version to be used globally in the current machine:

  ```
  pyenv global {{2.7.10}}
  ```

- Set Python version to be used in the current directory and all directories below it:

  ```
  pyenv local {{2.7.10}}
  ```

pyflakes

Checks Python source code files for errors.

More information: https://pypi.org/project/pyflakes

- Check a single Python file:

  ```
  pyflakes check {{path/to/file}}.py
  ```

- Check Python files in a specific directory:

  ```
  pyflakes check Path {{path/to/directory}}
  ```

- Check Python files in a directory recursively:

  ```
  pyflakes check Recursive {{path/to/directory}}
  ```

- Check all Python files found in multiple directories:

  ```
  pyflakes iterSourceCode {{path/to/directory_1}} {{path/to/directory_2}}
  ```

pygmentize

Python-based syntax highlighter.

- Highlight file syntax and print to standard output (language is inferred from the file extension):

  ```
  pygmentize {{file.py}}
  ```

- Explicitly set the language for syntax highlighting:

  ```
  pygmentize -l {{javascript}} {{input_file}}
  ```

- List available lexers (processors for input languages):

  ```
  pygmentize -L lexers
  ```

- Save output to a file in HTML format:

  ```
  pygmentize -f html -o {{output_file.html}} {{input_file.py}}
  ```

- List available output formats:

  ```
  pygmentize -L formatters
  ```

- Output an HTML file, with additional formatter options (full page, with line numbers):

  ```
  pygmentize -f html -O "full,linenos=True" -o
  {{output_file.html}} {{input_file}}
  ```

python

Python language interpreter.

More information: https://www.python.org.

- Call a Python interactive shell (REPL):

  ```
  python
  ```

- Execute script in a given Python file:

  ```
  python {{script.py}}
  ```

- Execute script as part of an interactive shell:

  ```
  python -i {{script.py}}
  ```

- Execute a Python expression:

  ```
  python -c "{{expression}}"
  ```

- Run library module as a script (terminates option list):

  ```
  python -m {{module}} {{arguments}}
  ```

- Install a package using pip:

  ```
  python -m pip install {{package_name}}
  ```

- Interactively debug a Python script:

  ```
  python -m pdb {{script.py}}
  ```

q

Execute SQL-like queries on .csv and .tsv files.

More information: https://harelba.github.io/q.

- Query `.csv` file by specifying the delimiter as `','`:

```
q -d',' "SELECT * from {{path/to/file}}"
```

- Query `.tsv` file:

```
q -t "SELECT * from {{path/to/file}}"
```

- Query file with header row:

```
q -d{{delimiter}} -H "SELECT * from {{path/to/file}}"
```

- Read data from stdin; `'-'` in the query represents the data from stdin:

```
{{output}} | q "select * from -"
```

- Join two files (aliased as `f1` and `f2` in the example) on column `c1`, a common column:

```
q "SELECT * FROM {{path/to/file}} f1 JOIN {{path/to/other_file}} f2 ON (f1.c1 = f2.c1)"
```

- Format output using an output delimiter with an output header line (note: command will output column names based on the input file header or the column aliases overridden in the query):

```
q -D{{delimiter}} -O "SELECT {{column}} as {{alias}} from {{path/to/file}}"
```

q cp

Copy files using the default text editor to define the filenames.

More information: https://www.nongnu.org/renameutils/.

- Copy a single file (open an editor with the source filename on the left and the target filename on the right):

```
q cp {{sou rce_file}}
```

- Copy multiple J P G files:

```
q cp {{* . j pg }}
```

- Copy files, but swap the positions of the source and the target filenames in the editor:

```
q cp - - option sw ap {{* . j pg }}
```

q dbus

Inter-Process Communication (IPC) and Remote Procedure Calling (RPC) mechanism originally developed for Linux.

More information: https://doc.qt.io/qt-5/qtdbus-index.html

- List available service names:

```
qdbus
```

- List object paths for a specific service:

```
qdbus {{service_name}}
```

- List methods, signals and properties available on a specific object:

```
qdbus {{service_name}} {{/path/to/object}}
```

- Execute a specific method passing arguments and display the returned value:

```
qdbus {{service_name}} {{/path/to/object}} {{method_name}}
{{argument1}} {{argument2}}
```

- Display the current brightness value in a KDE Plasma session:

```
qdbus {{org.kde.Solid.PowerManagement}} {{/org/kde/Solid/
PowerManagement/Actions/BrightnessControl}}
{{org.kde.Solid.PowerManagement.Actions.BrightnessControl.
```

- Set a specific brightness to a KDE Plasma session:

```
qdbus {{org.kde.Solid.PowerManagement}} {{/org/kde/Solid/
PowerManagement/Actions/BrightnessControl}}
{{org.kde.Solid.PowerManagement.Actions.BrightnessControl.
{{5000}}
```

- Invoke volume up shortcut in a KDE Plasma session:

```
qdbus {{org.kde.kglobalaccel}} {{/component/kmix}}
{{invokeShortcut}} "{{increase_volume}}"
```

- Display help:

```
qdbus --help
```

qemu-img

Tool for Quick Emulator Virtual HDD image creation and manipulation.

- Create disk image with a specific size (in gigabytes):

  ```
  qemu-img create {{image_name.img}} {{gigabytes}}G
  ```

- Show information about a disk image:

  ```
  qemu-img info {{image_name.img}}
  ```

- Increase or decrease image size:

  ```
  qemu-img resize {{image_name.img}} {{gigabytes}}G
  ```

- Dump the allocation state of every sector of the specified disk image:

  ```
  qemu-img map {{image_name.img}}
  ```

- Convert a VMware .vmdk disk image to a KVM .qcow2 disk image:

  ```
  qemu-img convert -O qcow2 {{path/to/file/foo.vmdk}} {{path/
  to/file/foo.qcow2}}
  ```

q emu

Generic machine emulator and virtualizer.

Supports a large variety of CPU architectures.

More information: https://www.qemu.org

- Boot from image emulating i386 architecture:

```
qemu-system-i386 -hda {{image_name.img}}
```

- Boot from image emulating x64 architecture:

```
qemu-system-x86_64 -hda {{image_name.img}}
```

- Boot QEMU instance with a live ISO image:

```
qemu-system-i386 -hda {{image_name.img}} -cdrom {{os_image.iso}} -boot d
```

- Specify amount of RAM for instance:

```
qemu-system-i386 -m 256 -hda image_name.img -cdrom os-image.iso -boot d
```

- Boot from physical device (e.g. from USB to test bootable medium):

```
qemu-system-i386 -hda /dev/{{storage_device}}
```

q mv

Move files and directories using the default text editor to define the filenames.

More information: https://www.nongnu.org/renameutils/.

- Move a single file (open an editor with the source filename on the left and the target filename on the right):

```
q m v {{sou rce_file}}
```

- Move multiple J P G files:

```
q m v {{* . j pg }}
```

- Move multiple directories:

```
q m v - d {{path/to/directory1 }} {{path/to/directory2 }} {{path/
to/directory3 }}
```

- Move all files and directories inside a directory:

```
q m v - - recu rsiv e {{path/to/directory}}
```

- Move files, but swap the positions of the source and the target filenames in the editor:

```
q m v - - option sw ap {{* . j pg }}
```

q pdf

Versatile PDF transformation software.

More information: https://github.com/q-pdf/q-pdf

- Extract pages 1-3,5 and 6-10 from a PDF file and save them as another one:

```
qpdf --empty --pages {{input.pdf}} {{1-3,5,6-10}} --
{{output.pdf}}
```

- Merge (concatenate) all the pages of a list of PDF files and save the result as a new PDF:

```
qpdf --empty --pages {{file1.pdf}} {{file2.pdf}}
{{file3.pdf}} --  {{output.pdf}}
```

- Merge (concatenate) given pages from a list of PDF files and save the result as a new PDF:

```
qpdf --empty --pages {{file1.pdf}} {{1,6-8}} {{file2.pdf}}
{{3,4,5}} --  {{output.pdf}}
```

- Write each group of n pages to a separate output file with a given filename pattern:

```
qpdf --split-pages=n {{input.pdf}} {{out_%d.pdf}}
```

- Rotate certain pages of a pdf with a given angle:

```
qpdf --rotate={{90:2,4,6}} --rotate={{180:7-8}} {{input
{{output.pdf}}
```

- Remove the password from a password protected file:

```
qpdf --password={{password}} --decrypt {{input.pdf}}
{{output.pdf}}
```

q r

Generate QR codes in the terminal with ANSI VT-100 escape codes.

More information: https://github.com/lincolnloop/python-qrcode/

- Generate a QR code:

```
echo "{{data}}" | qr
```

- Specify the error correction level (defaults to M):

```
echo "{{data}}" | qr --error-correction={{L|M|Q|H}}
```

q rencode

QR Code generator. Supports PNG and EPS.

More information: https://fukuchi.org/works/q-rencode

- Convert a string to a QR code and save to an output file:

```
q rencode -o {{path/to/output_file.png}} {{string}}
```

- Convert an input file to a QR code and save to an output file:

```
q rencode -o {{path/to/output_file.png}} -r {{path/to/input_file}}
```

- Convert a string to a QR code and print it in terminal:

```
q rencode -t ansiutf8 {{string}}
```

- Convert input from pipe to a QR code and print it in terminal:

```
echo {{string}} | q rencode -t ansiutf8
```

q tcreator

Cross-platform IDE for Qt applications.

More information: https://doc.qt.io/qtcreator/creator-cli.html

- Start Qt Creator:

  ```
  qtcreator
  ```

- Start Qt Creator and restore the last session:

  ```
  qtcreator -lastsession
  ```

- Start Qt Creator but don't load the specified plugin:

  ```
  qtcreator -noload {{plugin}}
  ```

- Start Qt Creator but don't load any plugins:

  ```
  qtcreator -noload {{all}}
  ```

- Start Qt Creator in presentation mode with pop-ups for keyboard shortcuts:

  ```
  qtcreator -presentationMode
  ```

- Start Qt Creator and show the diff from a specific commit:

  ```
  qtcreator -gitshow {{commit}}
  ```

q uilt

T ool to manage a series of patches.

More information: https://savannah.nongnu.org/proj ects/q uilt

- Import an existing patch from a file:

```
q u ilt im port {{path/to/filenam e. patch}}
```

- Create a new patch:

```
q u ilt new  {{filenam e. patch}}
```

- Add a file to the current patch:

```
q u ilt add {{path/to/file}}
```

- After editing the file, refresh the current patch with the changes:

```
q u ilt refresh
```

- Apply all the patches in the series file:

```
q u ilt pu sh - a
```

- R emove all applied patches:

```
q u ilt pop - a
```

q uota

Display users' disk space usage and allocated limits.

- S how disk q uotas in human readable units for the current user:

  ```
  q u ota - s
  ```

- V erbose output (also display q uotas on filesystems where no storage is allocated):

  ```
  q u ota - v
  ```

- Q uiet output (only display q uotas on filesystems where usage is over q uota):

  ```
  q u ota - q
  ```

- P rint q uotas for the groups of which the current user is a member:

  ```
  q u ota - g
  ```

- S how disk q uotas for another user:

  ```
  su do q u ota - u {{u sernam e}}
  ```

q utebrowser

A keyboard-driven, vim-like browser based on PyQt5.

More information: https://qutebrowser.org/

- Open qutebrowser with a specified storage directory:

  ```
  qutebrowser --basedir {{path/to/directory}}
  ```

- Open a qutebrowser instance with temporary settings:

  ```
  qutebrowser --set {{content.geolocation}} {{true|false}}
  ```

- Restore a named session of a qutebrowser instance:

  ```
  qutebrowser --restore {{session_name}}
  ```

- Launch qutebrowser, opening all URLs using the specified method:

  ```
  qutebrowser --target {{auto|tab|tab-bg|tab-silent|tab-b
  silent|window|private-window}}
  ```

- Open qutebrowser with a temporary base directory and print logs to stdout as JSON:

  ```
  qutebrowser --temp-basedir --json-logging
  ```

R

R language interpreter.

More information: https://www.r-project.org

- Start an R interactive shell (REPL):

```
R
```

- Check R version:

```
R --version
```

- Start R in vanilla mode (i.e. a blank session that doesn't save the workspace at the end):

```
R --vanilla
```

- Execute a file:

```
R -f {{path/to/file.R}}
```

- Execute an R expression and then exit:

```
R -e {{expr}}
```

- Run R with a debugger:

```
R -d {{debugger}}
```

- Check R packages from package sources:

```
R CMD check {{path/to/package_source}}
```

r2 e

Forwards RSS feeds to an email address.

Requires a configured `sendmail` or smtp setup.

More information: https://github.com/rss2_email/rss2_email

- Create a new feed database that sends email to an email address:

  ```
  r2 e new {{email_address}}
  ```

- Subscribe to a feed:

  ```
  r2 e add {{feed_name}} {{feed_URI}}
  ```

- Send new stories to an email address:

  ```
  r2 e run
  ```

- List all feeds:

  ```
  r2 e list
  ```

- Delete a feed at a specified index:

  ```
  r2 e delete {{index}}
  ```

rabin2

Get information about binary files (ELF, PE, Java CLASS, Mach-O) - symbols, sections, linked libraries, etc.

Comes bundled with **radare2**

- Display general information about a binary (architecture, type, endianness):

  ```
  rabin2 -I {{path/to/binary}}
  ```

- Display linked libraries:

  ```
  rabin2 -l {{path/to/binary}}
  ```

- Display symbols imported from libraries:

  ```
  rabin2 -i {{path/to/binary}}
  ```

- Display strings contained in the binary:

  ```
  rabin2 -z {{path/to/binary}}
  ```

- Display the output in JSON:

  ```
  rabin2 -j -I {{path/to/binary}}
  ```

radare2

A set of reverse engineering tools.

More information: https://radare.gitbooks.io/radare2_book/

- Open a file in write mode without parsing the file format headers:

```
radare2 -nw {{path/to/binary}}
```

- Debug a program:

```
radare2 -d {{path/to/binary}}
```

- Run a script before entering the interactive CLI:

```
radare2 -i {{path/to/script.r2}} {{path/to/binary}}
```

- Show help text for any command in the interactive CLI:

```
> {{radare2_command}}?
```

- Run a shell command from the interactive CLI:

```
> !{{shell_command}}
```

- Dump raw bytes of current block to a file:

```
> pr > {{path/to/file.bin}}
```

rails db

Various database-related subcommands for Ruby on Rails.

- Create databases, load the schema, and initialize with seed data:

  ```
  rails db:setup
  ```

- Access the database console:

  ```
  rails db
  ```

- Create the databases defined in the current environment:

  ```
  rails db:create
  ```

- Destroy the databases defined in the current environment:

  ```
  rails db:drop
  ```

- Run pending migrations:

  ```
  rails db:migrate
  ```

- View the status of each migration file:

  ```
  rails db:migrate:status
  ```

- Rollback the last migration:

  ```
  rails db:rollback
  ```

- Fill the current database with data defined in `db/seeds.rb`:

  ```
  rails db:seed
  ```

rails destroy

Destroy Rails resources.

More information: https://guides.rubyonrails.org/command_line.html#bin-rails-destroy.

- List all available generators to destroy:

  ```
  rails destroy
  ```

- Destroy a model named Post:

  ```
  rails destroy model {{Post}}
  ```

- Destroy a controller named Posts:

  ```
  rails destroy controller {{Posts}}
  ```

- Destroy a migration that creates Posts:

  ```
  rails destroy migration {{CreatePosts}}
  ```

- Destroy a scaffold for a model named Post:

  ```
  rails destroy scaffold {{Post}}
  ```

rails generate

Generate new Rails templates in an existing project.

More information: https://guides.rubyonrails.org/command_line.html#bin-rails-generate.

- List all available generators:

  ```
  rails generate
  ```

- Generate a new model named Post with attributes title and body:

  ```
  rails generate model {{Post}} {{title:string}} {{body:text}}
  ```

- Generate a new controller named Posts with actions index, show, new and create:

  ```
  rails generate controller {{Posts}} {{index}} {{show}}
  {{new}} {{create}}
  ```

- Generate a new migration that adds a category attribute to an existing model called Post:

  ```
  rails generate migration {{AddCategoryToPost}}
  {{category:string}}
  ```

- Generate a scaffold for a model named Post, predefining the attributes title and body:

  ```
  rails generate scaffold {{Post}} {{title:string}}
  {{body:text}}
  ```

rails routes

List routes in a Rails application.

More information: https://guides.rubyonrails.org/routing.html.

- List all routes:

  ```
  rails routes
  ```

- List all routes in an expanded format:

  ```
  rails routes --expanded
  ```

- List routes partially matching URL helper method name, HTTP verb, or URL path:

  ```
  rails routes -g {{posts_path|GET|/posts}}
  ```

- List routes that map to a specified controller:

  ```
  rails routes -c {{posts|Posts|Blogs::PostsController}}
  ```

rails

A server-side MVC framework written in Ruby.

More information: https://guides.rubyonrails.org/command_line.html

- Create a new rails project:

  ```
  rails new "{{project_name}}"
  ```

- Start local server for current project on port 3000:

  ```
  rails server
  ```

- Start local server for current project on a specified port:

  ```
  rails server -p "{{port}}"
  ```

- Open console to interact with application from command-line:

  ```
  rails console
  ```

- Check current version of rails:

  ```
  rails --version
  ```

rainbowstream

Terminal-based Twitter client supporting realtime tweetstream, trends, sending, search, favorites and user management.

Online help with **h**, up and down arrows for history, tab to auto-complete and 2-tab for suggestion.

More information: https://github.com/orakaro/rainbowstream.

- Open rainbowstream:

  `rainbow stream`

- Show your timeline (optional number of tweets to display, default is 5):

  `home [{{num_of_last_tweets}}]`

- Show profile of a given user:

  `whois @{{user}}`

- Tweet the message as-is:

  `t {{message}}`

- Retweet the tweet with given id (id is beside the time):

  `rt {{tweet_id}}`

- Favorite the tweet with given id:

  `fav {{tweet_id}}`

- Perform a search for a given word (with or without hashtag):

  `s {{word}}`

ranger

Console file manager with VI key bindings.

More information: https://github.com/ranger/ranger.

- Launch ranger:

  ```
  ranger
  ```

- Show only directories:

  ```
  ranger --show-only-dirs
  ```

- Change the configuration directory:

  ```
  ranger --confdir={{path/to/directory}}
  ```

- Change the data directory:

  ```
  ranger --datadir={{path/to/directory}}
  ```

- Print CPU usage statistics on exit:

  ```
  ranger --profile
  ```

rapper

The Raptor RDF parsing utility.

Part of the Raptor RDF Syntax Library.

More information: http://librdf.org/raptor/rapper.html.

- Convert an RDF/XML document to Turtle:

  ```
  rapper -i rdfxml -o turtle {{file}}
  ```

- Count the number of triples in a Turtle file:

  ```
  rapper -i turtle -c {{file}}
  ```

rar

The RAR archiver. Supports multi-volume archives that can be optionally self-extracting.

- Archive 1 or more files:

```
rar a {{path/to/archive_name.rar}} {{path/to/file1}} {{path/to/file2}} {{path/to/file3}}
```

- Archive a directory:

```
rar a {{path/to/archive_name.rar}} {{path/to/directory}}
```

- Split the archive into parts of equal size (50 M):

```
rar a -v{{50M}} -R {{path/to/archive_name.rar}} {{path/to/file_or_directory}}
```

- Password protect the resulting archive:

```
rar a -p{{password}} {{path/to/archive_name.rar}} {{path/to/file_or_directory}}
```

- Encrypt file data and headers with password:

```
rar a -hp{{password}} {{path/to/archive_name.rar}} {{path/to/file_or_directory}}
```

- Use a specific compression level (0-5):

```
rar a -m{{compression_level}} {{path/to/archive_name.rar}} {{path/to/file_or_directory}}
```

rbac-lookup

Find roles and cluster roles attached to any user, service account or group name in your Kubernetes cluster.

More information: https://github.com/reactiveops/rbac-lookup.

- View all RBAC bindings:

  ```
  rbac-lookup
  ```

- View RBAC bindings that match a given expression:

  ```
  rbac-lookup {{search_term}}
  ```

- View all RBAC bindings along with the source role binding:

  ```
  rbac-lookup -owide
  ```

- View all RBAC bindings filtered by subject:

  ```
  rbac-lookup -k {{user|group|serviceaccount}}
  ```

- View all RBAC bindings along with IAM roles (if you are using GKE):

  ```
  rbac-lookup --gke
  ```

rbash

Restricted Bash shell, equivalent to `bash --restricted`.

Does not permit changing the working directory, redirecting command output, or modifying environment variables, among other things.

See also `histexpand` for history expansion.

More information: https://www.gnu.org/software/bash/manual/html_node/The-Restricted-Shell.

- Start an interactive shell session:

  ```
  rbash
  ```

- Execute a command and then exit:

  ```
  rbash -c "{{command}}"
  ```

- Execute a script:

  ```
  rbash {{path/to/script.sh}}
  ```

- Execute a script, printing each command before executing it:

  ```
  rbash -x {{path/to/script.sh}}
  ```

- Execute commands from a script, stopping at the first error:

  ```
  rbash -e {{path/to/script.sh}}
  ```

- Read and execute commands from stdin:

  ```
  rbash -s
  ```

rbenv

A tool to easily install R uby versions and manage application environments.

More information: https://github.com/rbenv/rbenv.

- Install a R uby version:

  `rb env install {{v ersion}}`

- Display a list of the latest stable versions for each R uby:

  `rb env install - - list`

- Display a list of installed R uby versions:

  `rb env v ersions`

- U se a specific R uby version across the whole system:

  `rb env g lob al {{v ersion}}`

- U se a specific R uby version for an application/proj ect directory:

  `rb env local {{v ersion}}`

- Display the currently selected R uby version:

  `rb env v ersion`

- U ninstall a R uby version:

  `rb env u ninstall {{v ersion}}`

- Display all R uby versions that contain the specified executable:

  `rb env w hence {{execu tab le}}`

rbt

RBTools is a set of command-line tools for working with Review Board and RB Commons.

More information: https://www.reviewboard.org/docs/rbtools/dev/.

- Post changes to Review Board:

  `rbt post {{change_number}}`

- Display the diff that will be sent to Review Board:

  `rbt diff`

- Land a change in a local branch or on a review request:

  `rbt land {{branch_name}}`

- Patch your tree with a change on a review request:

  `rbt patch {{review_request_id}}`

- Set up RBTool to talk to a repository:

  `rbt setup-repo`

rclone

CLI program to copy/sync/move files and directories to and from many cloud services.

More information: https://rclone.org.

- List contents of a directory on an rclone remote:

  ```
  rclone lsf {{remote_name}}:{{path/to/directory}}
  ```

- Copy file or directory from local source to remote destination:

  ```
  rclone copy {{path/to/source_file_or_directory}} {{remote_name}}:{{path/to/destination_directory}}
  ```

- Copy file or directory from remote source to local destination:

  ```
  rclone copy {{remote_name}}:{{path/to/source_file_or_directory}} {{path/to/destination_directory}}
  ```

- Sync local source to remote destination, changing the destination only:

  ```
  rclone sync {{path/to/file_or_directory}} {{remote_name}}:{{path/to/directory}}
  ```

- Move file or directory from local source to remote destination:

  ```
  rclone move {{path/to/file_or_directory}} {{remote_name}}:{{path/to/directory}}
  ```

- Delete remote file or directory (use `--dry-run` to test, remove it to actually delete):

  ```
  rclone --dry-run delete {{remote_name}}:{{path/to/file_or_directory}}
  ```

- Mount rclone remote (experimental):

  ```
  rclone mount {{remote_name}}:{{path/to/directory}} {{path/to/mount_point}}
  ```

- Unmount rclone remote if CTRL-C fails (experimental):

  ```
  fusermount -u {{path/to/mount_point}}
  ```

rdfind

Find files with duplicate content and get rid of them.

More information: https://rdfind.pauldreik.se.

- Identify all duplicates in a given directory and output a summary:

```
rdfind -dryrun true {{path/to/directory}}
```

- Replace all duplicates with hardlinks:

```
rdfind -makehardlinks true {{path/to/directory}}
```

- Replace all duplicates with symlinks/soft links:

```
rdfind -makesymlinks true {{path/to/directory}}
```

- Delete all duplicates and do not ignore empty files:

```
rdfind -deleteduplicates true -ignoreempty false {{path/to/directory}}
```

react-native start

Command-line tools to start the React Native server.

More information: https://github.com/react-native-community/cli/blob/master/docs/commands.md#_start

- Start the server that communicates with connected devices:

  ```
  react-native start
  ```

- Start the metro bundler with a clean cache:

  ```
  react-native start --reset-cache
  ```

- Start the server in a custom port (defaults to 8081):

  ```
  react-native start --port {{3000}}
  ```

- Start the server in verbose mode:

  ```
  react-native start --verbose
  ```

- Specify the maximum number of workers for transforming files (default is the number of CPU cores):

  ```
  react-native start --max-workers {{count}}
  ```

- Disable interactive mode:

  ```
  react-native start --no-interactive
  ```

react-native

A framework for building native apps with React.

More information: https://reactnative.dev.

- Initialize a new React Native project in a directory of the same name:

  ```
  react-native init {{project_name}}
  ```

- Start the metro bundler:

  ```
  react-native start
  ```

- Start the metro bundler with a clean cache:

  ```
  react-native start --reset-cache
  ```

- Build the current application and start it on a connected Android device or emulator:

  ```
  react-native run-android
  ```

- Build the current application and start it on an iOS simulator:

  ```
  react-native run-ios
  ```

- Build the current application in `release` mode and start it on a connected Android device or emulator:

  ```
  react-native run-android --variant={{release}}
  ```

- Start `logkitty` and print logs to stdout:

  ```
  react-native log-android
  ```

- Start `tail system.log` for an iOS simulator and print logs to stdout:

  ```
  react-native log-ios
  ```

read

BASH builtin for retrieving data from standard input.

- Store data that you type from the keyboard:

  ```
  read {{variable}}
  ```

- Store each of the next lines you enter as values of an array:

  ```
  read -a {{array}}
  ```

- Specify the number of maximum characters to be read:

  ```
  read -n {{character_count}} {{variable}}
  ```

- Use a specific character as a delimiter instead of a new line:

  ```
  read -d {{new_delimiter}} {{variable}}
  ```

- Do not let backslash () act as an escape character:

  ```
  read -r {{variable}}
  ```

- Display a prompt before the input:

  ```
  read -p "{{Enter your input here:}}" {{variable}}
  ```

- Do not echo typed characters (silent mode):

  ```
  read -s {{variable}}
  ```

- Read stdin and perform an action on every line:

  ```
  while read line; do echo "$line"; done
  ```

readlink

Follow symlinks and get symlink information.

More information: https://www.gnu.org/software/coreutils/readlink.

- Get the actual file to which the symlink points:

  ```
  readlink {{filename}}
  ```

- Get the absolute path to a file:

  ```
  readlink -f {{filename}}
  ```

realpath

Display the resolved absolute path for a file or directory.

More information: https://www.gnu.org/software/coreutils/realpath.

- Display the absolute path for a file or directory:

```
realpath {{path/to/file_or_directory}}
```

- Require all path components to exist:

```
realpath --canonicalize-existing {{path/to/file_or_directory}}
```

- Resolve "..." components before symlinks:

```
realpath --logical {{path/to/file_or_directory}}
```

- Disable symlink expansion:

```
realpath --no-symlinks {{path/to/file_or_directory}}
```

- Suppress error messages:

```
realpath --quiet {{path/to/file_or_directory}}
```

recsel

Print records from a recfile: a human-editable, plain text database.

More information: https://www.gnu.org/software/recutils/manual/recutils.html.

- Extract name and version field:

```
recsel -p name,version {{data.rec}}
```

- Use "~" to match a string with a given regular expression:

```
recsel -e "{{field_name}} ~ '{{regular_expression}}' {{data.rec}}"
```

- Use a predicate to match a name and a version:

```
recsel -e "name ~ '{{regular_expression}}' && version ~ '{{regular_expression}}'" {{data.rec}}
```

rector

An automated tool for updating and refactoring PHP 5.3+ code.

More information: https://github.com/rectorphp/rector.

- Process a specific directory:

  ```
  rector process {{path/to/directory}}
  ```

- Process a directory without applying changes (dry run):

  ```
  rector process {{path/to/directory}} --dry-run
  ```

- Process a directory and apply coding standards:

  ```
  rector process {{path/to/directory}} --with-style
  ```

- Display a list of available levels:

  ```
  rector levels
  ```

- Process a directory with a specific level:

  ```
  rector process {{path/to/directory}} --level {{level_name}}
  ```

redis-cli

Opens a connection to a Redis server.

More information: https://redis.io/topics/rediscli.

- Connect to the local server:

  ```
  redis-cli
  ```

- Connect to a remote server on the default port (6379):

  ```
  redis-cli -h {{host}}
  ```

- Connect to a remote server specifying a port number:

  ```
  redis-cli -h {{host}} -p {{port}}
  ```

- Connect to a remote server specifying an URI:

  ```
  redis-cli -u {{uri}}
  ```

- Specify a password:

  ```
  redis-cli -a {{password}}
  ```

- Execute Redis command:

  ```
  redis-cli {{redis_command}}
  ```

- Connect to the local cluster:

  ```
  redis-cli -c
  ```

redis-server

Persistent key-value database.

More information: https://redis.io.

- Start Redis server, using the default port (6379), and write logs to stdout:

  ```
  redis-server
  ```

- Start Redis server, using the default port, as a background process:

  ```
  redis-server --daemonize yes
  ```

- Start Redis server, using the specified port, as a background process:

  ```
  redis-server --port {{port}} --daemonize yes
  ```

- Start Redis server with a custom configuration file:

  ```
  redis-server {{path/to/redis.conf}}
  ```

- Start Redis server with verbose logging:

  ```
  redis-server --loglevel {{warning|notice|verbose|debug}}
  ```

redshift

Adj ust the color temperature of your screen according to your surroundings.

More information: http://j onls.dk/redshift

- T urn on R edshift with 5 7 0 0 K temperature during day and 3 6 0 0 K at night:

  ```
  redshift - t {{5 7 0 0 }}: {{3 6 0 0 }}
  ```

- T urn on R edshift with a manually-specified custom location:

  ```
  redshift - l {{latitu de}}: {{long itu de}}
  ```

- T urn on R edshift with 7 0 % screen brightness during day and 4 0 % brightness at night:

  ```
  redshift - b {{0 . 7 }}: {{0 . 4 }}
  ```

- T urn on R edshift with custom gamma levels (between 0 and 1):

  ```
  redshift - g {{red}}: {{g reen}}: {{b lu e}}
  ```

- T urn on R edshift with a constant unchanging color temperature:

  ```
  redshift - O {{tem peratu re}}
  ```

reflac

Recompress FLAC files in-place while preserving metadata.

More information: https://github.com/chungy/reflac

- Recompress a directory of FLAC files:

  ```
  reflac {{path/to/directory}}
  ```

- Enable maximum compression (very slow):

  ```
  reflac --best {{path/to/directory}}
  ```

- Display filenames as they are processed:

  ```
  reflac --verbose {{path/to/directory}}
  ```

- Recurse into subdirectories:

  ```
  reflac --recursive {{path/to/directory}}
  ```

- Preserve file modification times:

  ```
  reflac --preserve {{path/to/directory}}
  ```

reflex

Tool to watch a directory and rerun a command when certain files change.

More information: https://github.com/cespare/reflex

- Rebuild with `make` if any file changes:

  `reflex make`

- Compile and run Go application if any `.go` file changes:

  `reflex --regex='{{\.go$}}' {{go run .}}`

- Ignore a directory when watching for changes:

  `reflex --inverse-regex='{{^dir/}}' {{command}}`

- Run command when reflex starts and restarts on file changes:

  `reflex --start-service=true {{command}}`

- Substitute the filename that changed in:

  `reflex -- echo {}`

renice

Alters the scheduling priority/nicenesses of one or more running processes.

N iceness values range from -2 0 (most favorable to the process) to 1 9 (least favorable to the process).

- Change priority of a running process:

```
renice -n {{niceness_value}} -p {{pid}}
```

- Change priority of all processes owned by a user:

```
renice -n {{niceness_value}} -u {{user}}
```

- Change priority of all processes that belong to a process group:

```
renice -n {{niceness_value}} --pgrp {{process_group}}
```

repren

Multi-pattern string replacement and file renaming tool.

More information: https://github.com/j_levy/repren

- Do a dry-run renaming a directory of pngs with a literal string replacement:

```
repren --dry-run --rename --literal --from '{{find_string}}'
--to '{{replacement_string}}' {{*.png}}
```

- Do a dry-run renaming a directory of jpegs with a regular expression:

```
repren --rename --dry-run --from '{{regular_expression}}' --
to '{{replacement_string}}' {{*.jpg}} {{*.jpeg}}
```

- Do a find-and-replace on the contents of a directory of csv files:

```
repren --from '{{([0-9]+) example_string}}' --to
'{{replacement_string \1}}' {{*.csv}}
```

- Do both a find-and-replace and a rename operation at the same time, using a pattern file:

```
repren --patterns {{path/to/patfile.ext}} --full {{*.txt}}
```

- Do a case-insensitive rename:

```
repren --rename --insensitive --patterns {{path/to/
patfile.ext}} *
```

restic

A backup program that aims to be fast, secure and efficient.

More information: https://restic.net.

- Initialize a backup repository in the specified local directory:

  ```
  restic init --repo {{path/to/repository}}
  ```

- Backup a directory to the repository:

  ```
  restic --repo {{path/to/repository}} backup {{path/to/directory}}
  ```

- Show backup snapshots currently stored in the repository:

  ```
  restic --repo {{path/to/repository}} snapshots
  ```

- Restore a specific backup snapshot to a target directory:

  ```
  restic --repo {{path/to/repository}} restore {{latest|snapshot_id}} --target {{path/to/target}}
  ```

- Restore a specific path from a specific backup to a target directory:

  ```
  restic --repo {{path/to/repository}} restore {{snapshot_id}} --target {{path/to/target}} --include {{path/to/restore}}
  ```

- Clean up the repository and keep only the most recent snapshot of each unique backup:

  ```
  restic forget --keep-last 1 --prune
  ```

resume

CLI tool to easily setup a new resume.

More information: https://github.com/j sonresume/resume-cli

- Create a new `resu m e. j son` file in the current working directory:

  ```
  resu m e init
  ```

- V alidate a `resu m e. j son` against schema tests to ensure it complies with the standard:

  ```
  resu m e v alidate
  ```

- E xport a resume locally in a styliz ed H T ML or P DF format:

  ```
  resu m e export {{path/to/htm l_or_pdf}}
  ```

- S tart a web server that serves a local `resu m e. j son`:

  ```
  resu m e serv e
  ```

rev

Reverse a line of text.

- Reverse the text string "hello":

```
echo "hello" | rev
```

- Reverse an entire file and print to stdout:

```
rev {{file}}
```

rg

Ripgrep is a recursive line-oriented CLI search tool.

Aims to be a faster alternative to **`grep`**

More information: https://github.com/BurntSushi/ripgrep

- Recursively search the current directory for a regular expression:

  ```
  rg {{regular_expression}}
  ```

- Search for regular expressions recursively in the current directory, including hidden files and files listed in `.gitignore`

  ```
  rg --no-ignore --hidden {{regular_expression}}
  ```

- Search for a regular expression only in a certain filetype (e.g., html, css, etc.):

  ```
  rg --type {{filetype}} {{regular_expression}}
  ```

- Search for a regular expression only in a subset of directories:

  ```
  rg {{regular_expression}} {{set_of_subdirs}}
  ```

- Search for a regular expression in files matching a glob (e.g., `README.*`):

  ```
  rg {{regular_expression}} --glob {{glob}}
  ```

- Only list matched files (useful when piping to other commands):

  ```
  rg --files-with-matches {{regular_expression}}
  ```

- Show lines that do not match the given regular expression:

  ```
  rg --invert-match {{regular_expression}}
  ```

- Search a literal string pattern:

  ```
  rg --fixed-strings --{{string}}
  ```

rm

Remove files or directories.

More information: https://www.gnu.org/software/coreutils/rm.

- Remove files from arbitrary locations:

```
rm {{path/to/file}} {{path/to/another/file}}
```

- Recursively remove a directory and all its subdirectories:

```
rm -r {{path/to/directory}}
```

- Forcibly remove a directory, without prompting for confirmation or showing error messages:

```
rm -rf {{path/to/directory}}
```

- Interactively remove multiple files, with a prompt before every removal:

```
rm -i {{file(s)}}
```

- Remove files in verbose mode, printing a message for each removed file:

```
rm -v {{path/to/directory/*}}
```

rmdir

Removes a directory.

More information: https://www.gnu.org/software/coreutils/rmdir.

- Remove directory, provided it is empty. Use `rm -r` to remove non-empty directories:

  ```
  rmdir {{path/to/directory}}
  ```

- Remove the target and its parent directories (useful for nested dirs):

  ```
  rmdir -p {{path/to/directory}}
  ```

roave-backward-compatibility-check

A tool that can be used to verify backward compatibility breaks between two versions of a PHP library.

More information: https://github.com/Roave/BackwardCompatibilityCheck

- Check for breaking changes since the last tag:

  ```
  roave-backward-compatibility-check
  ```

- Check for breaking changes since a specific tag:

  ```
  roave-backward-compatibility-check --from={{git_reference}}
  ```

- Check for breaking changes between the last tag and a specific reference:

  ```
  roave-backward-compatibility-check --to={{git_reference}}
  ```

- Check for breaking changes and output to Markdown:

  ```
  roave-backward-compatibility-check --format=markdown > {{results.md}}
  ```

robo

PHP task runner.

More information: https://robo.li/.

- List available commands:

  ```
  robo list
  ```

- Run a specific command:

  ```
  robo {{foo}}
  ```

- Simulate running a specific command:

  ```
  robo --simulate {{foo}}
  ```

roll

Rolls a user-defined dice sequence.

- Roll 3 6-sided dice and sums the results:

  ```
  roll {{3d6}}
  ```

- Roll 1 8-sided die, add 3 and sum the results:

  ```
  roll {{d8 + 3}}
  ```

- Roll 4 6-sided dice, keep the 3 highest results and sum the results:

  ```
  roll {{4d6h3}}
  ```

- Roll 2 12-sided dice 2 times and show every roll:

  ```
  roll --verbose {{2{2d12}}}
  ```

- Roll 2 20-sided dice until the result is bigger than 10:

  ```
  roll "{{2d20>10}}"
  ```

- Roll 2 5-sided dice 3 times and show the total sum:

  ```
  roll --sum-series {{3{2d5}}}
  ```

route

U se route cmd to set the route table.

- Display the information of route table:

  ```
  rou te - n
  ```

- Add route rule:

  ```
  su do rou te add - net {{ip_address}} netm ask
  {{netm ask _address}} g w  {{g w _address}}
  ```

- Delete route rule:

  ```
  su do rou te del - net {{ip_address}} netm ask
  {{netm ask _address}} dev  {{g w _address}}
  ```

rr

Debugging tool designed to record and replay program execution.

More information: https://rr-proj ect.org/

- R ecord an application:

  ```
  rr record {{path/to/b inary - - arg 1  - - arg 2 }}
  ```

- R eplay latest recorded execution:

  ```
  rr replay
  ```

rspec

Behavior-driven development testing framework written in Ruby to test Ruby code.

More information: https://rspec.info.

- Initialise an .rspec config and a spec helper file:

  ```
  rspec --init
  ```

- Run all tests:

  ```
  rspec
  ```

- Run a specific directory of tests:

  ```
  rspec {{path/to/directory}}
  ```

- Run a specific test file:

  ```
  rspec {{path/to/file}}
  ```

- Run multiple test files:

  ```
  rspec {{path/to/file1}} {{path/to/file2}}
  ```

- Run a specific test in a file (e.g. the test starts on line 83):

  ```
  rspec {{path/to/file}}:{{83}}
  ```

- Run specs with a specific seed:

  ```
  rspec --seed {{seed_number}}
  ```

rsstail

`tail` for RSS feeds.

More information: https://github.com/gvalkov/rsstail.py.

- Show the feed of a given URL and wait for new entries appearing at the bottom:

  ```
  rsstail -u {{url}}
  ```

- Show the feed in reverse chronological order (newer at the bottom):

  ```
  rsstail -r -u {{url}}
  ```

- Include publication date and link:

  ```
  rsstail -pl -u {{url}}
  ```

- Set update interval:

  ```
  rsstail -u {{url}} -i {{interval_in_seconds}}
  ```

- Show feed and exit:

  ```
  rsstail -1 -u {{url}}
  ```

rsync

Transfer files either to or from a remote host (not between two remote hosts).

Can transfer single files, or multiple files matching a pattern.

- Transfer file from local to remote host:

```
rsync {{path/to/local_file}} {{remote_host}}:{{path/to/remote_directory}}
```

- Transfer file from remote host to local:

```
rsync {{remote_host}}:{{path/to/remote_file}} {{path/to/local_directory}}
```

- Transfer file in [a]rchive (to preserve attributes) and compressed ([z]ipped) mode with [v]erbose and [h]uman-readable [P]rogress:

```
rsync -azvhP {{path/to/local_file}} {{remote_host}}:{{path/to/remote_directory}}
```

- Transfer a directory and all its children from a remote to local:

```
rsync -r {{remote_host}}:{{path/to/remote_directory}} {{path/to/local_directory}}
```

- Transfer directory contents (but not the directory itself) from a remote to local:

```
rsync -r {{remote_host}}:{{path/to/remote_directory}}/ {{path/to/local_directory}}
```

- Transfer a directory [r]ecursively, in [a]rchive to preserve attributes, resolving contained soft[l]inks, and ignoring already transferred files [u]nless newer:

```
rsync -rauL {{remote_host}}:{{path/to/remote_file}} {{path/to/local_directory}}
```

- Transfer file over SSH and delete local files that do not exist on remote host:

```
rsync -e ssh --delete {{remote_host}}:{{path/to/remote_file}} {{path/to/local_file}}
```

- Transfer file over SSH using a different port than the default and show global progress:

```
rsync -e 'ssh -p {{port}}' --info=progress2 {{remote_host}}:{{path/to/remote_file}} {{path/to/local_file}}
```

rtmpdump

A tool to dump media content streamed over the RTMP protocol.

More information: http://rtmpdump.mplayerhq.hu/

- Download a file:

```
rtmpdump --rtmp {{rtmp://example.com/path/to/video}} -o {{file.ext}}
```

- Download a file from a Flash player:

```
rtmpdump --rtmp {{rtmp://example.com/path/to/video}} --swfVfy {{http://example.com/player}} --flashVer " {{LNX 10,0,32,1
-o {{file.ext}}
```

- Specify connection parameters if they are not detected correctly:

```
rtmpdump --rtmp {{rtmp://example.com/path/to/video}} --app {{app_name}} --playpath {{path/to/video}} -o {{file.ext}}
```

- Download a file from a server that requires a referrer:

```
rtmpdump --rtmp {{rtmp://example.com/path/to/video}} --pageUrl {{http://example.com/webpage}} -o {{file.ext}}
```

rtv

R eddit T erminal V iewer.

U se arrow keys to navigate. R ight and Left to view and return from a submission, respectively.

More information: https://github.com/michael-laz ar/rtv

- O pen the front page:

  `/front`

- O pen a subreddit:

  `/r/{{su b reddit_nam e}}`

- E xpand/collapse comments:

  `[ space]`

- O pen link:

  `o`

- Log in:

  `u`

- O pen the help screen:

  `?`

ruby

Ruby programming language interpreter.

More information: https://www.ruby-lang.org.

- Open an Interactive Ruby Shell (REPL):

  ```
  irb
  ```

- Execute a Ruby script:

  ```
  ruby {{script.rb}}
  ```

- Execute a single Ruby command in the command-line:

  ```
  ruby -e {{command}}
  ```

- Check for syntax errors on a given Ruby script:

  ```
  ruby -c {{script.rb}}
  ```

- Show the version of Ruby you are using:

  ```
  ruby -v
  ```

runit

3 -stage init system.

More information: https://wiki.archlinux.org/index.php/R unit

- S tart runit' s 3 -stage init scheme:

  ```
  ru nit
  ```

- S hut down runit:

  ```
  k ill - - C O N T {{ru nit_pid}}
  ```

runsv

Start and manage a runit service.

More information: https://manpages.ubuntu.com/manpages/latest/man8/runsv.8.html

- Start a runit service as the current user:

```
runsv {{path/to/service}}
```

- Start a runit service as root:

```
sudo runsv {{path/to/service}}
```

runsvchdir

Change the directory `runsvdir` uses by default.

More information: https://manpages.ubuntu.com/manpages/latest/man8/runsvchdir.8.html

- Switch `runsvdir` directories:

  `sudo runsvchdir {{path/to/directory}}`

runsvdir

Run an entire directory of services.

More information: https://manpages.ubuntu.com/manpages/latest/man8/runsvdir.8.html

- Start and manage all services in a directory as the current user:

```
runsvdir {{path/to/services}}
```

- Start and manage all services in a directory as root:

```
sudo runsvdir {{path/to/services}}
```

- Start services in separate sessions:

```
runsvdir -P {{path/to/services}}
```

rustc

The Rust compiler.

Processes, compiles and links Rust language source files.

More information: https://doc.rust-lang.org/rustc.

- Compile a single file:

```
rustc {{file.rs}}
```

- Compile with high optimization:

```
rustc -O {{file.rs}}
```

- Compile with debugging information:

```
rustc -g {{file.rs}}
```

- Compile with architecture-specific optimizations for the current CPU:

```
rustc -C target-cpu=native {{path/to/file.rs}}
```

- Display architecture-specific optimizations for the current CPU:

```
rustc -C target-cpu=native --print cfg
```

- Display target list:

```
rustc --print target-list
```

- Compile for a specific target:

```
rustc --target {{target_triple}} {{path/to/file.rs}}
```

rustfmt

Tool for formatting Rust source code.

More information: https://github.com/rust-lang/rustfmt.

- Format a file, overwriting the original file in-place:

  ```
  rustfmt {{source.rs}}
  ```

- Check a file for formatting and display any changes on the console:

  ```
  rustfmt --check {{source.rs}}
  ```

- Backup any modified files before formatting (the original file is renamed with a `.bk` extension):

  ```
  rustfmt --backup {{source.rs}}
  ```

rustup

Rust toolchain installer.

Install, manage, and update Rust toolchains.

More information: https://github.com/rust-lang/rustup.rs.

- Install the nightly toolchain for your system:

  `rustup install nightly`

- Switch the default toolchain to nightly so that the `cargo` and `rustc` commands will use it:

  `rustup default nightly`

- Use the nightly toolchain when inside the current project, but leave global settings unchanged:

  `rustup override set nightly`

- Update all toolchains:

  `rustup update`

- List installed toolchains:

  `rustup show`

- Run cargo build with a certain toolchain:

  `rustup run {{toolchain_name}} cargo build`

- Open the local rust documentation in the default web browser:

  `rustup doc`

rvm

A tool for easily installing, managing, and working with multiple ruby environments.

More information: https://rvm.io.

- Install one or more space-separated versions of Ruby:

  ```
  rvm install {{version(s)}}
  ```

- Display a list of installed versions:

  ```
  rvm list
  ```

- Use a specific version of Ruby:

  ```
  rvm use {{version}}
  ```

- Set the default Ruby version:

  ```
  rvm --default use {{version}}
  ```

- Upgrade a version of Ruby to a new version:

  ```
  rvm upgrade {{current_version}} {{new_version}}
  ```

- Uninstall a version of Ruby and keep its sources:

  ```
  rvm uninstall {{version}}
  ```

- Remove a version of Ruby and its sources:

  ```
  rvm remove {{version}}
  ```

- Show specific dependencies for your OS:

  ```
  rvm requirements
  ```

S

Web search from the terminal.

- Search for a query on Google(default provider):

  ```
  s {{query}}
  ```

- List all providers:

  ```
  s --list-providers
  ```

- Search for a query with a given provider:

  ```
  s --provider {{provider}} {{query}}
  ```

- Use a specified binary to perform the search query:

  ```
  s --binary "{{binary}} {{arguments}}" {{query}}
  ```

safe

A CLI to interact with HashiCorp Vault.

More information: https://github.com/starkandwayne/safe.

- Add a safe target:

  ```
  safe target {{vault_addr}} {{target_name}}
  ```

- Authenticate the CLI client against the Vault server, using an authentication token:

  ```
  safe auth {{authentication_token}}
  ```

- Print the environment variables describing the current target:

  ```
  safe env
  ```

- Display a tree hierarchy of all reachable keys for a given path:

  ```
  safe tree {{path}}
  ```

- Move a secret from one path to another:

  ```
  safe move {{old/path/to/secret}} {{new/path/to/secret}}
  ```

- Generate a new 2048-bit SSH key-pair and store it:

  ```
  safe ssh {{2048}} {{path/to/secret}}
  ```

- Set non-sensitive keys for a secret:

  ```
  safe set {{path/to/secret}} {{key}}={{value}}
  ```

- Set auto-generated password in a secret:

  ```
  safe gen {{path/to/secret}} {{key}}
  ```

sails

S ails.j s is a realtime enterprise level MV C framework built on top of N ode.j s.

More information: https://sailsj s.com

- S tart S ails:

  ```
  sails lift
  ```

- Create new S ails proj ect:

  ```
  sails new {{proj ectN am e}}
  ```

- G enerate S ails AP I:

  ```
  sails g enerate {{nam e}}
  ```

- G enerate S ails Controller:

  ```
  sails g enerate controller {{nam e}}
  ```

- G enerate S ails Model:

  ```
  sails g enerate m odel {{nam e}}
  ```

salt-call

Invoke salt locally on a salt minion.

More information: https://docs.saltstack.com/ref/cli/salt-call.html.

- Perform a highstate on this minion:

  ```
  salt-call state.highstate
  ```

- Perform a highstate dry-run, compute all changes but don't actually perform them:

  ```
  salt-call state.highstate test=true
  ```

- Perform a highstate with verbose debugging output:

  ```
  salt-call -ldebug state.highstate
  ```

- List this minion's grains:

  ```
  salt-call grains.items
  ```

salt-key

Manages salt minion keys on the salt master.

N eeds to be run on the salt master, likely as root or with sudo.

More information: https://docs.saltstack.com/ref/cli/salt-key.html.

- List all accepted, unaccepted and rej ected minion keys:

  ```
  salt- k ey - L
  ```

- Accept a minion key by name:

  ```
  salt- k ey - a {{M I N I O N _I D }}
  ```

- R ej ect a minion key by name:

  ```
  salt- k ey - r {{M I N I O N _I D }}
  ```

- P rint fingerprints of all public keys:

  ```
  salt- k ey - F
  ```

salt-run

Frontend for executing salt-runners on minions.

More information: https://docs.saltstack.com/ref/cli/salt-run.html.

- Show status of all minions:

  ```
  salt-run manage.status
  ```

- Show all minions which are disconnected:

  ```
  salt-run manage.up
  ```

salt

Execute commands and assert state on remote salt minions.

More information: https://docs.saltstack.com/ref/cli/salt.html.

- List connected minions:

  ```
  salt '*' test.ping
  ```

- Execute a highstate on all connected minions:

  ```
  salt '*' state.highstate
  ```

- Upgrade packages using the OS package manager (apt, yum, brew) on a subset of minions:

  ```
  salt '*.example.com' pkg.upgrade
  ```

- Execute an arbitrary command on a particular minion:

  ```
  salt '{{minion_id}}' cmd.run "ls"
  ```

samtools

Tools for handling high-throughput sequencing (genomics) data.

Used for reading/writing/editing/indexing/viewing of data in SAM/BAM/CRAM format.

- Convert a SAM input file to BAM stream and save to file:

  ```
  samtools view -S -b {{input.sam}} > {{output.bam}}
  ```

- Take input from stdin (-) and print the SAM header and any reads overlapping a specific region to stdout:

  ```
  {{other_command}} | samtools view -h - chromosome:start-end
  ```

- Sort file and save to BAM (the output format is automatically determined from the output file's extension):

  ```
  samtools sort {{input}} -o {{output.bam}}
  ```

- Index a sorted BAM file (creates {{sorted_input.bam.bai}}):

  ```
  samtools index {{sorted_input.bam}}
  ```

- Print alignment statistics about a file:

  ```
  samtools flagstat {{sorted_input}}
  ```

- Count alignments to each index (chromosome / contig):

  ```
  samtools idxstats {{sorted_indexed_input}}
  ```

- Merge multiple files:

  ```
  samtools merge {{output}} {{input1 input2 ...}}
  ```

- Split input file according to read groups:

  ```
  samtools split {{merged_input}}
  ```

sass

Converts SCSS or Sass files to CSS.

More information: https://sass-lang.com/documentation/cli/dart-sass.

- Convert a SCSS or Sass file to CSS and print out the result:

```
sass {{input.scss|input.sass}}
```

- Convert a SCSS or Sass file to CSS and save the result to a file:

```
sass {{input.scss|input.sass}} {{output.css}}
```

- Watch a SCSS or Sass file for changes and output or update the CSS file with same filename:

```
sass --watch {{input.scss|input.sass}}
```

- Watch a SCSS or Sass file for changes and output or update the CSS file with the given filename:

```
sass --watch {{input.scss|input.sass}}:
{{output.css}}
```

satis

The command-line utility for the Satis static Composer repository.

More information: https://github.com/composer/satis.

- Initialise a Satis configuration:

  ```
  satis init {{satis.json}}
  ```

- Add a VCS repository to the Satis configuration:

  ```
  satis add {{repository_url}}
  ```

- Build the static output from the configuration:

  ```
  satis build {{satis.json}} {{path/to/output_directory}}
  ```

- Build the static output by updating only the specified repository:

  ```
  satis build --repository-url {{repository_url}}
  {{satis.json}} {{path/to/output_directory}}
  ```

- Remove useless archive files:

  ```
  satis purge {{satis.json}} {{path/to/output_directory}}
  ```

sbcl

High performance Common Lisp compiler.

More information: http://www.sbcl.org/.

- Start an SBCL interactive shell (REPL):

  ```
  sbcl
  ```

- Execute a Lisp script:

  ```
  sbcl --script {{path/to/script.lisp}}
  ```

sbt

Build tool for Scala and Java projects.

More information: https://www.scala-sbt.org/1.0/docs/

- Start the SBT interactive shell (REPL):

  ```
  sbt
  ```

- Create a new Scala project from an existing Giter8 template hosted on GitHub:

  ```
  sbt new {{scala/hello-world.g8}}
  ```

- Use the specified version of sbt:

  ```
  sbt -sbt-version {{version}}
  ```

- Use a specific jar file as the sbt launcher:

  ```
  sbt -sbt-jar {{path}}
  ```

- List all sbt options:

  ```
  sbt -h
  ```

S C-IM

A curses based, vim-like spreadsheet calculator.

U se hj kl or arrow keys to navigate.

More information: https://github.com/andmarti1 4 2 4 /sc-im

- S tart S C-IM:

  ```
  scim {{filenam e}}. csv
  ```

- E nter a string into the current cell:

  ```
  < or >
  ```

- E nter a numeric constant into the current cell:

  ```
  =
  ```

- E dit string in the current cell:

  ```
  E
  ```

- E dit number in the current cell:

  ```
  e
  ```

- Center align the current cell:

  ```
  |
  ```

scala

Scala application launcher and interactive interpreter.

More information: https://scala-lang.org.

- Start a Scala interactive shell (REPL):

  ```
  scala
  ```

- Start the interpreter with a dependency in the classpath:

  ```
  scala -classpath {{filename.jar}} {{command}}
  ```

- Execute a Scala script:

  ```
  scala {{script.scala}}
  ```

- Execute a .jar program:

  ```
  scala {{filename.jar}}
  ```

- Execute a single Scala command in the command-line:

  ```
  scala -e {{command}}
  ```

scc

Tool written in Go that counts lines of code.

More information: https://github.com/boyter/scc.

- Print lines of code in the current directory:

```
scc
```

- Print lines of code in the target directory:

```
scc {{path/to/directory}}
```

- Display output for every file:

```
scc --by-file
```

- Display output using a specific output format (defaults to `tabular`):

```
scc --format {{tabular|wide|json|csv|cloc-yaml|html|html-table}}
```

- Only count files with specific file extensions:

```
scc --include-ext {{go,java,js}}
```

- Exclude directories from being counted:

```
scc --exclude-dir {{.git,.hg}}
```

- Display output and sort by column (defaults to by files):

```
scc --sort {{files|name|lines|blanks|code|comments|complexity}}
```

- Print help for scc:

```
scc -h
```

scheme

MIT Scheme language interpreter and REPL (interactive shell).

More information: https://www.gnu.org/software/mit-scheme.

- Open an interactive shell (REPL):

  ```
  scheme
  ```

- Run a scheme program (with no REPL output):

  ```
  scheme --quiet < {{script.scm}}
  ```

- Load a scheme program into the REPL:

  ```
  scheme --load {{script.scm}}
  ```

- Load scheme expressions into the REPL:

  ```
  scheme --eval "{{(define foo 'x)}}"
  ```

- Open the REPL in quiet mode:

  ```
  scheme --quiet
  ```

scp

Secure copy.

Copy files between hosts using Secure Copy Protocol over SSH.

More information: https://man.openbsd.org/scp.

- Copy a local file to a remote host:

```
scp {{path/to/local_file}} {{remote_host}}:{{path/to/remote_file}}
```

- Use a specific port when connecting to the remote host:

```
scp -P {{port}} {{path/to/local_file}} {{remote_host}}:{{path/to/remote_file}}
```

- Copy a file from a remote host to a local directory:

```
scp {{remote_host}}:{{path/to/remote_file}} {{path/to/local_directory}}
```

- Recursively copy the contents of a directory from a remote host to a local directory:

```
scp -r {{remote_host}}:{{path/to/remote_directory}} {{path/to/local_directory}}
```

- Copy a file between two remote hosts transferring through the local host:

```
scp -3 {{host1}}:{{path/to/remote_file}} {{host2}}:{{path/to/remote_directory}}
```

- Use a specific username when connecting to the remote host:

```
scp {{path/to/local_file}} {{remote_username}}@{{remote_host}}:{{path/to/remote_directory}}
```

- Use a specific ssh private key for authentication with the remote host:

```
scp -i {{~/.ssh/private_key}} {{local_file}} {{remote_host}}:{{/path/remote_file}}
```

scrapy

Web-crawling framework.

More information: https://scrapy.org.

- Create a project:

  ```
  scrapy startproject {{project_name}}
  ```

- Create a spider (in project directory):

  ```
  scrapy genspider {{spider_name}} {{website_domain}}
  ```

- Edit spider (in project directory):

  ```
  scrapy edit {{spider_name}}
  ```

- Run spider (in project directory):

  ```
  scrapy crawl {{spider_name}}
  ```

- Fetch a webpage as scrapy sees it and print source in stdout:

  ```
  scrapy fetch {{url}}
  ```

- Open a webpage in the default browser as scrapy sees it (disable JavaScript for extra fidelity):

  ```
  scrapy view {{url}}
  ```

- Open scrapy shell for URL, which allows interaction with the page source in python shell (or ipython if available):

  ```
  scrapy shell {{url}}
  ```

scrcpy

Display and control your Android device on a desktop.

More information: https://github.com/Genymobile/scrcpy

- Display a mirror of a connected device:

  ```
  scrcpy
  ```

- Display a mirror of a specific device based on its ID or IP address (find it under the `adb devices` command):

  ```
  scrcpy --serial {{0123456789abcdef|192.168.0.1:5555
  ```

- Start display in fullscreen mode:

  ```
  scrcpy --fullscreen
  ```

- Rotate the display screen. Each incremental value adds a 90 degree counterclockwise rotation:

  ```
  scrcpy --rotation {{0|1|2|3}}
  ```

- Show touches on physical device:

  ```
  scrcpy --show-touches
  ```

- Record display screen:

  ```
  scrcpy --record {{path/to/file.mp4}}
  ```

- Set target directory for pushing files to device by drag and drop (non-APK):

  ```
  scrcpy --push-target {{path/to/directory}}
  ```

screen

Hold a session open on a remote server. Manage multiple windows with a single SSH connection.

- Start a new screen session:

  ```
  screen
  ```

- Start a new named screen session:

  ```
  screen -S {{session_name}}
  ```

- Start a new daemon and log the output to `screenlog.x`

  ```
  screen -dmLS {{session_name}} {{command}}
  ```

- Show open screen sessions:

  ```
  screen -ls
  ```

- Reattach to an open screen:

  ```
  screen -r {{session_name}}
  ```

- Detach from inside a screen:

  ```
  Ctrl+A, D
  ```

- Kill the current screen session:

  ```
  Ctrl+A, K
  ```

- Kill a detached screen:

  ```
  screen -X -S {{session_name}} quit
  ```

screenfetch

Display system information.

More information: https://github.com/KittyKatt/screenFetch.

- Start screenfetch:

```
screenfetch
```

- Take a screenshot (requires 'scrot'):

```
screenfetch -s
```

- Specify distribution logo:

```
screenfetch -A '{{distribution_name}}'
```

- Specify distribution logo and text:

```
screenfetch -D '{{distribution_name}}'
```

- Strip all color:

```
screenfetch -N
```

script

Make a typescript file of a terminal session.

- Start recording in file named "typescript":

  ```
  script
  ```

- Stop recording:

  ```
  exit
  ```

- Start recording in a given file:

  ```
  script {{log file.log}}
  ```

- Append to an existing file:

  ```
  script -a {{log file.log}}
  ```

- Execute quietly without start and done messages:

  ```
  script -q {{log file.log}}
  ```

sd

Intuitive find & replace CLI.

- Trim some whitespace using a regular expression:

```
{{echo ' lorem  ipsum  23  ' }} |  sd ' \s+$ '  ' '
```

- Replace words using capture groups:

```
{{echo ' cargo + nightly watch' }} |  sd ' (\w+)\s+\+(\w+)\s
(\w+) '  ' cmd:  $1 ,  channel:  $2 ,  subcmd:  $3 '
```

- Find and replace in a file printing the result to stdout:

```
sd -p {{' window.fetch' }} {{' fetch' }} {{http.js}}
```

- Find and replace across a project changing each file in place:

```
sd {{' from  " react" ' }} {{' from  " preact" ' }} $(find .  -type f)
```

sdcv

StarDict, a command-line dictionary client.

Dictionaries are provided separately from the client.

More information: https://manned.org/sdcv.

- Start sdcv interactively:

```
sdcv
```

- List installed dictionaries:

```
sdcv --list-dicts
```

- Display a definition from a specific dictionary:

```
sdcv --use-dict {{dictionary_name}} {{search_term}}
```

- Look up a definition with a fuzzy search:

```
sdcv {{search_term}}
```

- Look up a definition with an exact search:

```
sdcv --exact-search {{search_term}}
```

- Look up a definition and format the output as JSON:

```
sdcv --json {{search_term}}
```

- Search for dictionaries in a specific directory:

```
sdvc --data-dir {{path/to/directory}} {{search_term}}
```

sdiff

Compare the differences between and optionally merge 2 files.

- Compare 2 files:

  ```
  sdiff {{path/to/file1 }} {{path/to/file2 }}
  ```

- Compare 2 files, ignoring all tabs and whitespace:

  ```
  sdiff - W {{path/to/file1 }} {{path/to/file2 }}
  ```

- Compare 2 files, ignoring whitespace at the end of lines:

  ```
  sdiff - Z {{path/to/file1 }} {{path/to/file2 }}
  ```

- Compare 2 files in a case-insensitive manner:

  ```
  sdiff - i {{path/to/file1 }} {{path/to/file2 }}
  ```

- Compare and then merge, writing the output to a new file:

  ```
  sdiff - o {{path/to/m erg ed_file}} {{path/to/file1 }} {{path/to/
  file2 }}
  ```

sdk

Tool for managing parallel versions of multiple Software Development Kits.

Supports Java, Groovy, Scala, Kotlin, Gradle, Maven, Vert.x and many others.

More information: https://sdkman.io/usage.

- Install a specific version of Gradle:

  ```
  sdk install {{gradle}} {{gradle_version}}
  ```

- Switch to a specific version of Gradle:

  ```
  sdk use {{gradle}} {{gradle_version}}
  ```

- Check current Gradle version:

  ```
  sdk current {{gradle}}
  ```

- List all Software Development Kits available to install:

  ```
  sdk list
  ```

- Update Gradle to the latest version:

  ```
  sdk upgrade {{gradle}}
  ```

- Uninstall a particular version of Gradle:

  ```
  sdk rm {{gradle}} {{gradle_version}}
  ```

sdkmanager

Tool to install packages for the Android SDK.

More information: https://developer.android.com/studio/command-line/sdkmanager.

- List available packages:

  ```
  sdkmanager --list
  ```

- Install a package:

  ```
  sdkmanager {{package}}
  ```

- Update every installed package:

  ```
  sdkmanager --update
  ```

- Uninstall a package:

  ```
  sdkmanager --uninstall {{package}}
  ```

secrethub

A tool to keep secrets out of config files.

More information: https://secrethub.io.

- Print a secret to stdout:

  ```
  secrethub read {{path/to/secret}}
  ```

- Generate a random value and store it as a new or updated secret:

  ```
  secrethub generate {{path/to/secret}}
  ```

- Store a value from the clipboard as a new or updated secret:

  ```
  secrethub write --clip {{path/to/secret}}
  ```

- Store a value supplied on stdin as a new or updated secret:

  ```
  echo "{{secret_value}}" | secrethub write {{path/to/secret}}
  ```

- Audit a repository or secret:

  ```
  secrethub audit {{path/to/repo_or_secret}}
  ```

security-checker

Check if a PHP application uses dependencies with known security vulnerabilities.

More information: https://github.com/sensiolabs/security-checker.

- Look for security issues in the project dependencies (based on the `composer.lock` file in the current directory):

  ```
  security-checker security:check
  ```

- Use a specific `composer.lock` file:

  ```
  security-checker security:check {{path/to/composer.lock}}
  ```

- Return results as a JSON object:

  ```
  security-checker security:check --format=json
  ```

sed

Edit text in a scriptable manner.

More information: https://man.archlinux.org/man/sed.1.

- Replace the first occurrence of a regular expression in each line of a file, and print the result:

```
sed 's/{{regular_expression}}/{{replace}}/' {{filename}}
```

- Replace all occurrences of an extended regular expression in a file, and print the result:

```
sed -r 's/{{regular_expression}}/{{replace}}/g' {{filename}}
```

- Replace all occurrences of a string in a file, overwriting the file (i.e. in-place):

```
sed -i 's/{{find}}/{{replace}}/g' {{filename}}
```

- Replace only on lines matching the line pattern:

```
sed '/{{line_pattern}}/s/{{find}}/{{replace}}/' {{filename}}
```

- Delete lines matching the line pattern:

```
sed '/{{line_pattern}}/d' {{filename}}
```

- Print the first 11 lines of a file:

```
sed 11q {{filename}}
```

- Apply multiple find-replace expressions to a file:

```
sed -e 's/{{find}}/{{replace}}/' -e 's/{{find}}/{{replace}}/' {{filename}}
```

- Replace separator/ by any other character not used in the find or replace patterns, e.g., #:

```
sed 's#{{find}}#{{replace}}#' {{filename}}
```

sendmail

Send email from the command-line.

- Send a message with the content of `message.txt` to the mail directory of local user `username`:

  ```
  sendmail {{username}} < {{message.txt}}
  ```

- Send an email from you@yourdomain.com (assuming the mail server is configured for this) to test@gmail.com containing the message in `message.txt`

  ```
  sendmail -f {{you@yourdomain.com}} {{test@gmail.com}} < {{message.txt}}
  ```

- Send an email from you@yourdomain.com (assuming the mail server is configured for this) to test@gmail.com containing the file `file.zip`

  ```
  sendmail -f {{you@yourdomain.com}} {{test@gmail.com}} < {{file.zip}}
  ```

seq

Output a sequence of numbers to stdout.

More information: https://www.gnu.org/software/coreutils/seq.

- Sequence from 1 to 10:

```
seq 10
```

- Every 3rd number from 5 to 20:

```
seq 5 3 20
```

- Separate the output with a space instead of a newline:

```
seq -s " " 5 3 20
```

- Format output width to a minimum of 4 digits padding with zeros as necessary:

```
seq -f "%04g" 5 3 20
```

seq ueliz e

P romise-based N ode.j s O R M for P ostgres, MyS Q L, MariaDB , S Q Lite and Microsoft S Q L S erver.

More information: https://seq ueliz e.org/

- Create a model with 3 fields and a migration file:

```
seq u elize m odel: g enerate - - nam e {{tab le_nam e}} - - attrib u tes
{{field1 : integ er, field2 : string , field3 : b oolean}}
```

- R un the migration file:

```
seq u elize db : m ig rate
```

- R evert all migrations:

```
seq u elize db : m ig rate: u ndo: all
```

- Create a seed file with the specified name to populate the database:

```
seq u elize seed: g enerate - - nam e {{seed_filenam e}}
```

- P opulate database using all seed files:

```
seq u elize db : seed: all
```

serverless

Toolkit for deploying and operating serverless architectures on AWS, Google Cloud, Azure and IBM OpenWhisk.

Commands can be run either using the `serverless` command or its alias `sls`.

More information: https://serverless.com/.

- Create a serverless project:

  ```
  serverless create
  ```

- Create a serverless project from a template:

  ```
  serverless create --template {{template_name}}
  ```

- Deploy to a cloud provider:

  ```
  serverless deploy
  ```

- Display information about a serverless project:

  ```
  serverless info
  ```

- Invoke a deployed function:

  ```
  serverless invoke -f {{function_name}}
  ```

- Follow the logs for a project:

  ```
  serverless logs -t
  ```

set

Display, set or unset values of shell attributes and positional parameters.

- Display the names and values of shell variables:

  ```
  set
  ```

- Mark variables that are modified or created for export:

  ```
  set -a
  ```

- Notify of job termination immediately:

  ```
  set -b
  ```

- Set various options, e.g. enable vi style line editing:

  ```
  set -o {{vi}}
  ```

sfdp

Render an image of a **scaled force-directed** network graph from a **graphviz** file.

Layouts: **dot**, **neato**, **twopi**, **circo**, **fdp**, **sfdp**, **osage** & **patchwork**

More information: https://graphviz.org/doc/info/command.html

- Render a `png` image with a filename based on the input filename and output format (uppercase -O):

  ```
  sfdp -T {{png}} -O {{path/to/input.gv}}
  ```

- Render a `svg` image with the specified output filename (lowercase -o):

  ```
  sfdp -T {{svg}} -o {{path/to/image.svg}} {{path/to/input.gv}}
  ```

- Render the output in `ps`, `pdf`, `svgz`, `fig`, `png`, `gif`, `jpg`, `jpeg`, or `dot` format:

  ```
  sfdp -T {{format}} -O {{path/to/input.gv}}
  ```

- Render a `gif` image using stdin and stdout:

  ```
  echo "{{digraph {this -> that}}}" | sfdp -T {{gif}} > {{path/to/image.gif}}
  ```

- Display help:

  ```
  sfdp -?
  ```

sftp

Secure File Transfer Program.

Interactive program to copy files between hosts over SSH.

For non-interactive file transfers, see `scp` or `rsync`.

- Connect to a remote server and enter an interactive command mode:

  ```
  sftp {{remote_user}}@{{remote_host}}
  ```

- Connect using an alternate port:

  ```
  sftp -P {{remote_port}} {{remote_user}}@{{remote_host}}
  ```

- Transfer remote file to the local system:

  ```
  get {{/path/remote_file}}
  ```

- Transfer local file to the remote system:

  ```
  put {{/path/local_file}}
  ```

- Transfer remote directory to the local system recursively (works with `put` too):

  ```
  get -R {{/path/remote_directory}}
  ```

- Get list of files on local machine:

  ```
  lls
  ```

- Get list of files on remote machine:

  ```
  ls
  ```

sh

Bourne shell, the standard command language interpreter.

See also `history` for history expansion.

More information: https://manned.org/sh.

- Start an interactive shell session:

  ```
  sh
  ```

- Execute a command and then exit:

  ```
  sh -c "{{command}}"
  ```

- Execute a script:

  ```
  sh {{path/to/script.sh}}
  ```

- Read and execute commands from stdin:

  ```
  sh -s
  ```

sha1 sum

Calculate SHA1 cryptographic checksums.

More information: https://www.gnu.org/software/coreutils/sha1sum

- Calculate the SHA1 checksum for a file:

```
sha1sum {{filename1}}
```

- Calculate SHA1 checksums for multiple files:

```
sha1sum {{filename1}} {{filename2}}
```

- Calculate and save the list of SHA1 checksums to a file:

```
sha1sum {{filename1}} {{filename2}} > {{filename.sha1}}
```

- Read a file of SHA1 sums and verify all files have matching checksums:

```
sha1sum --check {{filename.sha1}}
```

- Only show a message for files for which verification fails:

```
sha1sum --check --quiet {{filename.sha1}}
```

sha2 2 4 sum

Calculate S H A2 2 4 cryptographic checksums.

More information: https://www.gnu.org/software/coreutils/manual/html_ node/sha2 - utilities.html.

- Calculate the S H A2 2 4 checksum for a file:

```
sha2 2 4 su m {{filenam e1 }}
```

- Calculate S H A2 2 4 checksums for multiple files:

```
sha2 2 4 su m {{filenam e1 }} {{filenam e2 }}
```

- Calculate and save the list of S H A2 2 4 checksums to a file:

```
sha2 2 4 su m {{filenam e1 }} {{filenam e2 }} > {{filenam e. sha2 2 4 }}
```

- R ead a file of S H A2 2 4 sums and verify all files have matching checksums:

```
sha2 2 4 su m - - check {{filenam e. sha2 2 4 }}
```

- O nly show a message for files for which verification fails:

```
sha2 2 4 su m - - check - - q u iet {{filenam e. sha2 2 4 }}
```

sha2 5 6 sum

Calculate S H A2 5 6 cryptographic checksums.

More information: https://www.gnu.org/software/coreutils/manual/html_ node/sha2 - utilities.html.

- Calculate the S H A2 5 6 checksum for a file:

```
sha2 5 6 su m {{filenam e1 }}
```

- Calculate S H A2 5 6 checksums for multiple files:

```
sha2 5 6 su m {{filenam e1 }} {{filenam e2 }}
```

- Calculate and save the list of S H A2 5 6 checksums to a file:

```
sha2 5 6 su m {{filenam e1 }} {{filenam e2 }} > {{filenam e. sha2 5 6 }}
```

- R ead a file of S H A2 5 6 sums and verify all files have matching checksums:

```
sha2 5 6 su m - - check {{filenam e. sha2 5 6 }}
```

- O nly show a message for files for which verification fails:

```
sha2 5 6 su m - - check - - q u iet {{filenam e. sha2 5 6 }}
```

sha3 8 4 sum

Calculate S H A3 8 4 cryptographic checksums.

More information: https://www.gnu.org/software/coreutils/manual/html_ node/sha2 - utilities.html.

- Calculate the S H A3 8 4 checksum for a file:

```
sha3 8 4 su m {{filenam e1 }}
```

- Calculate S H A3 8 4 checksums for multiple files:

```
sha3 8 4 su m {{filenam e1 }} {{filenam e2 }}
```

- Calculate and save the list of S H A3 8 4 checksums to a file:

```
sha3 8 4 su m {{filenam e1 }} {{filenam e2 }} > {{filenam e. sha3 8 4 }}
```

- R ead a file of S H A3 8 4 sums and verify all files have matching checksums:

```
sha3 8 4 su m - - check {{filenam e. sha3 8 4 }}
```

- O nly show a message for files for which verification fails:

```
sha3 8 4 su m - - check - - q u iet {{filenam e. sha3 8 4 }}
```

sha512sum

Calculate SHA512 cryptographic checksums.

More information: https://www.gnu.org/software/coreutils/manual/html_node/sha2-utilities.html.

- Calculate the SHA512 checksum for a file:

```
sha512sum {{filename}}
```

- Calculate SHA512 checksums for multiple files:

```
sha512sum {{filename1}} {{filename2}}
```

- Calculate and save the list of SHA512 checksums to a file:

```
sha512sum {{filename1}} {{filename2}} > {{filename.sha512}}
```

- Read a file of SHA512 sums and verify all files have matching checksums:

```
sha512sum --check {{filename.sha512}}
```

- Only show a message for files for which verification fails:

```
sha512sum --check --quiet {{filename.sha512}}
```

shards

Dependency management tool for the Crystal language.

More information: https://crystal-lang.org/reference/the_shards_command

- Create a skeleton `shard.yml` file:

  ```
  shards init
  ```

- Install dependencies from a `shard.yml` file:

  ```
  shards install
  ```

- Update all dependencies:

  ```
  shards update
  ```

- List all installed dependencies:

  ```
  shards list
  ```

- List version of dependency:

  ```
  shards version {{path/to/dependency_directory}}
  ```

shasum

Calculate or check cryptographic SHA checksums.

- Calculate the SHA1 checksum for a file:

  ```
  shasum {{filename}}
  ```

- Calculate the SHA256 checksum for a file:

  ```
  shasum --algorithm 256 {{filename}}
  ```

- Calculate the SHA512 checksum for multiple files:

  ```
  shasum --algorithm 512 {{filename1}} {{filename2}}
  ```

- Calculate and save the list of SHA256 checksums to a file:

  ```
  shasum --algorithm 256 {{filename1}} {{filename2}} > {{filename.sha256}}
  ```

- Check a file with a list of sums against the directory's files:

  ```
  shasum --check {{list_file}}
  ```

- Check a list of sums and only show a message for files for which verification fails:

  ```
  shasum --check --quiet {{list_file}}
  ```

- Calculate the SHA1 checksum from stdin:

  ```
  {{command}} | shasum
  ```

shc

Generic shell script compiler.

- Compile a shell script:

  ```
  shc -f {{script}}
  ```

- Compile a shell script and specify an output binary file:

  ```
  shc -f {{script}} -o {{binary}}
  ```

- Compile a shell script and set an expiration date for the executable:

  ```
  shc -f {{script}} -e {{dd/mm/yyyy}}
  ```

- Compile a shell script and set a message to display upon expiration:

  ```
  shc -f {{script}} -e {{dd/mm/yyyy}} -m "{{Please contact your provider}}"
  ```

shellcheck

S hell script static analysis tool.

Check shell scripts for errors, usage of deprecated/insecure features, and bad practices.

More information: https://www.shellcheck.net.

- Check a shell script:

```
shellcheck {{path/to/script.sh}}
```

- Check a shell script interpreting it as the specified shell dialect (overrides the shebang at the top of the script):

```
shellcheck --shell {{sh|bash|dash|ksh}} {{path/to/script.sh}}
```

- Ignore one or more error types:

```
shellcheck --exclude {{SC1009,SC1073}} {{path/to/script.sh
```

- Also check any sourced shell scripts:

```
shellcheck --check-sourced {{path/to/script.sh}}
```

- Display output in the specified format (defaults to `tty`):

```
shellcheck --format {{tty|checkstyle|diff|gcc|json|json1|
quiet}} {{path/to/script.sh}}
```

- E nable one or more optional checks:

```
shellcheck --enable={{add-default-case|avoid-nullary-
conditions}}
```

- List all available optional checks that are disabled by default:

```
shellcheck --list-optional
```

shift

Bash built-in command that shifts the arguments passed to the calling function or script by a specified number of places.

More information: https://www.gnu.org/software/bash/manual/bash.html#_index-shift

- Move arguments by one place dropping the first argument:

```
shift
```

- Move arguments by N places dropping the first N arguments:

```
shift {{N}}
```

shiori

Simple bookmark manager built with Go.

More information: https://github.com/go-shiori/shiori.

- Import bookmarks from HTML Netscape bookmark format file:

  ```
  shiori import {{path/to/bookmarks.html}}
  ```

- Save the specified URL as bookmark:

  ```
  shiori add {{url}}
  ```

- List the saved bookmarks:

  ```
  shiori print
  ```

- Open the saved bookmark in a browser:

  ```
  shiori open {{bookmark_id}}
  ```

- Start the web interface for managing bookmarks at port 8181:

  ```
  shiori serve --port {{8181}}
  ```

shopt

Manage Bash shell options: variables (stored in `$BASHOPTS`) that control behavior specific to the Bash shell.

Generic POSIX shell variables (stored in `$SHELLOPTS`) are managed with the `set` command instead.

- List of all settable options and whether they are set:

  ```
  shopt
  ```

- Set an option:

  ```
  shopt -s {{option_name}}
  ```

- Unset an option:

  ```
  shopt -u {{option_name}}
  ```

- Print a list of all options and their status formatted as runnable `shopt` commands:

  ```
  shopt -p
  ```

- Show help for the command:

  ```
  help shopt
  ```

showfigfonts

Display available figlet fonts.

S ee also: `figlet`.

More information: https://manned.org/showfigfonts.

- Display available fonts:

  ```
  show fig fonts
  ```

- Display available fonts using a specific text:

  ```
  show fig fonts {{inpu t_text}}
  ```

shred

O verwrite files to securely delete data.

More information: https://www.gnu.org/software/coreutils/shred.

- O verwrite a file:

  ```
  shred {{file}}
  ```

- O verwrite a file, leaving z eroes instead of random data:

  ```
  shred - - zero {{file}}
  ```

- O verwrite a file 2 5 times:

  ```
  shred - n2 5 {{file}}
  ```

- O verwrite a file and remove it:

  ```
  shred - - rem ov e {{file}}
  ```

shuf

Generate random permutations.

More information: https://www.gnu.org/software/coreutils/shuf.

- Randomize the order of lines in a file and output the result:

  ```
  shuf {{filename}}
  ```

- Only output the first 5 entries of the result:

  ```
  shuf --head-count={{5}} {{filename}}
  ```

- Write the output to another file:

  ```
  shuf {{filename}} --output={{output_filename}}
  ```

- Generate 3 random numbers in the range 1-10 (inclusive):

  ```
  shuf --head-count={{3}} --input-range={{1-10}} --repeat
  ```

siege

HTTP loadtesting and benchmarking tool.

More information: https://www.joedog.org/siege-manual/

- Test a URL with default settings:

  ```
  siege {{https://example.com}}
  ```

- Test a list of URLs:

  ```
  siege --file {{path/to/url_list.txt}}
  ```

- Test list of URLs in a random order (Simulates internet traffic):

  ```
  siege --internet --file {{path/to/url_list.txt}}
  ```

- Benchmark a list of URLs (without waiting between requests):

  ```
  siege --benchmark --file {{path/to/url_list.txt}}
  ```

- Set the amount of concurrent connections:

  ```
  siege --concurrent={{50}} --file {{path/to/url_list.txt}}
  ```

- Set how long for the siege to run for:

  ```
  siege --time={{30s}} --file {{path/to/url_list.txt}}
  ```

sindresorhus

Sindre Sorhus's personal CLI.

More information: https://github.com/sindresorhus/sindresorhus-cli.

- Start Sindre's interactive CLI:

  ```
  sindresorhus
  ```

singularity

Manage Singularity containers and images.

- Download a remote image from Sylabs Cloud:

  ```
  singularity pull --name {{image.sif}} {{library://godlovedc/funny/lolcow:latest}}
  ```

- Rebuild a remote image using latest Singularity image format:

  ```
  singularity build {{image.sif}} {{docker://godlovedc/lolcow}}
  ```

- Start a container from an image and get a shell inside of it:

  ```
  singularity shell {{image.sif}}
  ```

- Start a container from an image and run a command:

  ```
  singularity exec {{image.sif}} {{command}}
  ```

- Start a container from an image and execute the internal runscript:

  ```
  singularity run {{image.sif}}
  ```

- Build a singularity image from a recipe file:

  ```
  sudo singularity build {{image.sif}} {{recipe}}
  ```

sk

Fuzzy finder written in Rust.

Similar to `fzf`.

More information: https://github.com/lotabout/skim.

- Start skim on all files in the specified directory:

```
find {{path/to/directory}} -type f | sk
```

- Start skim for running processes:

```
ps aux | sk
```

- Start skim with a specified query:

```
sk --query "{{query}}"
```

- Select multiple files with `Shift + Tab` and write to a file:

```
find {{path/to/directory}} -type f | sk --multi > {{filename}}
```

skaffold

A tool that facilitates continuous development for Kubernetes applications.

More information: https://skaffold.dev.

- Build the artifacts:

  ```
  skaffold build -f {{skaffold.yaml}}
  ```

- Build and deploy your app every time your code changes:

  ```
  skaffold dev -f {{skaffold.yaml}}
  ```

- Run a pipeline file:

  ```
  skaffold run -f {{skaffold.yaml}}
  ```

- Run a diagnostic on Skaffold:

  ```
  skaffold diagnose -f {{skaffold.yaml}}
  ```

- Deploy the artifacts:

  ```
  skaffold deploy -f {{skaffold.yaml}}
  ```

skicka

Manage your G oogle Drive.

More information: https://github.com/google/skicka.

- U pload a file/folder to G oogle Drive:

  ```
  sk ick a  u pload {{path/to/local}} {{path/to/rem ote}}
  ```

- Download a file/folder from G oogle Drive:

  ```
  sk ick a  dow nload {{path/to/rem ote}} {{path/to/local}}
  ```

- List files:

  ```
  sk ick a  ls {{path/to/folder}}
  ```

- S how amount of space used by children folders:

  ```
  sk ick a  du  {{path/to/parent/folder}}
  ```

- Create a folder:

  ```
  sk ick a  m k dir {{path/to/folder}}
  ```

- Delete a file:

  ```
  sk ick a  rm  {{path/to/file}}
  ```

sl

Steam locomotive running through your terminal.

More information: https://github.com/mtoyoda/sl.

- Let a steam locomotive run through your terminal:

  ```
  sl
  ```

- The train burns, people scream:

  ```
  sl -a
  ```

- Let the train fly:

  ```
  sl -F
  ```

- Make the train little:

  ```
  sl -l
  ```

- Let the user exit (CTRL+C):

  ```
  sl -e
  ```

slackcat

Utility for passing files and command output to Slack.

More information: https://github.com/bcicen/slackcat.

- Post a file to Slack:

```
slack cat --channel {{channel_name}} {{path/to/file}}
```

- Post a file to Slack with a custom filename:

```
slack cat --channel {{channel_name}} --filename={{filename}}
{{path/to/file}}
```

- Pipe command output to Slack as a text snippet:

```
{{command}} | slack cat --channel {{channel_name}} --
filename={{snippet_name}}
```

- Stream command output to Slack continuously:

```
{{command}} | slack cat --channel {{channel_name}} --stream
```

sleep

Delay for a specified amount of time.

More information: https://www.gnu.org/software/coreutils/sleep.

- Delay in seconds:

  ```
  sleep {{seconds}}
  ```

- Delay in minutes:

  ```
  sleep {{m inu tes}}m
  ```

- Delay in hours:

  ```
  sleep {{hou rs}}h
  ```

slimrb

Convert Slim files to HTML.

- Convert a Slim file to HTML:

  ```
  slim rb {{input.slim}} {{output.html}}
  ```

- Convert a Slim file and output to prettified HTML:

  ```
  slim rb --pretty {{input.slim}} {{output.html}}
  ```

- Convert a Slim file to ERB:

  ```
  slim rb --erb {{input.slim}} {{output.erb}}
  ```

smartctl

View a disk's SMART data and other information.

More information: https://en.wikipedia.org/wiki/S.M.A.R.T.

- View SMART health summary:

  ```
  sudo smartctl --health {{/dev/sdX}}
  ```

- View device information:

  ```
  sudo smartctl --info {{/dev/sdX}}
  ```

- Begin a short self-test:

  ```
  sudo smartctl --test short {{/dev/sdX}}
  ```

- View current/last self-test status and other SMART capabilities:

  ```
  sudo smartctl --capabilities {{/dev/sdX}}
  ```

- View SMART self-test log (if supported):

  ```
  sudo smartctl --log=selftest {{/dev/sdX}}
  ```

sn

Mono StrongName utility for signing and verifying IL assemblies.

- Generate a new StrongNaming key:

  ```
  sn -k {{path/to/key.snk}}
  ```

- Re-sign an assembly with the specified private key:

  ```
  sn -R {{path/to/assembly.dll}} {{path/to/key_pair.snk}}
  ```

- Show the public key of the private key that was used to sign an assembly:

  ```
  sn -T {{path/to/assembly.exe}}
  ```

- Extract the public key to a file:

  ```
  sn -e {{path/to/assembly.dll}} {{path/to/output.pub}}
  ```

snyk

Find vulnerabilities in your code and remediate risks.

More information: https://snyk.io.

- Log in to your Snyk account:

  ```
  snyk auth
  ```

- Test your code for any known vulnerabilities:

  ```
  snyk test
  ```

- Test a local Docker image for any known vulnerabilities:

  ```
  snyk test --docker {{docker_image}}
  ```

- Record the state of dependencies and any vulnerabilities on snyk.io:

  ```
  snyk monitor
  ```

- Auto patch and ignore vulnerabilities:

  ```
  snyk wizard
  ```

socat

Multipurpose relay (S O cket CAT).

- Listen to a port, wait for an incoming connection and transfer data to S T DIO :

  ```
  socat - T C P - L I S T E N : 8 0 8 0 , fork
  ```

- Create a connection to a host and port, transfer data in S T DIO to connected host:

  ```
  socat - T C P 4 : w w w . exam ple. com : 8 0
  ```

- F orward incoming data of a local port to another host and port:

  ```
  socat T C P - L I S T E N : 8 0 , fork T C P 4 : w w w . exam ple. com : 8 0
  ```

solo

Interact with S olo hardware security keys.

More information: https://github.com/solokeys/solo-python.

- List connected S olos:

  ```
  solo ls
  ```

- U pdate the currently connected S olo' s firmware to the latest version:

  ```
  solo k ey u pdate
  ```

- B link the led of a specific S olo:

  ```
  solo k ey w ink - - serial {{serial_nu m b er}}
  ```

- G enerate random bytes using the currently connected S olo' s secure random number generator:

  ```
  solo k ey rng raw
  ```

- Monitor the serial output of a S olo:

  ```
  solo m onitor {{path/to/serial_port}}
  ```

sonar-scanner

SonarScanner is a generic scanner for SonarQube projects that do not use build tools such as Maven, Gradle, or Ant.

More information: https://docs.sonarqube.org/latest/analysis/scan/sonarscanner/

- Scan a project with configuration file in your project's root directory named `sonar-project.properties`:

  ```
  sonar-scanner
  ```

- Scan a project using configuration file other than `sonar-project.properties`:

  ```
  sonar-scanner -D{{project.settings=myproject.properties}}
  ```

- Print help information:

  ```
  sonar-scanner -h
  ```

- Print debugging information:

  ```
  sonar-scanner -X
  ```

sops

SOPS: Secrets OPerationS.

Tool for managing secrets.

More information: https://github.com/mozilla/sops

- Encrypt a file:

  ```
  sops -e {{path/to/myfile.json}} > {{path/to/myfile.enc.json}}
  ```

- Decrypt a file to the standard output:

  ```
  sops -d {{path/to/myfile.enc.json}}
  ```

- Rotate data keys for a sops file:

  ```
  sops -r {{path/to/myfile.enc.yaml}}
  ```

- Change the extension of the file once encrypted:

  ```
  sops -d --input-type json {{path/to/myfile.enc.json}}
  ```

- Extract keys by naming them, and array elements by numbering them:

  ```
  sops -d --extract '["an_array"][1]' {{path/to/
  myfile.enc.json}}
  ```

- Show the difference between two sops files:

  ```
  diff <(sops -d {{path/to/secret1.enc.yaml}}) <(sops -d
  {{path/to/secret2.enc.yaml}})
  ```

sort

Sort lines of text files.

More information: https://www.gnu.org/software/coreutils/sort.

- Sort a file in ascending order:

  ```
  sort {{path/to/file}}
  ```

- Sort a file in descending order:

  ```
  sort --reverse {{path/to/file}}
  ```

- Sort a file in case-insensitive way:

  ```
  sort --ignore-case {{path/to/file}}
  ```

- Sort a file using numeric rather than alphabetic order:

  ```
  sort --numeric-sort {{path/to/file}}
  ```

- Sort `/etc/passwd` by the 3rd field of each line numerically, using `":"` as a field separator:

  ```
  sort --field-separator={{:}} --key={{3n}} {{/etc/passwd}}
  ```

- Sort a file preserving only unique lines:

  ```
  sort --unique {{path/to/file}}
  ```

- Sort a file, printing the output to the specified output file (can be used to sort a file in-place):

  ```
  sort --output={{path/to/file}} {{path/to/file}}
  ```

- Sort numbers with exponents:

  ```
  sort --general-numeric-sort {{path/to/file}}
  ```

source

Execute commands from a file in the current shell.

- Evaluate contents of a given file:

```
source {{path/to/file}}
```

SOX

Sound eXchange: play, record and convert audio files.

Audio formats are identified by the extension.

More information: http://sox.sourceforge.net.

- Merge two audio files into one:

  ```
  sox -m {{input_audiofile1}} {{input_audiofile2}} {{output_audiofile}}
  ```

- Trim an audio file to the specified times:

  ```
  sox {{input_audiofile}} {{output_audiofile}} trim {{start}} {{end}}
  ```

- Normalize an audio file (adjust volume to the maximum peak level, without clipping):

  ```
  sox --norm {{input_audiofile}} {{output_audiofile}}
  ```

- Reverse and save an audio file:

  ```
  sox {{input_audiofile}} {{output_audiofile}} reverse
  ```

- Print statistical data of an audio file:

  ```
  sox {{input_audiofile}} -n stat
  ```

- Increase the volume of an audio file by 2x:

  ```
  sox -v 2.0 {{input_audiofile}} {{output_audiofile}}
  ```

spark

The Laravel Spark command-line tool.

More information: https://spark.laravel.com.

- Register your API token:

  ```
  spark register {{token}}
  ```

- Display the currently registered API token:

  ```
  spark token
  ```

- Create a new Spark project:

  ```
  spark new {{project_name}}
  ```

- Create a new Spark project with Braintree stubs:

  ```
  spark new {{project_name}} --braintree
  ```

- Create a new Spark project with team based billing stubs:

  ```
  spark new {{project_name}} --team-billing
  ```

spatial

A set of commands for managing and developing SpatialOS projects.

- Run this when you use a project for the first time:

  ```
  spatial worker build
  ```

- Build workers for local deployment on Unity on macOS:

  ```
  spatial worker build --target=development --target=Osx
  ```

- Build workers for local deployment on Unreal on Windows:

  ```
  spatial worker build --target=local --target=Windows
  ```

- Deploy locally:

  ```
  spatial local launch {{launch_config}} --
  snapshot={{snapshot_file}}
  ```

- Launch a local worker to connect to your local deployment:

  ```
  spatial local worker launch {{worker_type}} {{launch_config}}
  ```

- Upload an assembly to use for cloud deployments:

  ```
  spatial cloud upload {{assembly_name}}
  ```

- Launch a cloud deployment:

  ```
  spatial cloud launch {{assembly_name}} {{launch_config}}
  {{deployment_name}}
  ```

- Clean worker directories:

  ```
  spatial worker clean
  ```

speed-test

Test your internet connection speed and ping using speedtest.net from the CLI.

More information: https://github.com/sindresorhus/speed-test.

- Test your internet connection and ping speed:

  ```
  speed-test
  ```

- Output the results as JSON:

  ```
  speed-test --json
  ```

- Output the results in megabytes per second (MBps):

  ```
  speed-test --bytes
  ```

- Output more detailed information:

  ```
  speed-test --verbose
  ```

speedtest-cli

Unofficial command-line interface for testing internet bandwidth using https://speedtest.net.

See also `speedtest` for the official CLI.

More information: https://github.com/sivel/speedtest-cli.

- Run a speed test:

  `speedtest-cli`

- Run a speed test and display values in bytes, instead of bits:

  `speedtest-cli --bytes`

- Run a speed test using `HTTPS` instead of `HTTP`:

  `speedtest-cli --secure`

- Run a speed test without performing download tests:

  `speedtest-cli --no-download`

- Run a speed test and generate an image of the results:

  `speedtest-cli --share`

- List all `speedtest.net` servers, sorted by distance:

  `speedtest-cli --list`

- Run a speed test to a specific speedtest.net server:

  `speedtest-cli --server {{server_id}}`

- Run a speed test and display the results as JSON (suppresses progress information):

  `speedtest-cli --json`

speedtest

Official command-line interface for testing internet bandwidth using https://speedtest.net.

Note: some platforms link `speedtest` to `speedtest-cli`. Some of the examples in this page don't work, see `speedtest-cli`.

More information: https://www.speedtest.net/apps/cli.

- Run a speed test:

  ```
  speedtest
  ```

- Run a speed test and specify the unit of the output:

  ```
  speedtest --unit={{auto-decimal-bits|auto-decimal-bytes|auto-
  binary-bits|auto-binary-bytes}}
  ```

- Run a speed test and specify the output format:

  ```
  speedtest --format={{human-readable|csv|tsv|json|jsonl|jso
  pretty}}
  ```

- Run a speed test and specify the number of decimal points to use (0 to 8, defaults to 2):

  ```
  speedtest --precision={{precision}}
  ```

- Run a speed test and print it's progress (only available for output format `human-readable` and `json`):

  ```
  speedtest --progress={{yes|no}}
  ```

- List all `speedtest.net` servers, sorted by distance:

  ```
  speedtest --servers
  ```

- Run a speed test to a specific `speedtest.net` server:

  ```
  speedtest --server-id={{server_id}}
  ```

spfq uery

Q uery S ender P olicy F ramework records to validate e-mail senders.

More information: https://www.libspf2 .org/

- Check if an IP address is allowed to send an e-mail from the specified e-mail address:

  ```
  spfq u ery - ip {{8 . 8 . 8 . 8 }} - sender {{sender@ exam ple. com }}
  ```

- T urn on debugging output:

  ```
  spfq u ery - ip {{8 . 8 . 8 . 8 }} - sender {{sender@ exam ple. com }} - -
  deb u g
  ```

sphinx-build

S phinx documentation generator.

More information: http://www.sphinx-doc.org/en/master/man/sphinx-build.html.

- B uild documentation:

  ```
  sphinx- b u ild - b {{htm l| epu b | text| latex| m an| . . . }} {{path/to/
  sou rce_dir}} {{path/to/b u ild_dir}}
  ```

- B uild documentations intended for readthedocs.io (req uires the sphinx-rtd-theme pip package):

  ```
  sphinx- b u ild - b {{htm l}} {{path/to/docs_dir}} {{path/to/
  b u ild_dir}}
  ```

spike

A fully featured static website generator written in JavaScript.

More information: https://spike.js.org

- Create a new project using the default template:

  ```
  spike new {{project_name}}
  ```

- Compile your project, watch for changes, and auto-reload the browser:

  ```
  spike watch
  ```

- Compile your project once to the "public" directory:

  ```
  spike compile
  ```

- Remove the output directory:

  ```
  spike clean
  ```

split

Split a file into pieces.

More information: https://www.gnu.org/software/coreutils/split.

- Split a file, each split having 10 lines (except the last split):

```
split -l {{10}} {{filename}}
```

- Split a file into 5 files. File is split such that each split has same size (except the last split):

```
split -n {{5}} {{filename}}
```

- Split a file with 512 bytes in each split (except the last split; use 512k for kilobytes and 512m for megabytes):

```
split -b {{512}} {{filename}}
```

- Split a file with at most 512 bytes in each split without breaking lines:

```
split -C {{512}} {{filename}}
```

sponge

Soak up the input before writing the output file.

More information: https://manned.org/sponge.

- Append file content to the source file:

```
cat {{path/to/file}} | sponge -a {{path/to/file}}
```

- Remove all lines starting with # in a file:

```
grep -v '^{{#}}' {{path/to/file}} | sponge {{path/to/file}}
```

sqlite3

The command-line interface to SQLite 3, which is a self-contained file-based embedded SQL engine.

More information: https://sqlite.org

- Start an interactive shell with a new database:

```
sqlite3
```

- Open an interactive shell against an existing database:

```
sqlite3 {{path/to/database.sqlite3}}
```

- Execute an SQL statement against a database and then exit:

```
sqlite3 {{path/to/database.sqlite3}} '{{SELECT * FROM some_table;}}'
```

sqlmap

Detect and exploit SQL injection flaws.

More information: https://sqlmap.org

- Run sqlmap against a single target URL:

```
python sqlmap.py -u "{{http://www.target.com/vuln.php?id=1
```

- Send data in a POST request (`--data` implies POST request):

```
python sqlmap.py -u "{{http://www.target.com/vuln.php}}" --
data="{{id=1}}"
```

- Change the parameter delimiter (& is the default):

```
python sqlmap.py -u "{{http://www.target.com/vuln.php}}" --
data="{{query=foobar;id=1}}" --param-del="{{;}}"
```

- Select a random `User-Agent` from `./txt/useragents.txt` and use it:

```
python sqlmap.py -u "{{http://www.target.com/vuln.php}}" --
random-agent
```

- Provide user credentials for HTTP protocol authentication:

```
python sqlmap.py -u "{{http://www.target.com/vuln.php}}" --
auth-type {{Basic}} --auth-cred "{{testuser:testpass}}"
```

sqsc

A command-line AWS Simple Queue Service client.

More information: https://github.com/yongfei25/sqsc

- List all queues:

```
sqsc lq {{queue_prefix}}
```

- List all messages in a queue:

```
sqsc ls {{queue_name}}
```

- Copy all messages from one queue to another:

```
sqsc cp {{source_queue}} {{destination_queue}}
```

- Move all messages from one queue to another:

```
sqsc mv {{source_queue}} {{destination_queue}}
```

- Describe a queue:

```
sqsc describe {{queue_name}}
```

- Query a queue with SQL syntax:

```
sqsc query "SELECT body FROM {{queue_name}} WHERE body
'%user%'"
```

- Pull all messages from a queue into a local sqlite database in your present working directory:

```
sqsc pull {{queue_name}}
```

srm

Securely remove files or directories.

Overwrites the existing data one or multiple times. Drop in replacement for rm.

More information: http://srm.sourceforge.net/srm.html.

- Remove a file after a single-pass overwriting with random data:

```
srm -s {{path/to/file}}
```

- Remove a file after seven passes of overwriting with random data:

```
srm -m {{path/to/file}}
```

- Recursively remove a directory and its contents overwriting each file with a single-pass of random data:

```
srm -r -s {{path/to/directory}}
```

- Prompt before every removal:

```
srm -i {{\*}}
```

ss-local

Run a Shadowsocks client as a SOCKS5 proxy.

More information: https://github.com/shadowsocks/shadowsocks-libev/blob/master/doc/ss-local.asciidoc.

- Run a Shadowsocks proxy by specifying the host, server port, local port, password, and encryption method:

  ```
  ss-local -s {{host}} -p {{server_port}} -l {{local port}} -k {{password}} -m {{encrypt_method}}
  ```

- Run a Shadowsocks proxy by specifying the config file:

  ```
  ss-local -c {{path/to/config/file.json}}
  ```

- Use a plugin to run the proxy client:

  ```
  ss-local --plugin {{plugin_name}} --plugin-opts {{plugin_options}}
  ```

- Enable TCP fast open:

  ```
  ss-local --fast-open
  ```

ssh-agent

Spawn an SSH Agent process.

An SSH Agent holds SSH keys decrypted in memory until removed or the process is killed.

See also `ssh-add`, which can add and manage keys held by an SSH Agent.

- Start an SSH Agent for the current shell:

  ```
  eval $(ssh-agent)
  ```

- Kill the currently running agent:

  ```
  ssh-agent -k
  ```

ssh-copy-id

Install your public key in a remote machine' s authoriz ed_ keys.

- Copy your keys to the remote machine:

  ```
  ssh- copy- id {{u sernam e@ rem ote_host}}
  ```

- Copy the given public key to the remote:

  ```
  ssh- copy- id - i {{path/to/certificate}} {{u sernam e}}
  @ {{rem ote_host}}
  ```

- Copy the given public key to the remote with specific port:

  ```
  ssh- copy- id - i {{path/to/certificate}} - p {{port}}
  {{u sernam e}}@ {{rem ote_host}}
  ```

ssh-keygen

Generate ssh keys used for authentication, password-less logins, and other things.

- Generate a key interactively:

```
ssh-keygen
```

- Specify file in which to save the key:

```
ssh-keygen -f ~/.ssh/{{filename}}
```

- Generate an ed25519 key with 100 key derivation function rounds:

```
ssh-keygen -t ed25519 -a 100
```

- Generate an RSA 4096 bit key with email as a comment:

```
ssh-keygen -t rsa -b 4096 -C "{{email}}"
```

- Retrieve the key fingerprint from a host (useful for confirming the authenticity of the host when first connecting to it via SSH):

```
ssh-keygen -l -F {{remote_host}}
```

- Remove the keys of a host from the known_hosts file (useful when a known host has a new key):

```
ssh-keygen -R {{remote_host}}
```

- Retrieve the fingerprint of a key in MD5 Hex:

```
ssh-keygen -l -E md5 -f ~/.ssh/{{filename}}
```

- Change the password of a key:

```
ssh-keygen -p -f ~/.ssh/{{filename}}
```

ssh-keyscan

G et the public ssh keys of remote hosts.

- R etrieve all public ssh keys of a remote host:

  ```
  ssh- k eyscan {{host}}
  ```

- R etrieve all public ssh keys of a remote host listening on a specific port:

  ```
  ssh- k eyscan - p {{port}} {{host}}
  ```

- R etrieve certain types of public ssh keys of a remote host:

  ```
  ssh- k eyscan - t {{rsa, dsa, ecdsa, ed2 5 5 1 9 }} {{host}}
  ```

- Manually update the ssh known_ hosts file with the fingerprint of a given host:

  ```
  ssh- k eyscan - H  {{host}} > >  ~ /. ssh/k now n_hosts
  ```

ssh

Secure Shell is a protocol used to securely log onto remote systems.

It can be used for logging or executing commands on a remote server.

- Connect to a remote server:

  ```
  ssh {{user name}}@{{remote_host}}
  ```

- Connect to a remote server with a specific identity (private key):

  ```
  ssh -i {{path/to/key_file}} {{user name}}@{{remote_host}}
  ```

- Connect to a remote server using a specific port:

  ```
  ssh {{user name}}@{{remote_host}} -p {{2222}}
  ```

- Run a command on a remote server:

  ```
  ssh {{remote_host}} {{command -with -flags}}
  ```

- SSH tunneling: Dynamic port forwarding (SOCKS proxy on localhost:1080):

  ```
  ssh -D {{1080}} {{user name}}@{{remote_host}}
  ```

- SSH tunneling: Forward a specific port (localhost:9999 to example.org:80) along with disabling pseudo-[t]ty allocation and executio[n] of remote commands:

  ```
  ssh -L {{9999}}:{{example.org}}:{{80}} -N -T {{user name}}
  @{{remote_host}}
  ```

- SSH jumping: Connect through a jumphost to a remote server (Multiple jump hops may be specified separated by comma characters):

  ```
  ssh -J {{user name}}@{{jump_host}} {{user name}}
  @{{remote_host}}
  ```

- Agent forwarding: Forward the authentication information to the remote machine (see `man ssh_config` for available options):

  ```
  ssh -A {{user name}}@{{remote_host}}
  ```

sshfs

Filesystem client based on ssh.

More information: https://github.com/libfuse/sshfs.

- Mount remote directory:

```
sshfs {{username}}@{{remote_host}}:{{remote_directory}} {{mountpoint}}
```

- Unmount remote directory:

```
umount {{mountpoint}}
```

- Mount remote directory from server with specific port:

```
sshfs {{username}}@{{remote_host}}:{{remote_directory}} -p {{2222}}
```

- Use compression:

```
sshfs {{username}}@{{remote_host}}:{{remote_directory}} -C
```

- Follow symbolic links:

```
sshfs -o follow_symlinks {{username}}@{{remote_host}}:{{remote_directory}} {{mountpoint}}
```

sshpass

An ssh password provider.

It works by creating a `TTY`, feeding the password into it, and then redirecting stdin to the ssh session.

- Connect to a remote server using a password supplied on a file descriptor (in this case, stdin):

  ```
  sshpass -d {{0}} ssh {{user}}@{{hostname}}
  ```

- Connect to a remote server with the password supplied as an option, and automatically accept unknown ssh keys:

  ```
  sshpass -p {{password}} ssh -o StrictHostKeyCheking=no {{user}}@{{hostname}}
  ```

- Connect to a remote server using the first line of a file as the password, automatically accept unknown ssh keys, and launch a command:

  ```
  sshpass -f {{file}} ssh -o StrictHostKeyCheking=no {{user}}@{{hostname}} "{{command}}"
  ```

st-flash

Flash binary files to STM32 ARM Cortex microcontrollers.

More information: https://github.com/texane/stlink.

- Read 4096 bytes from the device starting from 0x800000:

```
st-flash read {{firmware}}.bin {{0x800000}} {{4096}}
```

- Write firmware to device starting from 0x800000:

```
st-flash write {{firmware}}.bin {{0x800000}}
```

- Erase firmware from device:

```
st-flash erase
```

st-info

Provides information about connected STLink and STM32 devices.

More information: https://github.com/texane/stlink.

- Display amount of program memory available:

  ```
  st-info --flash
  ```

- Display amount of sram memory available:

  ```
  st-info --sram
  ```

- Display summarized information of the device:

  ```
  st-info --probe
  ```

st-util

Run GDB (GNU Debugger) server to interact with STM32 ARM Cortex microcontoller.

More information: https://github.com/texane/stlink.

- Run GDB server on port 4500:

```
st-util -p {{4500}}
```

- Connect to GDB server:

```
(gdb) target extended-remote {{localhost}}:{{4500}}
```

- Write firmware to device:

```
(gdb) load {{firmware.elf}}
```

stack

Tool for managing Haskell projects.

More information: https://github.com/commercialhaskell/stack.

- Create a new package:

  `stack new {{package_name}} {{template_name}}`

- Compile a package:

  `stack build`

- Run tests inside a package:

  `stack test`

- Compile a project and re-compile every time a file changes:

  `stack build --file-watch`

- Compile a project and execute a command after compilation:

  `stack build --exec "{{command}}"`

- Run a program and pass an argument to it:

  `stack exec {{program_name}} -- {{argument}}`

standard

The JavaScript Standard Style tool for linting and fixing JavaScript code.

More information: https://standardjs.com

- Lint all JavaScript source files in the current directory:

  ```
  standard
  ```

- Lint specific JavaScript file(s):

  ```
  standard {{path/to/file(s)}}
  ```

- Apply automatic fixes during linting:

  ```
  standard --fix
  ```

- Declare any available global variables:

  ```
  standard --global {{variable}}
  ```

- Use a custom ESLint plugin when linting:

  ```
  standard --plugin {{plugin}}
  ```

- Use a custom JS parser when linting:

  ```
  standard --parser {{parser}}
  ```

- Use a custom ESLint environment when linting:

  ```
  standard --env {{environment}}
  ```

starship init

Print shell integration code for starship.

More information: https://starship.rs.

- Display the subcommand help:

  `starship init --help`

- Print the starship integration code for the specified shell:

  `starship init {{bash|elvish|fish|ion|powershell|tcsh|zsh}}`

- Append the `starship` integration code to `~/.bashrc`:

  `starship init {{bash}} >> {{~/.bashrc}}`

- Append the `starship` integration code to `~/.zshrc`:

  `starship init {{zsh}} >> {{~/.zshrc}}`

starship

The minimal, blazing-fast, and infinitely customizable prompt for any shell.

More information: https://starship.rs.

- Print the starship integration code for the specified shell:

  `starship init {{bash|elvish|fish|ion|powershell|tcsh|zsh}}`

- Explain each part of the current prompt and show the time taken to render them:

  `starship explain`

- Print the computed starship configuration (use `--default` to print default configuration instead):

  `starship print-config`

- List supported modules:

  `starship module --list`

- Edit the starship configuration in the default editor:

  `starship configure`

- Create a bug report GitHub issue pre-populated with information about the system and starship configuration:

  `starship bug-report`

- Print the completion script for the specified shell:

  `starship completions {{bash|elvish|fish|powershell|zsh}}`

- Display help for a subcommand:

  `starship {{subcommand}} --help`

stat

Display file and filesystem information.

- Show file properties such as size, permissions, creation and access dates among others:

  ```
  stat {{file}}
  ```

- Same as above but in a more concise way:

  ```
  stat -t {{file}}
  ```

- Show filesystem information:

  ```
  stat -f {{file}}
  ```

- Show only octal file permissions:

  ```
  stat -c "%a %n" {{file}}
  ```

- Show owner and group of the file:

  ```
  stat -c "%U %G" {{file}}
  ```

- Show the size of the file in bytes:

  ```
  stat -c "%s %n" {{file}}
  ```

stdbuf

Run a command with modified buffering operations for its standard streams.

More information: https://www.gnu.org/software/coreutils/stdbuf.

- Change the standard input buffer size to 512 KiB:

```
stdbuf --input={{512K}} {{command}}
```

- Change the standard output buffer to line-buffered:

```
stdbuf --output={{L}} {{command}}
```

- Change the standard error buffer to unbuffered:

```
stdbuf --error={{0}} {{command}}
```

steam

Video game platform by Valve.

More information: https://developer.valvesoftware.com/wiki/Command_Line_Options#Steam_.28Windows.29

- Launch Steam, printing debug messages to stdout:

```
steam
```

- Launch Steam and enable its in-app debug console tab:

```
steam -console
```

- Log into Steam with the specified credentials:

```
steam -login {{username}} {{password}}
```

- Launch Steam in Big Picture Mode:

```
steam -tenfoot
```

- Exit Steam:

```
steam -shutdown
```

steamcmd

A command-line version of the S team client.

More information: https://manned.org/steamcmd.

- Install or update an application anonymously:

```
steam cm d + log in {{anonym ou s}} + app_u pdate {{appid}} + q u it
```

- Install or update an application using the specified credentials:

```
steam cm d + log in {{u sernam e}} + app_u pdate {{appid}} + q u it
```

- Install an application for a specific platform:

```
steam cm d + @ sS team C m dF orceP latform T ype {{w indow s}} + log in
{{anonym ou s}} + app_u pdate {{appid}} v alidate + q u it
```

stern

Tail multiple pods and containers from Kubernetes.

More information: https://github.com/wercker/stern/.

- Tail all pods within a current namespace:

  ```
  stern .
  ```

- Tail all pods with a specific status:

  ```
  stern . --container-state {{running|waiting|terminated}}
  ```

- Tail all pods that matches a given regular expression:

  ```
  stern {{pod_query}}
  ```

- Tail matched pods from all namespaces:

  ```
  stern {{pod_query}} --all-namespaces
  ```

- Tail matched pods from 15 minutes ago:

  ```
  stern {{pod_query}} --since {{15m}}
  ```

- Tail matched pods with a specific label:

  ```
  stern {{pod_query}} --selector {{release=canary}}
  ```

stl2gts

Convert STL files into the GTS (GNU triangulated surface library) file format.

More information: https://manned.org/stl2gts

- Convert an STL file to a GTS file:

```
stl2gts < {{path/to/file.stl}} > {{path/to/file.gts}}
```

- Convert an STL file to a GTS file and revert face normals:

```
stl2gts --revert < {{path/to/file.stl}} > {{path/to/file.gts}}
```

- Convert an STL file to a GTS file and do not merge vertices:

```
stl2gts --nomerge < {{path/to/file.stl}} > {{path/to/file.gts}}
```

- Convert an STL file to a GTS file and display surface statistics:

```
stl2gts --verbose < {{path/to/file.stl}} > {{path/to/file.gts}}
```

- Print help for `stl2gts`:

```
stl2gts --help
```

stolonctl

CLI for Stolon, a cloud native PostgreSQL manager for PostgreSQL high availability.

More information: https://github.com/sorintlab/stolon.

- Get cluster status:

  ```
  stolonctl --cluster-name {{cluster_name}} --store-backend {{store_backend}} --store-endpoints {{store_endpoints}} status
  ```

- Get cluster data:

  ```
  stolonctl --cluster-name {{cluster_name}} --store-backend {{store_backend}} --store-endpoints {{store_endpoints}} clusterdata
  ```

- Get cluster specification:

  ```
  stolonctl --cluster-name {{cluster_name}} --store-backend {{store_backend}} --store-endpoints {{store_endpoints}} spec
  ```

- Update cluster specification with a patch in json format:

  ```
  stolonctl --cluster-name {{cluster_name}} --store-backend {{store_backend}} --store-endpoints {{store_endpoints}} update --patch '{{cluster_spec}}'
  ```

Stormlock

Centralized locking system.

More information: https://github.com/tmccombs/stormlock.

- Acquire a lease for resource:

  ```
  storm lock acquire {{resource}}
  ```

- Release the given lease for the given resource:

  ```
  storm lock release {{resource}} {{lease_id}}
  ```

- Show information on the current lease for a resource, if any:

  ```
  storm lock current {{resource}}
  ```

- Test if a lease for given resource is currently active:

  ```
  storm lock is-held {{resource}} {{lease_id}}
  ```

stow

Symlink manager.

Often used to manage dotfiles.

More information: https://www.gnu.org/software/stow.

- Symlink all files recursively to a given directory:

```
stow --target={{path/to/target_directory}} {{file1 directory1 file2 directory2 }}
```

- Delete symlinks recursively from a given directory:

```
stow --delete --target={{path/to/target_directory}} {{file1 directory1 file2 directory2 }}
```

- Simulate to see what the result would be like:

```
stow --simulate --target={{path/to/target_directory}} {{file1 directory1 file2 directory2 }}
```

- Delete and resymlink:

```
stow --restow --target={{path/to/target_directory}} {{file1 directory1 file2 directory2 }}
```

- Exclude files matching a regular expression:

```
stow --ignore={{regular_expression}} --target={{path/to/target_directory}} {{file1 directory1 file2 directory2 }}
```

streamlink

Extracts streams from various services and pipes them into a video player of choice.

More information: https://streamlink.github.io.

- Attempt to extract streams from the URL specified, and if it's successful, print out a list of available streams to choose from:

  ```
  streamlink {{example.com/stream}}
  ```

- Open a stream with the specified quality:

  ```
  streamlink {{example.com/stream}} {{720p60}}
  ```

- Select the highest or lowest available quality:

  ```
  streamlink {{example.com/stream}} {{best|worst}}
  ```

- Specify which player to use to feed stream data to (VLC is used by default if found):

  ```
  streamlink --player={{mpv}} {{example.com/stream}} {{best}}
  ```

- Specify the amount of time to skip from the beginning of the stream. For live streams, this is a negative offset from the end of the stream (rewind):

  ```
  streamlink --hls-start-offset {{[HH:]MM:SS}} {{example.com
  stream}} {{best}}
  ```

- Skip to the beginning of a live stream, or as far back as possible:

  ```
  streamlink --hls-live-restart {{example.com/stream}} {{best}}
  ```

- Write stream data to a file instead of playing it:

  ```
  streamlink --output {{path/to/file.ts}} {{example.com/
  stream}} {{best}}
  ```

- Open the stream in the player, while at the same time writing it to a file:

  ```
  streamlink --record {{path/to/file.ts}} {{example.com/
  stream}} {{best}}
  ```

strings

Find printable strings in an object file or binary.

- Print all strings in a binary:

  ```
  strings {{file}}
  ```

- Limit results to strings at least length characters long:

  ```
  strings -n {{length}} {{file}}
  ```

- Prefix each result with its offset within the file:

  ```
  strings -td {{file}}
  ```

- Prefix each result with its offset within the file in hexadecimal:

  ```
  strings -tx {{file}}
  ```

strip-nondeterminism

A tool to remove non-deterministic information (e.g. timestamps) from files.

More information: https://salsa.debian.org/reproducible-builds/strip-nondeterminism.

- Strip nondeterministic information from a file:

  ```
  strip-nondeterminism {{path/to/file}}
  ```

- Strip nondeterministic information from a file manually specifying the filetype:

  ```
  strip-nondeterminism --type {{filetype}} {{path/to/file}}
  ```

- Strip nondeterministic information from a file; instead of removing timestamps set them to the specified UNIX timestamp:

  ```
  strip-nondeterminism --timestamp {{unix_timestamp}} {{path/to/file}}
  ```

stripe

Interact with a Stripe account.

More information: https://github.com/stripe/stripe-cli.

- Follow the logs of activity on the account:

  ```
  stripe logs tail
  ```

- Listen for events, filtering on events with the name `charge.succeeded` and forwarding them to localhost:3000/events:

  ```
  stripe listen --events="{{charge.succeeded}}" --forward-to="{{localhost:3000/events}}"
  ```

- Send a test webhook event:

  ```
  stripe trigger {{charge.succeeded}}
  ```

- Create a customer:

  ```
  stripe customers create --email="{{test@example.com}}" --name="{{Jenny Rosen}}"
  ```

- Print to JSON:

  ```
  stripe listen --print-json
  ```

stty

Set options for a terminal device interface.

More information: https://www.gnu.org/software/coreutils/stty.

- Display all settings for the current terminal:

```
stty -a
```

- Set the number of rows:

```
stty rows {{rows}}
```

- Set the number of columns:

```
stty cols {{cols}}
```

- Get the actual transfer speed of a device:

```
stty -F {{path/to/device_file}} speed
```

- Reset all modes to reasonable values for the current terminal:

```
stty sane
```

su

Switch shell to another user.

- Switch to superuser (requires the root password):

  ```
  su
  ```

- Switch to a given user (requires the user's password):

  ```
  su {{username}}
  ```

- Switch to a given user and simulate a full login shell:

  ```
  su - {{username}}
  ```

- Execute a command as another user:

  ```
  su - {{username}} -c "{{command}}"
  ```

subfinder

A subdomain discovery tool that discovers valid subdomains for websites.

Designed as a passive framework to be useful for bug bounties and safe for penetration testing.

More information: https://github.com/subfinder/subfinder.

- Find subdomains for a specific domain:

```
subfinder -d {{example.com}}
```

- Show only the subdomains found:

```
subfinder --silent -d {{example.com}}
```

- Use a brute-force attack to find subdomains:

```
subfinder -d {{example.com}} -b
```

- Remove wildcard subdomains:

```
subfinder -nW -d {{example.com}}
```

- Use a given comma-separated list of resolvers:

```
subfinder -r {{8.8.8.8}},{{1.1.1.1}} -d {{example.com}}
```

subl

Sublime Text editor.

More information: https://www.sublimetext.com.

- Open the current directory in Sublime Text:

  ```
  subl {{.}}
  ```

- Open a file or directory in Sublime Text:

  ```
  subl {{path/to/file_or_directory}}
  ```

- Open a file and jump to a specific line number:

  ```
  subl {{path/to/file}}:{{line_number}}
  ```

- Open a file or directory in the currently open window:

  ```
  subl -a {{path/to/file}}
  ```

- Open a file or directory in a new window:

  ```
  subl -n {{path/to/file}}
  ```

subliminal

P ython-based subtitle downloader.

More information: https://github.com/Diaoul/subliminal.

- Download E nglish subtitles for a video:

```
su b lim inal dow nload - l {{en}} {{v ideo. ext}}
```

sublist3 r

F ast subdomains enumeration tool for penetration testers.

More information: https://github.com/aboul3 la/S ublist3 r

- F ind subdomains for a domain:

```
su b list3 r - - dom ain {{dom ain_nam e}}
```

- F ind subdomains for a domain, also enabling brute force search:

```
su b list3 r - - dom ain {{dom ain_nam e}} - - b ru teforce
```

- S ave the found subdomains to a text file:

```
su b list3 r - - dom ain {{dom ain_nam e}} - - ou tpu t {{path/to/
ou tpu t_file}}
```

- O utput all available options:

```
su b list3 r - - help
```

sudo

Executes a single command as the superuser or another user.

More information: https://www.sudo.ws/sudo.html.

- Run a command as the superuser:

  ```
  sudo {{less /var/log/syslog}}
  ```

- Edit a file as the superuser with your default editor:

  ```
  sudo --edit {{/etc/fstab}}
  ```

- Run a command as another user and/or group:

  ```
  sudo --user={{user}} --group={{group}} {{id -a}}
  ```

- Repeat the last command prefixed with `sudo` (only in `bash`, `zsh`, etc.):

  ```
  sudo !!
  ```

- Launch the default shell with superuser privileges and run login-specific files (`.profile`, `.bash_profile`, etc.):

  ```
  sudo --login
  ```

- Launch the default shell with superuser privileges without changing the environment:

  ```
  sudo --shell
  ```

- Launch the default shell as the specified user, loading the user's environment and reading login-specific files (`.profile`, `.bash_profile`, etc.):

  ```
  sudo --login --user={{user}}
  ```

- List the allowed (and forbidden) commands for the invoking user:

  ```
  sudo --list
  ```

sum

Compute checksums and the number of blocks for a file.

A predecessor to the more modern `cksum`

More information: https://www.gnu.org/software/coreutils/sum.

- Compute a checksum with BSD-compatible algorithm and 1024-byte blocks:

  ```
  sum {{file}}
  ```

- Compute a checksum with System V-compatible algorithm and 512-byte blocks:

  ```
  sum --sysv {{file}}
  ```

supervisorctl

Supervisor is a client/server system that allows its users to control a number of processes on UNIX-like operating systems.

Supervisorctl is the command-line client piece of the supervisor which provides a shell-like interface.

More information: http://supervisord.org.

- Start/stop/restart a process:

```
supervisorctl {{start|stop|restart}} {{process_name}}
```

- Start/stop/restart all processes in a group:

```
supervisorctl {{start|stop|restart}} {{group_name}}:*
```

- Show last 100 bytes of process stderr:

```
supervisorctl tail -100 {{process_name}} stderr
```

- Keep displaying stdout of a process:

```
supervisorctl tail -f {{process_name}} stdout
```

- Reload process config file to add/remove processes as necessary:

```
supervisorctl update
```

supervisord

Supervisor is a client/server system for controlling some processes on UNIX-like operating systems.

Supervisord is the server part of supervisor; it is primarily managed via a configuration file.

More information: http://supervisord.org.

- Start supervisord with specified configuration file:

  ```
  supervisord -c {{path/to/file}}
  ```

- Run supervisord in the foreground:

  ```
  supervisord -n
  ```

surfraw

CLI to query a variety of web search engines.

Consists of a collection of elvi, each of which knows how to search a specific website.

More information: http://surfraw.org.

- Display the list of supported website search scripts (elvi):

  ```
  surfraw -elvi
  ```

- Open the elvi's results page for a specific search in the browser:

  ```
  surfraw {{elvi}} "{{search_terms}}"
  ```

- Display an elvi description and its specific options:

  ```
  surfraw {{elvi}} -local-help
  ```

- Search using an elvi with specific options and open the results page in the browser:

  ```
  surfraw {{elvi}} {{elvi_options}} "{{search_terms}}"
  ```

- Display the URL to the elvi's results page for a specific search:

  ```
  surfraw -print {{elvi}} "{{search_terms}}"
  ```

- Search using the alias:

  ```
  sr {{elvi}} "{{search_terms}}"
  ```

surge

Simple command-line web publishing.

More information: https://surge.sh.

- Upload a new site to surge.sh:

```
surge {{path/to/my_project}}
```

- Deploy site to custom domain (note that the DNS records must point to the surge.sh subdomain):

```
surge {{path/to/my_project}} {{my_custom_domain.com}}
```

- List your surge projects:

```
surge list
```

- Remove a project:

```
surge teardown {{my_custom_domain.com}}
```

sv

Control a running runsv service.

More information: https://manpages.ubuntu.com/manpages/latest/man8/sv.8.html

- Start a service:

```
sudo sv up {{path/to/service}}
```

- Stop a service:

```
sudo sv down {{path/to/service}}
```

- Get service status:

```
sudo sv status {{path/to/service}}
```

- Reload a service:

```
sudo sv reload {{path/to/service}}
```

- Start a service, but only if it's not running and don't restart it if it stops:

```
sudo sv once {{path/to/service}}
```

svgcleaner

S V G image optimizing utility.

More information: https://github.com/RazrFalcon/svgcleaner

- Optimize an SVG image:

  ```
  svgcleaner {{input.svg}} {{output.svg}}
  ```

- Optimize an SVG image multiple times:

  ```
  svgcleaner --multipass {{input.svg}} {{output.svg}}
  ```

svgo

SVGO ptimizer: a Node.js-based tool for optimizing Scalable Vector Graphics files.

It applies a series of transformation rules (plugins), which can be toggled individually.

More information: https://github.com/svg/svgo.

- Optimize a file using the default plugins (overwrites the original file):

  ```
  svgo {{test.svg}}
  ```

- Optimize a file and save the result to another file:

  ```
  svgo {{test.svg}} -o {{test.min.svg}}
  ```

- Optimize all SVG files within a directory (overwrites the original files):

  ```
  svgo -f {{path/to/directory/with/svg/files}}
  ```

- Optimize all SVG files within a directory and save the resulting files to another directory:

  ```
  svgo -f {{path/to/input/directory}} -o {{path/to/output/directory}}
  ```

- Optimize SVG content passed from another command, and save the result to a file:

  ```
  {{cat test.svg}} | svgo -i - -o {{test.min.svg}}
  ```

- Optimize a file and print out the result:

  ```
  svgo {{test.svg}} -o -
  ```

- Optimize a file making sure a given plugin is enabled:

  ```
  svgo --enable={{plugin_name}}
  ```

- Show available plugins:

  ```
  svgo --show-plugins
  ```

svn changelist

Associate a changelist with a set of files.

More information: http://svnbook.red-bean.com/en/1.7/svn.advanced.changelists.html

- Add files to a changelist, creating the changelist if it does not exist:

```
svn changelist {{changelist_name}} {{path/to/file1}} {{path/to/file2}}
```

- Remove files from a changelist:

```
svn changelist --remove {{path/to/file1}} {{path/to/file2}}
```

- Remove the whole changelist at once:

```
svn changelist --remove --recursive --changelist {{changelist_name}} .
```

- Add the contents of a space-separated list of directories to a changelist:

```
svn changelist --recursive {{changelist_name}} {{path/to/directory1}} {{path/to/directory2}}
```

- Commit a changelist:

```
svn commit --changelist {{changelist_name}}
```

svn

Subversion command-line client tool.

More information: https://subversion.apache.org.

- Check out a working copy from a repository:

  ```
  svn co {{url/to/repository}}
  ```

- Bring changes from the repository into the working copy:

  ```
  svn up
  ```

- Put files and directories under version control, scheduling them for addition to repository. They will be added in next commit:

  ```
  svn add {{PATH}}
  ```

- Send changes from your working copy to the repository:

  ```
  svn ci -m {{commit_log_message}} [{{PATH}}]
  ```

- Display changes from the last 10 revisions, showing modified files for each revision:

  ```
  svn log -vl {{10}}
  ```

- Show detailed help:

  ```
  svn help
  ```

swagger-codegen

Generate code and documentation for your REST api from a OpenAPI/swagger definition.

More information: https://github.com/swagger-api/swagger-codegen.

- Generate documentation and code from an OpenAPI/swagger file:

  ```
  swagger-codegen generate -i {{swagger_file}} -l {{language}}
  ```

- Generate Java code using the library retrofit2 and the option useRxJava2:

  ```
  swagger-codegen generate -i {{http://petstore.swagger.io/v2/
  swagger.json}} -l {{java}} --library {{retrofit2}} -
  D {{useRxJava2}}={{true}}
  ```

- List available languages:

  ```
  swagger-codegen langs
  ```

- Display help options for the generate command:

  ```
  swagger-codegen help {{generate}}
  ```

swift

Create, run and build Swift projects.

More information: https://swift.org.

- Invoke the interactive interpreter (REPL):

  ```
  swift
  ```

- Execute a program:

  ```
  swift {{file.swift}}
  ```

- Start a new project with the package manager:

  ```
  swift package init
  ```

- Generate an Xcode project file:

  ```
  swift package generate-xcodeproj
  ```

- Update dependencies:

  ```
  swift package update
  ```

- Compile project for release:

  ```
  swift build -c release
  ```

swig

Generate bindings between C / C++ code and various high level languages such as Java Script, Python, C#, and more.

It uses special .i or .swg files to generate the bindings (C/C++ with SWIG directives, then outputs a C/C++ file that contains all of the wrapper code needed to build an extension module.

- Generate a binding between C++ and Python:

  ```
  swig -c++ -python -o {{path/to/output_wrapper.cpp}} {{path/to/swig_file.i}}
  ```

- Generate a binding between C++ and Go:

  ```
  swig -go -cgo -intgo -size 64 -c++ {{path/to/swig_file.i}}
  ```

- Generate a binding between C and Java:

  ```
  swig -java {{path/to/swig_file.i}}
  ```

- Generate a binding between C and Ruby and prefix the Ruby module with {{foo::bar::}}:

  ```
  swig -ruby -prefix "{{foo::bar::}}" {{path/to/swig_file.i}}
  ```

symfony

The console component for the Symfony framework.

More information: https://symfony.com.

- Create a new Symfony project:

  ```
  symfony new {{name}}
  ```

- Run a local web server:

  ```
  symfony serve
  ```

- Stop the local web server:

  ```
  symfony server:stop
  ```

- Check for security issues in the project's dependencies:

  ```
  symfony security:check
  ```

sync

Flushes all pending write operations to the appropriate disks.

More information: https://www.gnu.org/software/coreutils/sync.

- Flush all pending write operations on all disks:

```
sync
```

- Flush all pending write operations on a single file to disk:

```
sync {{path/to/file}}
```

syncthing

Continuous bidirectional decentralised folder synchronisation tool.

More information: https://docs.syncthing.net/.

- Start syncthing:

  ```
  syncthing
  ```

- Start syncthing without opening a web browser:

  ```
  syncthing --no-browser
  ```

- Print the device ID:

  ```
  syncthing --device-id
  ```

- Change the home directory:

  ```
  syncthing --home={{path/to/directory}}
  ```

- Force a full index exchange:

  ```
  syncthing --reset-deltas
  ```

- Change the address upon which the web interface listens:

  ```
  syncthing --gui-address={{ip_address:port|path/to/socket.sock}}
  ```

- Show filepaths to the files used by syncthing:

  ```
  syncthing --paths
  ```

- Disable the syncthing monitor process:

  ```
  syncthing --no-restart
  ```

tabula

Extract tables from PDF files.

More information: https://tabula.technology.

- Extract all tables from a PDF to a CSV file:

```
tabula -o {{file.csv}} {{file.pdf}}
```

- Extract all tables from a PDF to a JSON file:

```
tabula --format JSON -o {{file.json}} {{file.pdf}}
```

- Extract tables from pages 1, 2, 3, and 6 of a PDF:

```
tabula --pages {{1-3,6}} {{file.pdf}}
```

- Extract tables from page 1 of a PDF, guessing which portion of the page to examine:

```
tabula --guess --pages {{1}} {{file.pdf}}
```

- Extract all tables from a PDF, using ruling lines to determine cell boundaries:

```
tabula --spreadsheet {{file.pdf}}
```

- Extract all tables from a PDF, using blank space to determine cell boundaries:

```
tabula --no-spreadsheet {{file.pdf}}
```

tac

Print and concatenate files in reverse (last line first).

More information: https://www.gnu.org/software/coreutils/tac.

- Print the contents of file1 reversed to the standard output:

  ```
  tac {{file1}}
  ```

- Print the contents of the standard input reversed to the standard output:

  ```
  {{command}} | tac
  ```

- Concatenate several files reversed into the target file:

  ```
  tac {{file1}} {{file2}} > {{target_file}}
  ```

tail

Display the last part of a file.

More information: https://www.gnu.org/software/coreutils/tail.

- Show last 'num' lines in file:

```
tail -n {{num}} {{file}}
```

- Show all file since line 'num':

```
tail -n +{{num}} {{file}}
```

- Show last 'num' bytes in file:

```
tail -c {{num}} {{file}}
```

- Keep reading file until `Ctrl + C`:

```
tail -f {{file}}
```

- Keep reading file until `Ctrl + C`, even if the file is rotated:

```
tail -F {{file}}
```

- Show last 'num' lines in 'file' and refresh every 'n' seconds:

```
tail -n {{num}} -s {{n}} -f {{file}}
```

takeout

A Docker-based development-only dependency manager.

More information: https://github.com/tighten/takeout.

- Display a list of available services:

  ```
  tak eou t enab le
  ```

- E nable a specific service:

  ```
  tak eou t enab le {{nam e}}
  ```

- E nable a specific service with the default parameters:

  ```
  tak eou t enab le - - defau lt {{nam e}}
  ```

- Display a list of enabled services:

  ```
  tak eou t disab le
  ```

- Disable a specific service:

  ```
  tak eou t disab le {{nam e}}
  ```

- Disable all services:

  ```
  tak eou t disab le - - all
  ```

- S tart a specific container:

  ```
  tak eou t start {{container_id}}
  ```

- S top a specific container:

  ```
  tak eou t stop {{container_id}}
  ```

tar

Archiving utility.

Often combined with a compression method, such as gzip or bzip2.

More information: https://www.gnu.org/software/tar.

- [c]reate an archive and write it to a [f]ile:

```
tar cf {{target.tar}} {{file1}} {{file2}} {{file3}}
```

- [c]reate a g[z]ipped archive and write it to a [f]ile:

```
tar czf {{target.tar.gz}} {{file1}} {{file2}} {{file3}}
```

- [c]reate a g[z]ipped archive from a directory using relative paths:

```
tar czf {{target.tar.gz}} --directory={{path/to/directory}} .
```

- E[x]tract a (compressed) archive [f]ile into the current directory [v]erbosely:

```
tar xvf {{source.tar[.gz|.bz2|.xz]}}
```

- E[x]tract a (compressed) archive [f]ile into the target directory:

```
tar xf {{source.tar[.gz|.bz2|.xz]}} --directory={{directory}}
```

- [c]reate a compressed archive and write it to a [f]ile, using [a]rchive suffix to determine the compression program:

```
tar caf {{target.tar.xz}} {{file1}} {{file2}} {{file3}}
```

- Lis[t] the contents of a tar [f]ile [v]erbosely:

```
tar tvf {{source.tar}}
```

- E[x]tract files matching a pattern from an archive [f]ile:

```
tar xf {{source.tar}} --wildcards "{{*.html}}"
```

task

TODO list manager.

- Add new task:

  ```
  task add {{thing_to_do}}
  ```

- List tasks:

  ```
  task list
  ```

- Mark task as completed:

  ```
  task {{task_id}} done
  ```

- Modify task:

  ```
  task {{task_id}} modify {{new_thing_to_do}}
  ```

- Delete task:

  ```
  task {{task_id}} delete
  ```

tb

CLI for managing tasks and notes across multiple boards.

More information: https://github.com/klaussinani/taskbook.

- Add a new task to a board:

  ```
  tb --task {{task_description}} @{{board_name}}
  ```

- Add a new note to a board:

  ```
  tb --note {{note_description}} @{{board_name}}
  ```

- Edit item's priority:

  ```
  tb --priority @{{item_id}} {{priority}}
  ```

- Check/uncheck item:

  ```
  tb --check {{item_id}}
  ```

- Archive all checked items:

  ```
  tb --clear
  ```

- Move item to a board:

  ```
  tb --move @{{item_id}} {{board_name}}
  ```

tcpdump

Dump traffic on a network.

More information: https://www.tcpdump.org.

- List available network interfaces:

```
tcpdump -D
```

- Capture the traffic of a specific interface:

```
tcpdump -i {{eth0}}
```

- Capture all TCP traffic showing contents (ASCII) in console:

```
tcpdump -A tcp
```

- Capture the traffic from or to a host:

```
tcpdump host {{www.example.com}}
```

- Capture the traffic from a specific interface, source, destination and destination port:

```
tcpdump -i {{eth0}} src {{192.168.1.1}} and dst {{192.168.1.2}} and dst port {{80}}
```

- Capture the traffic of a network:

```
tcpdump net {{192.168.1.0/24}}
```

- Capture all traffic except traffic over port 22 and save to a dump file:

```
tcpdump -w {{dumpfile.pcap}} port not {{22}}
```

- Read from a given dump file:

```
tcpdump -r {{dumpfile.pcap}}
```

tea

A command-line tool to interact with G itea servers.

More information: https://gitea.com/gitea/tea.

- Log into a G itea server:

```
tea log in add - - nam e " {{nam e}}"  - - u rl " {{u rl}}"  - - tok en " {{tok en}}"
```

- Display all repositories:

```
tea repos ls
```

- Display a list of issues:

```
tea issu es ls
```

- Display a list of issues for a specific repository:

```
tea issu es ls - - repo " {{repository}}"
```

- Create a new issue:

```
tea issu es create - - title " {{title}}"  - - b ody " {{b ody}}"
```

- Display a list of open pull req uests:

```
tea pu lls ls
```

- O pen the current repository in a browser:

```
tea open
```

tee

Read from standard input and write to standard output and files (or commands).

More information: https://www.gnu.org/software/coreutils/tee.

- Copy standard input to each FILE, and also to standard output:

```
echo "example" | tee {{FILE}}
```

- Append to the given FILEs, do not overwrite:

```
echo "example" | tee -a {{FILE}}
```

- Print standard input to the terminal, and also pipe it into another program for further processing:

```
echo "example" | tee {{/dev/tty}} | {{xargs printf "[%s]"}}
```

- Create a directory called "example", count the number of characters in "example" and write "example" to the terminal:

```
echo "example" | tee >(xargs mkdir) >(wc -c)
```

telnet

Connect to a specified port of a host using the telnet protocol.

- Telnet to the default port of a host:

  ```
  telnet {{host}}
  ```

- Telnet to a specific port of a host:

  ```
  telnet {{ip_address}} {{port}}
  ```

- Exit a telnet session:

  ```
  quit
  ```

- Emit the default escape character combination for terminating the session:

  ```
  Ctrl+]
  ```

- Start telnet with "x" as the session termination character:

  ```
  telnet -e {{x}} {{ip_address}} {{port}}
  ```

- Telnet to Star Wars animation:

  ```
  telnet {{towel.blinkenlights.nl}}
  ```

terminaliz er

U tility program which records the terminal and generate animated gifs or share a video.

More information: https://terminaliz er.com

- Create the global config directory:

  ```
  term inalizer init
  ```

- R ecord the terminal and create a recording file:

  ```
  term inalizer record {{filenam e}}
  ```

- P lay a recorded file on the terminal:

  ```
  term inalizer play {{filenam e}}
  ```

- R ender a recording file as an animated gif image:

  ```
  term inalizer render {{filenam e}}
  ```

- U pload a video to terminaliz er.com:

  ```
  term inalizer share {{filenam e}}
  ```

terraform fmt

Format configuration according to Terraform language style conventions.

More information: https://www.terraform.io/docs/commands/fmt.html.

- Format the configuration in the current directory:

  ```
  terraform fmt
  ```

- Format the configuration in the current directory and subdirectories:

  ```
  terraform fmt -recursive
  ```

- Display diffs of formatting changes:

  ```
  terraform fmt -diff
  ```

- Do not list files that were formatted to stdout:

  ```
  terraform fmt -list=false
  ```

terraform

Create and deploy infrastructure as code to cloud providers.

More information: https://www.terraform.io/.

- Initialize a new or existing Terraform configuration:

  ```
  terraform init
  ```

- Verify that the configuration files are syntactically valid:

  ```
  terraform validate
  ```

- Format configuration according to Terraform language style conventions:

  ```
  terraform fmt
  ```

- Generate and show an execution plan:

  ```
  terraform plan
  ```

- Build or change infrastructure:

  ```
  terraform apply
  ```

- Destroy Terraform-managed infrastructure:

  ```
  terraform destroy
  ```

tesseract

OCR (Optical Character Recognition) engine.

More information: https://github.com/tesseract-ocr/tesseract.

- Recognize text in an image and save it to `output.txt` (the `.txt` extension is added automatically):

  ```
  tesseract {{image}} {{output}}
  ```

- Specify a custom language (default is English) with an ISO 639-2 code (e.g. deu = Deutsch = German):

  ```
  tesseract -l deu {{image}} {{output}}
  ```

- List the ISO 639-2 codes of available languages:

  ```
  tesseract --list-langs
  ```

- Specify a custom page segmentation mode (default is 3):

  ```
  tesseract -psm {{0_to_10}} {{image}} {{output}}
  ```

- List page segmentation modes and their descriptions:

  ```
  tesseract --help-psm
  ```

test

Evaluate condition.

Returns 0 if the condition evaluates to true, 1 if it evaluates to false.

More information: https://www.gnu.org/software/coreutils/test.

- Test if a given variable is equal to a given string:

```
test "{{$MY_VAR}}" == "{{/bin/zsh}}"
```

- Test if a given variable is empty:

```
test -z "{{$GIT_BRANCH}}"
```

- Test if a file exists:

```
test -f "{{path/to/file_or_directory}}"
```

- Test if a directory does not exist:

```
test ! -d "{{path/to/directory}}"
```

- If-else statement:

```
test {{condition}} && {{echo "true"}} || {{echo "false"}}
```

testssl

Check SSL/TLS protocols and ciphers supported by a server.

More information: https://testssl.sh/.

- Test a server (run every check) on port 443:

  ```
  testssl {{example.com}}
  ```

- Test a different port:

  ```
  testssl {{example.com:465}}
  ```

- Only check available protocols:

  ```
  testssl --protocols {{example.com}}
  ```

- Only check vulnerabilities:

  ```
  testssl --vulnerable {{example.com}}
  ```

- Only check HTTP security headers:

  ```
  testssl --headers {{example.com}}
  ```

tex

Compile a DVI document from TEX source files.

More information: https://www.tug.org/begin.html.

- Compile a DVI document:

```
tex {{source.tex}}
```

- Compile a DVI document, specifying an output directory:

```
tex -output-directory={{path/to/directory}} {{source.tex}}
```

- Compile a DVI document, halting on each error:

```
tex -halt-on-error {{source.tex}}
```

texdoc

Search for appropriate documentation for (La)TeX commands or packages.

More information: https://texdoc.org/index.html.

- Open the first search result in the default PDF viewer:

  ```
  texdoc {{search}}
  ```

- List the best search results:

  ```
  texdoc --list {{search}}
  ```

- Open full documentation of texdoc:

  ```
  texdoc {{texdoc}}
  ```

texliveonfly

Downloads missing TeX Live packages while compiling `.tex` files.

More information: https://ctan.org/pkg/texliveonfly

- Download missing packages while compiling:

  ```
  texliveonfly {{source.tex}}
  ```

- Use a specific compiler (defaults to `pdflatex`):

  ```
  texliveonfly --compiler={{compiler}} {{source.tex}}
  ```

- Use a custom TeX Live `bin` folder:

  ```
  texliveonfly --texlive_bin={{path/to/texlive_bin}}
  {{source.tex}}
  ```

theH arvester

A tool designed to be used in the early stages of a penetration test.

More information: https://github.com/laramies/theH arvester

- G ather information on a domain using G oogle:

  ```
  theH arv ester - - dom ain {{dom ain_nam e}} - - sou rce g oog le
  ```

- G ather information on a domain using multiple sources:

  ```
  theH arv ester - - dom ain {{dom ain_nam e}} - - sou rce
  {{g oog le, b ing , crtsh}}
  ```

- Change the limit of results to work with:

  ```
  theH arv ester - - dom ain {{dom ain_nam e}} - - sou rce {{g oog le}} - -
  lim it {{2 0 0 }}
  ```

- S ave the output to two files in xml and html format:

  ```
  theH arv ester - - dom ain {{dom ain_nam e}} - - sou rce {{g oog le}} - -
  file {{ou tpu t_file_nam e}}
  ```

- O utput all available options:

  ```
  theH arv ester - - help
  ```

thunderbird

Email client and RSS reader.

More information: https://thunderbird.net.

- Open thunderbird:

  ```
  thunderbird
  ```

- Use a specific user profile:

  ```
  thunderbird -P {{profile_name}}
  ```

- Use a specific user profile directory:

  ```
  thunderbird --profile {{path/to/profile/directory}}
  ```

tig

A text-mode interface for G it.

More information: https://github.com/j onas/tig

- S how the seq uence of commits starting from the current one in reverse chronological order:

  ```
  tig
  ```

- S how the history of a specific branch:

  ```
  tig {{b ranch}}
  ```

- S how the history of specific files or directories:

  ```
  tig {{path1 path2 … }}
  ```

- S how the difference between two references (such as branches or tags):

  ```
  tig {{b ase_ref}}. . {{com pared_ref}}
  ```

- Display commits from all branches and stashes:

  ```
  tig - - all
  ```

- S tart in stash view, displaying all saved stashes:

  ```
  tig stash
  ```

time

See how long a command takes.

- Time "ls":

```
time ls
```

timeout

Run a command with a time limit.

More information: https://www.gnu.org/software/coreutils/timeout.

- Run `sleep 10` and terminate it, if it runs for more than 3 seconds:

  ```
  timeout {{3s}} {{sleep 10}}
  ```

- Specify the signal to be sent to the command after the time limit expires. (By default, TERM is sent):

  ```
  timeout --signal {{INT}} {{5s}} {{sleep 10}}
  ```

timew

A time tracking tool used to measure the duration of activities.

More information: https://timewarrior.net/docs.

- Start a new stopwatch, giving a tag name to the activity being tracked:

  ```
  timew start {{activity_tag}}
  ```

- View running stopwatches:

  ```
  timew
  ```

- Stop the stopwatch with a given tag name:

  ```
  timew stop {{activity_tag}}
  ```

- Stop all running stopwatches:

  ```
  timew stop
  ```

- View tracked items:

  ```
  timew summary
  ```

tldr-lint

Lint and format **`tldr`** pages.

More information: https://github.com/tldr-pages/tldr-lint.

- Lint all pages:

  `tldr-lint {{pages_directory}}`

- Format a specific page to stdout:

  `tldr-lint --format {{page.md}}`

- Format all pages in place:

  `tldr-lint --format --in-place {{pages_directory}}`

tldr

Displays simple help pages for command-line tools, from the tldr-pages project.

More information: https://tldr.sh.

- Show the tldr page for a command (hint: this is how you got here!):

  `tldr {{command}}`

- Show the tldr page for `cd`, overriding the default platform:

  `tldr -p {{android|linux|osx|sunos|windows}} {{cd}}`

- Show the tldr page for a subcommand:

  `tldr {{git-checkout}}`

- Update local pages (if the client supports caching):

  `tldr -u`

tldrl

This command is an alias of `tldr-lint`

- View documentation for the original command:

  `tldr tldr-lint`

tlmgr

Manages packages and configuration options of an existing TeX Live installation.

More information: https://www.tug.org/texlive/tlmgr.html.

- Install a package and its dependencies:

  ```
  tlmgr install {{package}}
  ```

- Remove a package and its dependencies:

  ```
  tlmgr remove {{package}}
  ```

- Display information about a package:

  ```
  tlmgr info {{package}}
  ```

- Update all packages:

  ```
  tlmgr update --all
  ```

- Show possible updates without updating anything:

  ```
  tlmgr update --list
  ```

- Start a GUI version of tlmgr:

  ```
  tlmgr gui
  ```

- List all TeX Live configurations:

  ```
  tlmgr conf
  ```

tmpmail

A temporary email right from your terminal written in P O S IX sh.

More information: https://github.com/sdushantha/tmpmail.

- Create a temporary inbox:

  ```
  tm pm ail - - g enerate
  ```

- List messages and their numeric ID:

  ```
  tm pm ail
  ```

- Display the most recent received email:

  ```
  tm pm ail - - recent
  ```

- O pen a specific message:

  ```
  tm pm ail {{em ail_id}}
  ```

- V iew email as raw text without H T ML tags:

  ```
  tm pm ail - - text
  ```

- O pen email with a specific browser (default is w3 m):

  ```
  tm pm ail - - b row ser {{b row ser}}
  ```

tmux

Terminal multiplexer. It allows multiple sessions with windows, panes, and more.

More information: https://github.com/tmux/tmux.

- Start a new session:

  ```
  tmux
  ```

- Start a new named session:

  ```
  tmux new -s {{name}}
  ```

- List existing sessions:

  ```
  tmux ls
  ```

- Attach to the most recently used session:

  ```
  tmux attach
  ```

- Detach from the current session (inside a tmux session):

  ```
  Ctrl-B d
  ```

- Create a new window (inside a tmux session):

  ```
  Ctrl-B c
  ```

- Switch between sessions and windows (inside a tmux session):

  ```
  Ctrl-B w
  ```

- Kill a session by name:

  ```
  tmux kill-session -t {{name}}
  ```

tmuxinator

Create and manage tmux sessions easily.

More information: https://github.com/tmuxinator/tmuxinator.

- Create a new project:

  ```
  tmuxinator new {{project}}
  ```

- Edit a project:

  ```
  tmuxinator edit {{project}}
  ```

- List projects:

  ```
  tmuxinator list
  ```

- Start a tmux session based on project:

  ```
  tmuxinator start {{project}}
  ```

- Stop a project's tmux session:

  ```
  tmuxinator stop {{project}}
  ```

tokei

A program that prints out statistics about code.

More information: https://github.com/XAMPPRocky/tokei

- Get a report on the code in a directory and all subdirectories:

  ```
  tokei {{path/to/directory}}
  ```

- Get a report for a directory excluding `.min.js` files:

  ```
  tokei {{path/to/directory}} -e {{*.min.js}}
  ```

- Print out statistics for individual files in a directory:

  ```
  tokei {{path/to/directory}} --files
  ```

- Get a report for all files of type Rust and Markdown:

  ```
  tokei {{path/to/directory}} -t={{Rust}},{{Markdown}}
  ```

topgrade

Update all applications on the system.

More information: https://github.com/r-darwish/topgrade.

- Run updates:

  ```
  topgrade
  ```

- Say yes to all updates:

  ```
  topgrade -y
  ```

- Cleanup temporary/old files:

  ```
  topgrade -c
  ```

- Disable a certain update operation:

  ```
  topgrade -disable {{operation}}
  ```

- Only perform a certain update operation:

  ```
  topgrade --only {{operation}}
  ```

- Edit the config file with default editor:

  ```
  topgrade --edit-config
  ```

topydo

A todo list application that uses the todo.txt format.

More information: https://github.com/topydo/topydo.

- Add a todo to a specific project with a given context:

  ```
  topydo add "{{todo_message}} +{{project_name}} @{{context_name}}"
  ```

- Add a todo with a due date of tomorrow with a priority of `A`:

  ```
  topydo add "(A) {{todo_message}} due:{{1d}}"
  ```

- Add a todo with a due date of friday:

  ```
  topydo add "{{todo_message}} due:{{fri}}"
  ```

- Add a non-strict repeating todo (next due = now + rec):

  ```
  topydo add "water flowers due:{{mon}} rec:{{1w}}"
  ```

- Add a strict repeating todo (next due = currentdue + rec):

  ```
  topydo add "{{todo_message}} due:{{2020-01-01}} rec:{{+1m
  ```

- Revert the last `topydo` command executed:

  ```
  topydo revert
  ```

touch

Change a file access and modification times (atime, mtime).

More information: https://www.gnu.org/software/coreutils/touch.

- Create a new empty file(s) or change the times for existing file(s) to current time:

  ```
  touch {{filename}}
  ```

- Set the times on a file to a specific date and time:

  ```
  touch -t {{YYYYMMDDHHMM.SS}} {{filename}}
  ```

- Use the times from a file to set the times on a second file:

  ```
  touch -r {{filename}} {{filename2}}
  ```

- Create multiple files:

  ```
  touch {{file{1,2,3}.txt}}
  ```

tox

Automate Python testing across multiple Python versions.

Use tox.ini to configure environments and test command.

More information: https://github.com/tox-dev/tox.

- Run tests on all test environments:

  ```
  tox
  ```

- Create a `tox.ini` configuration:

  ```
  tox-quickstart
  ```

- List the available environments:

  ```
  tox --listenvs-all
  ```

- Run tests on a specific environment (e.g. python 3.6):

  ```
  tox -e {{py36}}
  ```

- Force the virtual environment to be recreated:

  ```
  tox --recreate -e {{py27}}
  ```

tpp

Command-Line based presentation tool.

More information: https://github.com/cbbrowne/tpp.

- View a presentation:

  ```
  tpp {{filename}}
  ```

- Output a presentation:

  ```
  tpp -t {{type}} -o {{outputname}} {{filename}}
  ```

tput

View and modify terminal settings and capabilities.

- Move the cursor to a screen location:

  ```
  tpu t cu p {{y_coordinate}} {{x_coordinate}}
  ```

- Set foreground (af) or background (ab) color:

  ```
  tpu t {{setaf| setab }} {{ansi_color_code}}
  ```

- Show number of columns, lines, or colors:

  ```
  tpu t {{cols| lines| colors}}
  ```

- Ring the terminal bell:

  ```
  tpu t b el
  ```

- Reset all terminal attributes:

  ```
  tpu t sg r0
  ```

- Enable / Disable word wrap:

  ```
  tpu t {{sm am | rm am }}
  ```

tr

Translate characters: run replacements based on single characters and character sets.

More information: https://www.gnu.org/software/coreutils/tr.

- Replace all occurrences of a character in a file, and print the result:

```
tr {{find_character}} {{replace_character}} < {{filename}}
```

- Replace all occurrences of a character from another command's output:

```
echo {{text}} | tr {{find_character}} {{replace_character}}
```

- Map each character of the first set to the corresponding character of the second set:

```
tr '{{abcd}}' '{{jkmn}}' < {{filename}}
```

- Delete all occurrences of the specified set of characters from the input:

```
tr -d '{{input_characters}}' < {{filename}}
```

- Compress a series of identical characters to a single character:

```
tr -s '{{input_characters}}' < {{filename}}
```

- Translate the contents of a file to upper-case:

```
tr "[:lower:]" "[:upper:]" < {{filename}}
```

- Strip out non-printable characters from a file:

```
tr -cd "[:print:]" < {{filename}}
```

traceroute

Print the route packets trace to network host.

- Traceroute to a host:

  ```
  traceroute {{host}}
  ```

- Disable IP address and host name mapping:

  ```
  traceroute -n {{host}}
  ```

- Specify wait time for response:

  ```
  traceroute -w {{0.5}} {{host}}
  ```

- Specify number of queries per hop:

  ```
  traceroute -q {{5}} {{host}}
  ```

- Specify size in bytes of probing packet:

  ```
  traceroute {{host}} {{42}}
  ```

traefik

A H T T P reverse proxy and load balancer.

More information: https://traefik.io.

- S tart server with default config:

  ```
  traefik
  ```

- S tart server with a custom config file:

  ```
  traefik - - c {{config _file}}. tom l
  ```

- S tart server with cluster mode enabled:

  ```
  traefik - - clu ster
  ```

- S tart server with web U I enabled:

  ```
  traefik - - w eb
  ```

trans

Translate Shell is a command-line translator.

More information: https://github.com/soimort/translate-shell.

- Translate a word (language is detected automatically):

  ```
  trans "{{word_or_sentence_to_translate}}"
  ```

- Get a brief translation:

  ```
  trans --brief "{{word_or_sentence_to_translate}}"
  ```

- Translate a word into french:

  ```
  trans :{{fr}} {{word}}
  ```

- Translate a word from German to English:

  ```
  trans {{de}}:{{en}} {{Schmetterling}}
  ```

- Behave like a dictionary to get the meaning of a word:

  ```
  trans -d {{word}}
  ```

transcode

Transcode video and audio codecs, and convert between media formats.

- Create stabilisation file to be able to remove camera shakes:

  ```
  transcode -J stabilize -i {{input_file}}
  ```

- Remove camera shakes after creating stabilisation file, transform video using xvid:

  ```
  transcode -J transform -i {{input_file}} -y xvid -o
  {{output_file}}
  ```

- Resize the video to 640x480 pixels and convert to MPEG4 codec using xvid:

  ```
  transcode -Z 640x480 -i {{input_file}} -y xvid -o
  {{output_file}}
  ```

transcrypt

Transparently encrypt files within a Git repository.

More information: https://github.com/elasticdog/transcrypt.

- Initialize an unconfigured repository:

  ```
  transcrypt
  ```

- List the currently encrypted files:

  ```
  git ls-crypt
  ```

- Display the credentials of a configured repository:

  ```
  transcrypt --display
  ```

- Initialize and decrypt a fresh clone of a configured repository:

  ```
  transcrypt --cipher={{cipher}}
  ```

- Rekey to change the encryption cipher or password:

  ```
  transcrypt --rekey
  ```

transmission-cli

A lightweight, command-line BitTorrent client.

This tool has been deprecated, please see `transmission-remote`.

More information: https://transmissionbt.com.

- Download a specific torrent:

```
transmission-cli {{url|magnet|path/to/file}}
```

- Download a torrent to a specific directory:

```
transmission-cli --download-dir {{path/to/download_directory}} {{url|magnet|path/to/file}}
```

- Create a torrent file from a specific file or directory:

```
transmission-cli --new {{path/to/source_file_or_directory}}
```

- Set the download speed limit to 50 KB/s:

```
transmission-cli --downlimit {{50}} {{url|magnet|path/to/file}}
```

- Set the upload speed limit to 50 KB/s:

```
transmission-cli --uplimit {{50}} {{url|magnet|path/to/file}}
```

- Use a specific port for connections:

```
transmission-cli --port {{port_number}} {{url|magnet|path/to/file}}
```

- Force encryption for peer connections:

```
transmission-cli --encryption-required {{url|magnet|path/to/file}}
```

- Use a Bluetack-formatted peer blocklist:

```
transmission-cli --blocklist {{blocklist_url|path/to/blocklist}} {{url|magnet|path/to/file}}
```

transmission-create

A CLI utility to create BitTorrent .torrent files.

More information: https://manned.org/transmission-create.

- Create a torrent with 2048 KB as the piece size:

```
transmission-create -o {{path/to/example.torrent}} --tracker {{tracker_announce_url}} --piecesize {{2048}} {{path/to/file_or_directory}}
```

- Create a private torrent with a 2048 KB piece size:

```
transmission-create -p -o {{path/to/example.torrent}} --tracker {{tracker_announce_url}} --piecesize {{2048}} {{path/to/file_or_directory}}
```

- Create a torrent with a comment:

```
transmission-create -o {{path/to/example.torrent}} --tracker {{tracker_url1}} -c {{comment}} {{path/to/file_or_directory}}
```

- Create a torrent with multiple trackers:

```
transmission-create -o {{path/to/example.torrent}} --tracker {{tracker_url1}} --tracker {{tracker_url2}} {{path/to/file_or_directory}}
```

- Show help page:

```
transmission-create --help
```

transmission-remote

Remote control utility for transmission-daemon and transmission.

More information: https://transmissionbt.com.

- Add a torrent file or magnet link to Transmission and download to a specified directory:

```
transmission-remote {{hostname}} -a {{torrent|url}} -w {{/path/to/download_directory}}
```

- Change the default download directory:

```
transmission-remote {{hostname}} -w {{/path/to/download_directory}}
```

- List all torrents:

```
transmission-remote {{hostname}} --list
```

- Start torrent 1 and 2, stop torrent 3:

```
transmission-remote {{hostname}} -t "{{1,2}}" --start -t {{3}} --stop
```

- Remove torrent 1 and 2, and also delete local data for torrent 2:

```
transmission-remote {{hostname}} -t {{1}} --remove -t {{2}} --remove-and-delete
```

- Stop all torrents:

```
transmission-remote {{hostname}} -t {{all}} --stop
```

- Move torrents 1-10 and 15-20 to a new directory (which will be created if it does not exist):

```
transmission-remote {{hostname}} -t "{{1-10,15-20}}" --m
{{/path/to/new_directory}}
```

trash-cli

A command-line interface to the trashcan APIs.

More information: https://github.com/andreafrancia/trash-cli.

- Trash files and directories:

  ```
  trash-put {{filename}}
  ```

- Empty the trashcan:

  ```
  trash-empty
  ```

- List trashed files:

  ```
  trash-list
  ```

- Restore a trashed file by choosing a number from the list that results from this command:

  ```
  trash-restore
  ```

- Remove individual files from the trashcan:

  ```
  trash-rm {{filename}}
  ```

travis

Command-line client to interface with Travis CI.

More information: https://github.com/travis-ci/travis.rb.

- Display the client version:

  `travis version`

- Authenticate the CLI client against the server, using an authentication token:

  `travis login`

- List repositories the user has permissions on:

  `travis repos`

- Encrypt values in `.travis.yml`:

  `travis encrypt {{token}}`

- Generate a `.travis.yml` file and enable the project:

  `travis init`

trawl

Prints out network interface information to the console, much like ifconfig/ipconfig/ip/ifdata.

More information: https://github.com/robphoenix/trawl.

- Show column names:

  ```
  trawl -n
  ```

- Filter interface names using a case insensitive regular expression:

  ```
  trawl -f wi
  ```

- Get a list of available interfaces:

  ```
  trawl -i
  ```

- Include the loopback interface:

  ```
  trawl -l
  ```

true

Returns a successful exit status code of 0.

Use this with the `||` operator to make a command always exit with 0.

More information: https://www.gnu.org/software/coreutils/true.

- Return a successful exit code:

```
true
```

truncate

Shrink or extend the size of a file to the specified size.

More information: https://www.gnu.org/software/coreutils/truncate.

- Set a size of 10 GB to an existing file, or create a new file with the specified size:

```
truncate --size {{10G}} {{filename}}
```

- Extend the file size by 50 M, fill with holes (which reads as zero bytes):

```
truncate --size +{{50M}} {{filename}}
```

- Shrink the file by 2 GiB, by removing data from the end of file:

```
truncate --size -{{2G}} {{filename}}
```

- Empty the file's content:

```
truncate --size 0 {{filename}}
```

- Empty the file's content, but do not create the file if it does not exist:

```
truncate --no-create --size 0 {{filename}}
```

ts-node

Run TypeScript code directly, without any compiling.

More information: https://www.npmjs.com/package/ts-node.

- Execute a TypeScript file without compiling (`node` + `tsc`):

  ```
  ts-node {{path/to/file.ts}}
  ```

- Execute a TypeScript file without loading `tsconfig.json`:

  ```
  ts-node --skip-project {{path/to/file.ts}}
  ```

- Evaluate TypeScript code passed as a literal on the command-line:

  ```
  ts-node --eval '{{console.log("Hello World")}}'
  ```

- Execute a TypeScript file in script mode:

  ```
  ts-node --script-mode {{path/to/file.ts}}
  ```

- Transpile a TypeScript file to JavaScript without executing it:

  ```
  ts-node --transpile-only {{path/to/file.ts}}
  ```

- Display TS-Node help:

  ```
  ts-node --help
  ```

tsc

TypeScript compiler.

More information: https://www.typescriptlang.org/docs/handbook/compiler-options.html.

- Compile a TypeScript file `foobar.ts` into a JavaScript file `foobar.js`:

  ```
  tsc {{foobar.ts}}
  ```

- Compile a TypeScript file into JavaScript using a specific target syntax (default is `ES3`):

  ```
  tsc --target {{ES5|ES2015|ES2016|ES2017|ES2018|
  {{foobar.ts}}
  ```

- Compile a TypeScript file into a JavaScript file with a custom name:

  ```
  tsc --outFile {{output.js}} {{input.ts}}
  ```

- Compile all `.ts` files of a TypeScript project defined in a `tsconfig.json` file:

  ```
  tsc --build {{tsconfig.json}}
  ```

- Run the compiler using command-line options and arguments fetched from a text file:

  ```
  tsc @{{args.txt}}
  ```

- Type-check multiple JavaScript files, and output only the errors:

  ```
  tsc --allowJs --checkJs --noEmit {{src/**/*.js}}
  ```

tslint

A pluggable linting utility for TypeScript.

More information: https://palantir.github.io/tslint.

- Create tslint config:

  ```
  tslint --init
  ```

- Lint on a given set of files:

  ```
  tslint {{filename}}.js {{filename1}}.js
  ```

- Fix lint issues:

  ```
  tslint --fix
  ```

- Lint with the config file in the project root:

  ```
  tslint --project {{path/to/project_root}}
  ```

tsort

P erform a topological sort.

A common use is to show the dependency order of nodes in a directed acyclic graph.

More information: https://www.gnu.org/software/coreutils/tsort.

- P erform a topological sort consistent with a partial sort per line of input separated by blanks:

  ```
  tsort {{file}}
  ```

tty

Returns terminal name.

More information: https://www.gnu.org/software/coreutils/tty.

- Print the file name of this terminal:

```
tty
```

tuir

A text user-interface (T U I) to view and interact with R eddit from your terminal.

N avigate with the V im keys.

More information: https://gitlab.com/aj_ak/tuir

- Launch tuir:

  ```
  tu ir
  ```

- O pen a subreddit:

  ```
  /{{su b reddit_nam e}}
  ```

- O pen a link:

  ```
  o
  ```

- O pen a specific subreddit on launch:

  ```
  tu ir - s {{su b reddit_nam e}}
  ```

- O pen external links using programs defined in the mailcap config:

  ```
  tu ir - - enab le- m edia
  ```

twm

A window manager for the X Window system.

More information: https://gitlab.freedesktop.org/xorg/app/twm.

- Connect to the default X server:

  ```
  twm
  ```

- Connect to a specific X server:

  ```
  twm -display {{display}}
  ```

- Only manage the default screen:

  ```
  twm -s
  ```

- Use a specific startup file:

  ```
  twm -f {{path/to/file}}
  ```

- Enable verbose mode and print unexpected errors in X:

  ```
  twm -v
  ```

twopi

Render an image of a **radial** network graph from a **graphviz** file.

Layouts: **dot**, **neato**, **twopi**, **circo**, **fdp**, **sfdp**, **osage** & **patchwork**

More information: https://graphviz.org/doc/info/command.html

- Render a `png` image with a filename based on the input filename and output format (uppercase -O):

  ```
  twopi -T {{png}} -O {{path/to/input.gv}}
  ```

- Render a `svg` image with the specified output filename (lowercase -o):

  ```
  twopi -T {{svg}} -o {{path/to/image.svg}} {{path/to/input.gv}}
  ```

- Render the output in `ps`, `pdf`, `svgz`, `fig`, `png`, `gif`, `jpg`, `jpeg`, or `dot` format:

  ```
  twopi -T {{format}} -O {{path/to/input.gv}}
  ```

- Render a `gif` image using stdin and stdout:

  ```
  echo "{{digraph {this -> that}}}" | twopi -T {{gif}} > {{path/to/image.gif}}
  ```

- Display help:

  ```
  twopi -?
  ```

type

Display the kind of command the shell will execute.

- Display the kind of a command:

  ```
  type {{com m and}}
  ```

- Display all locations containing the specified executable:

  ```
  type - a {{com m and}}
  ```

- Display the name of the disk file that would be executed:

  ```
  type - p {{com m and}}
  ```

ufraw-batch

Convert RAW files from cameras into standard image files.

- Simply convert RAW files to jpg:

  ```
  ufraw-batch --out-type=jpg {{input_file(s)}}
  ```

- Simply convert RAW files to png:

  ```
  ufraw-batch --out-type=png {{input_file(s)}}
  ```

- Extract the preview image from the raw file:

  ```
  ufraw-batch --embedded-image {{input_file(s)}}
  ```

- Save the file with size up to the given maximums MAX1 and MAX2:

  ```
  ufraw-batch --size=MAX1,MAX2 {{input_file(s)}}
  ```

ulimit

Get and set user limits.

- Get the properties of all the user limits:

  ```
  ulimit -a
  ```

- Get hard limit for the number of simultaneously opened files:

  ```
  ulimit -Hn
  ```

- Get soft limit for the number of simultaneously opened files:

  ```
  ulimit -Sn
  ```

- Set max per-user process limit:

  ```
  ulimit -u 30
  ```

umask

Manage the read/write/execute permissions that are masked out (i.e. restricted) for newly created files by the user.

- Display the current mask in octal notation:

  ```
  u m ask
  ```

- Display the current mask in symbolic (human-readable) mode:

  ```
  u m ask  - S
  ```

- Change the mask symbolically to allow read permission for all users (the rest of the mask bits are unchanged):

  ```
  u m ask  {{a+ r}}
  ```

- S et the mask (using octal) to restrict no permissions for the file' s owner, and restrict all permissions for everyone else:

  ```
  u m ask  {{0 7 7 }}
  ```

umount

U nlink a filesystem from its mount point, making it no longer accessible.

A filesystem cannot be unmounted when it is busy.

- U nmount a filesystem, by passing the path to the source it is mounted from:

  `u m ou nt {{path/to/dev ice_file}}`

- U nmount a filesystem, by passing the path to the target where it is mounted:

  `u m ou nt {{path/to/m ou nted_directory}}`

- U nmount all mounted filesystems (except the`proc` filesystem):

  `u m ou nt - a`

unalias

R emove aliases.

- R emove an alias:

  ```
  u nalias {{alias_nam e}}
  ```

- R emove all aliases:

  ```
  u nalias - a
  ```

uname

Print details about the current machine and the operating system running on it.

See also: `lsb_release`

More information: https://www.gnu.org/software/coreutils/uname.

- Print kernel name:

  `uname`

- Print system architecture and processor information:

  `uname --machine --processor`

- Print kernel name, kernel release and kernel version:

  `uname --kernel-name --kernel-release --kernel-version`

- Print system hostname:

  `uname --nodename`

- Print all available system information:

  `uname --all`

unar

Extract contents from archive files.

- Extract an archive to the current directory:

  ```
  unar {{archive}}
  ```

- Extract an archive to the specified directory:

  ```
  unar -o {{path/to/directory}} {{archive}}
  ```

- Force overwrite if files to be unpacked already exist:

  ```
  unar -f {{archive}}
  ```

- Force rename if files to be unpacked already exist:

  ```
  unar -r {{archive}}
  ```

- Force skip if files to be unpacked already exist:

  ```
  unar -s {{archive}}
  ```

unclutter

H ides the mouse cursor.

- H ide mouse cursor after 3 seconds:

  ```
  u nclu tter - idle {{3 }}
  ```

uncrustify

C, C++, C#, D, Java and Pawn source code formatter.

More information: https://github.com/uncrustify/uncrustify.

- Format a single file:

```
uncrustify -f {{path/to/file.cpp}} -o {{path/to/output.cpp}}
```

- Read filenames from stdin, and take backups before writing output back to the original filepaths:

```
find . -name "*.cpp" | uncrustify -F - --replace
```

- Don't make backups (useful if files are under version control):

```
find . -name "*.cpp" | uncrustify -F - --no-backup
```

- Use a custom configuration file and write the result to stdout:

```
uncrustify -c {{path/to/uncrustify.cfg}} -f {{path/to/file.cpp}}
```

- Explicitly set a configuration variable's value:

```
uncrustify --set {{option}}={{value}}
```

- Generate a new configuration file:

```
uncrustify --update-config -o {{path/to/new.cfg}}
```

unexpand

Convert spaces to tabs.

More information: https://www.gnu.org/software/coreutils/unexpand.

- Convert blanks in each file to tabs, writing to standard output:

  ```
  u nexpand {{file}}
  ```

- Convert blanks to tabs, reading from standard output:

  ```
  u nexpand
  ```

- Convert all blanks, instead of j ust initial blanks:

  ```
  u nexpand - a {{file}}
  ```

- Convert only leading seq uences of blanks (overrides -a):

  ```
  u nexpand - - first- only {{file}}
  ```

- H ave tabs a certain number of characters apart, not 8 (enables -a):

  ```
  u nexpand - t {{nu m b er}} {{file}}
  ```

uniq

Output the unique lines from the given input or file.

Since it does not detect repeated lines unless they are adjacent, we need to sort them first.

More information: https://www.gnu.org/software/coreutils/uniq.

- Display each line once:

```
sort {{file}} | uniq
```

- Display only unique lines:

```
sort {{file}} | uniq -u
```

- Display only duplicate lines:

```
sort {{file}} | uniq -d
```

- Display number of occurrences of each line along with that line:

```
sort {{file}} | uniq -c
```

- Display number of occurrences of each line, sorted by the most frequent:

```
sort {{file}} | uniq -c | sort -nr
```

unison

Bidirectional file synchronisation tool.

More information: https://www.cis.upenn.edu/~bcpierce/unison/download/releases/stable/unison-manual.html.

- Sync two directories (creates log first time these two directories are synchronised):

```
unison {{path/to/directory_1}} {{path/to/directory_2}}
```

- Automatically accept the (non-conflicting) defaults:

```
unison {{path/to/directory_1}} {{path/to/directory_2}} -auto
```

- Ignore some files using a pattern:

```
unison {{path/to/directory_1}} {{path/to/directory_2}} -ignore {{pattern}}
```

- Show documentation:

```
unison -doc {{topics}}
```

units

Provide the conversion between two units of measure.

More information: https://www.gnu.org/software/units/.

- Run in interactive mode:

  ```
  units
  ```

- List all units containing a specific string in interactive mode:

  ```
  search {{string}}
  ```

- Show the conversion between two simple units:

  ```
  units {{quarts}} {{tablespoons}}
  ```

- Convert between units with quantities:

  ```
  units "{{15 pounds}}" {{kilograms}}
  ```

- Show the conversion between two compound units:

  ```
  units "{{meters / second}}" "{{inches / hour}}"
  ```

- Show the conversion between units with different dimensions:

  ```
  units "{{acres}}" "{{ft^2}}"
  ```

- Show the conversion of byte multipliers:

  ```
  units "{{15 megabytes}}" {{bytes}}
  ```

unlink

Remove a link to a file from the filesystem.

The file contents is lost if the link is the last one to the file.

More information: https://www.gnu.org/software/coreutils/unlink.

- Remove the specified file if it is the last link:

```
unlink {{path/to/file}}
```

unrar

Extract RAR archives.

- Extract files with original directory structure:

  ```
  unrar x {{compressed.rar}}
  ```

- Extract files to a specified path with the original directory structure:

  ```
  unrar x {{compressed.rar}} {{path/to/extract}}
  ```

- Extract files into current directory, losing directory structure in the archive:

  ```
  unrar e {{compressed.rar}}
  ```

- Test integrity of each file inside the archive file:

  ```
  unrar t {{compressed.rar}}
  ```

- List files inside the archive file without decompressing it:

  ```
  unrar l {{compressed.rar}}
  ```

unzip

Extract compressed files in a ZIP archive.

- Extract zip file(s) (for multiple files, separate file paths by spaces):

```
unzip {{file(s)}}
```

- Extract zip files(s) to given path:

```
unzip {{compressed_file(s)}} -d {{path/to/put/extracted_file(s)}}
```

- List the contents of a zip file without extracting:

```
unzip -l {{file.zip}}
```

- Extract the contents of the file(s) to stdout alongside the extracted file names:

```
unzip -c {{file.zip}}
```

- Extract a zip file created on Windows, containing files with non-ASCII (e.g. Chinese or Japanese characters) filenames:

```
unzip -O {{gbk}} {{file.zip}}
```

uptime

Tell how long the system has been running and other information.

More information: https://www.gnu.org/software/coreutils/uptime.

- Print current time, uptime, number of logged-in users and other information:

  ```
  uptime
  ```

- Show only the amount of time the system has been booted for:

  ```
  uptime --pretty
  ```

- Print the date and time the system booted up at:

  ```
  uptime --since
  ```

- Show version information:

  ```
  uptime --version
  ```

upx

Compress or decompress executables.

More information: https://upx.github.io.

- Compress executable:

  ```
  u px {{file}}
  ```

- Decompress executable:

  ```
  u px - d {{file}}
  ```

- Detailed help:

  ```
  u px - - help
  ```

users

Display a list of logged in users.

More information: https://www.gnu.org/software/coreutils/users.

- Display a list of logged in users:

  ```
  u sers
  ```

- Display a list of logged in users according to a specific file:

  ```
  u sers {{/v ar/log /w m tp}}
  ```

uudecode

Decode files encoded by **`u u en code`**

More information: https://manned.org/uudecode.

- Decode a file that was encoded with `u u encode` and print the result to stdout:

  `u u decode {{path/to/encoded_file}}`

- Decode a file that was encoded with `u u encode` and write the result to a file:

  `u u decode - o {{path/to/decoded_file}} {{path/to/encoded_file}}`

uuencode

Encode binary files into ASCII for transport via mediums that only support simple ASCII encoding.

More information: https://manned.org/uuencode.

- Encode a file and print the result to stdout:

```
uuencode {{path/to/input_file}} {{output_file_name_after_decoding}}
```

- Encode a file and and write the result to a file:

```
uuencode -o {{path/to/output_file}} {{path/to/input_file}} {{output_file_name_after_decoding}}
```

- Encode a file using Base64 instead of the default uuencode encoding and and write the result to a file:

```
uuencode -m -o {{path/to/output_file}} {{path/to/input_file}} {{output_file_name_after_decoding}}
```

uvicorn

Python ASGI HTTPS server, for asynchronous projects.

More information: https://www.uvicorn.org/.

- Run Python web app:

  ```
  uvicorn {{import path: app_object}}
  ```

- Listen on port 8080 on localhost:

  ```
  uvicorn --host {{localhost}} --port {{8080}}
  {{import path: app_object}}
  ```

- Turn on live reload:

  ```
  uvicorn --reload {{import path: app_object}}
  ```

- Use 4 worker processes for handling requests:

  ```
  uvicorn --workers {{4}} {{import path: app_object}}
  ```

- Run app over HTTPS:

  ```
  uvicorn --ssl-certfile {{cert.pem}} --ssl-keyfile {{key.pem}
  {{import path: app_object}}
  ```

vagrant

Manage lightweight, reproducible, and portable development environments.

More information: https://www.vagrantup.com.

- Create Vagrantfile in current directory with the base Vagrant box:

  ```
  vagrant init
  ```

- Create Vagrantfile with the Ubuntu 14.04 (Trusty Tahr) box from HashiCorp Atlas:

  ```
  vagrant init ubuntu/trusty32
  ```

- Start and provision the vagrant environment:

  ```
  vagrant up
  ```

- Suspend the machine:

  ```
  vagrant suspend
  ```

- Halt the machine:

  ```
  vagrant halt
  ```

- Connect to machine via SSH:

  ```
  vagrant ssh
  ```

- Output the SSH configuration file of the running Vagrant machine:

  ```
  vagrant ssh-config
  ```

vala

Vala code runner.

Tutorial: https://wiki.gnome.org/Projects/Vala/Tutorial

More information: https://valadoc.org/.

- Run a vala file, with gtk+:

  ```
  vala {{path/to/file.vala}} --pkg {{gtk+-3.0}}
  ```

- Display version info:

  ```
  vala --version
  ```

- Display helper message:

  ```
  vala --help
  ```

valac

Vala code compiler.

Tutorial: https://wiki.gnome.org/Projects/Vala/Tutorial

More information: https://valadoc.org/.

- Compile a vala file, with gtk+ :

```
valac {{path/to/file.vala}} --pkg {{gtk+-3.0}}
```

- Display version info:

```
valac --version
```

- Display helper message:

```
valac --help
```

valgrind

Wrapper for a set of expert tools for profiling, optimiz ing and debugging programs.

Commonly used tools include **memch eck**, **ca ch eg ri**, **ca llg ri n**, **ma s s i f**, **h elg ri n**, and **drd**.

More information: http://www.valgrind.org.

- U se the (default) Memcheck tool to show a diagnostic of memory usage by `prog ram`:

  ```
  v alg rind {{prog ram }}
  ```

- U se Memcheck to report all possible memory leaks of `prog ram` in full detail:

  ```
  v alg rind - - leak - check = fu ll - - show - leak - k inds= all {{prog ram
  ```

- U se the Cachegrind tool to profile and log CP U cache operations of `prog ram`:

  ```
  v alg rind - - tool= cacheg rind {{prog ram }}
  ```

- U se the Massif tool to profile and log heap memory and stack usage of `prog ram`:

  ```
  v alg rind - - tool= m assif - - stack s= yes {{prog ram }}
  ```

var-dump-server

Symfony dump server.

Collects data dumped by the Symfony VarDumper component.

More information: https://symfony.com/doc/current/components/var_dumper.html#the-dump-server

- Start the server:

  ```
  var-dump-server
  ```

- Dump the data in an HTML file:

  ```
  var-dump-server --format=html > {{path/to/file.html}}
  ```

- Make the server listen on a specific address and port:

  ```
  var-dump-server --host {{127.0.0.1:9912}}
  ```

vault

A CLI to interact with HashiCorp Vault.

More information: https://www.vaultproject.io/docs/commands

- Connect to a Vault server and initialize a new encrypted data store:

  ```
  vault init
  ```

- Unseal (unlock) the vault, by providing one of the key shares needed to access the encrypted data store:

  ```
  vault unseal {{key-share-x}}
  ```

- Authenticate the CLI client against the Vault server, using an authentication token:

  ```
  vault auth {{authentication_token}}
  ```

- Store a new secret in the vault, using the generic back-end called "secret":

  ```
  vault write secret/{{hello}} value={{world}}
  ```

- Read a value from the vault, using the generic back-end called "secret":

  ```
  vault read secret/{{hello}}
  ```

- Read a specific field from the value:

  ```
  vault read -field={{field_name}} secret/{{hello}}
  ```

- Seal (lock) the Vault server, by removing the encryption key of the data store from memory:

  ```
  vault seal
  ```

V B oxManage

Command-line interface to V irtualB ox.

Includes all the functionality of the G U I and more.

More information: https://www.virtualbox.org/manual/ch0 8 .html# vboxmanage-intro

- List all V irtualB ox virtual machines:

  ```
  V B oxM anag e list v m s
  ```

- S how information about a particular virtual machine:

  ```
  V B oxM anag e show v m info {{nam e| u u id}}
  ```

- S tart a virtual machine:

  ```
  V B oxM anag e startv m {{nam e| u u id}}
  ```

- S tart a virtual machine in headless mode:

  ```
  V B oxM anag e startv m {{nam e| u u id}} - type headless
  ```

- S hutdown the virtual machine and save its current state:

  ```
  V B oxM anag e controlv m {{nam e| u u id}} sav estate
  ```

- S hutdown down the virtual machine without saving its state:

  ```
  V B oxM anag e controlv m {{nam e| u u id}} pow eroff
  ```

- U pdate V B ox extension packs:

  ```
  V B oxM anag e extpack install - - replace
  {{V b oxE xtensionP ack F ileN am e}}
  ```

vcsh

Version Control System for the home directory using Git repositories.

More information: https://github.com/RichiH/vcsh

- Initialize an (empty) repository:

  `vcsh init {{repository_name}}`

- Clone a repository into a custom directory name:

  `vcsh clone {{git_url}} {{repository_name}}`

- List all managed repositories:

  `vcsh list`

- Execute a Git command on a managed repository:

  `vcsh {{repository_name}} {{git_command}}`

- Push/pull all managed repositories to/from remotes:

  `vcsh {{push|pull}}`

- Write a custom `.gitignore` for a managed repository:

  `vcsh write-gitignore {{repository_name}}`

vdir

List directory contents.

Drop-in replacement for `ls -l`

More information: https://www.gnu.org/software/coreutils/vdir.

- List files and directories in the current directory, one per line, with details:

  ```
  vdir
  ```

- List with sizes displayed in human readable units (KB, MB, GB):

  ```
  vdir -h
  ```

- List including hidden files (starting with a dot):

  ```
  vdir -a
  ```

- List files and directories sorting entries by size (largest first):

  ```
  vdir -S
  ```

- List files and directories sorting entries by modification time (newest first):

  ```
  vdir -t
  ```

- List grouping directories first:

  ```
  vdir --group-directories-first
  ```

- Recursively list all files and directories in a specific directory:

  ```
  vdir --recursive {{path/to/directory}}
  ```

vectoriz e-pixelart

Convert P N G pixel art graphics to S V G /E P S vector images.

More information: https://github.com/und3 f/vectoriz e-pixelart

- Convert a P N G image to a vector image format:

```
v ectorize- pixelart {{path/to/inpu t. png }} {{path/to/
ou tpu t. sv g | . eps}}
```

vegeta

A command-line utility and a library for HTTP load testing.

See also: `ab`

More information: https://github.com/tsenart/vegeta.

- Launch an attack lasting 30 seconds:

```
echo "{{GET https://example.com}}" | vegeta attack -duration={{30s}}
```

- Launch an attack on a server with a self-signed https certificate:

```
echo "{{GET https://example.com}}" | vegeta attack -insecure -duration={{30s}}
```

- Launch an attack with a rate of 10 requests per second:

```
echo "{{GET https://example.com}}" | vegeta attack -duration={{30s}} -rate={{10}}
```

- Launch an attack and display a report:

```
echo "{{GET https://example.com}}" | vegeta attack -duration={{30s}} | vegeta report
```

- Launch an attack and plot the results on a graph (latency over time):

```
echo "{{GET https://example.com}}" | vegeta attack -duration={{30s}} | vegeta plot > {{path/to/results.html}}
```

- Launch an attack against multiple URLs from a file:

```
vegeta attack -duration={{30s}} -targets={{requests.txt}} | vegeta report
```

vela

Command-line tools for the V ela pipeline.

More information: https://go-vela.github.io/docs/reference/cli/.

- T rigger a pipeline to run from a G it branch, commit or tag:

  ```
  v ela  add  deploym ent  - - org  {{org anization}} - - repo
  {{repository_nam e}} - - targ et {{env ironm ent}} - - ref {{b ranch|
  com m it| refs/tag s/g it_tag }} - - description
  " {{deploy_description}}"
  ```

- List deployments for a repository:

  ```
  v ela  g et  deploym ent  - - org  {{org anization}} - - repo
  {{repository_nam e}}
  ```

- Inspect a specific deployment:

  ```
  v ela  v iew  deploym ent  - - org  {{org anization}} - - repo
  {{repository_nam e}} - - deploym ent {{deploym ent_nu m b er}}
  ```

velero

Backup and migrate Kubernetes applications and their persistent volumes.

More information: https://github.com/heptio/velero.

- Create a backup containing all resources:

```
velero backup create {{backup_name}}
```

- List all backups:

```
velero backup get
```

- Delete a backup:

```
velero backup delete {{backup_name}}
```

- Create a weekly backup, each living for 90 days (2160 hours):

```
velero schedule create {{schedule_name}} --
schedules="{{@every 7d}}" --ttl {{2160h0m0s}}
```

- Create a restore from the latest successful backup triggered by specific schedule:

```
velero restore create --from-schedule {{schedule_name}}
```

vercel

Deploy and manage your V ercel deployments.

More information: https://vercel.com/docs/cli.

- Deploy the current directory:

  ```
  v ercel
  ```

- Deploy the current directory to production:

  ```
  v ercel - - prod
  ```

- Deploy a directory:

  ```
  v ercel {{path/to/proj ect}}
  ```

- Initializ e an example proj ect:

  ```
  v ercel init
  ```

- Deploy with E nvironment V ariables:

  ```
  v ercel - - env {{E N V }}= {{v ar}}
  ```

- B uild with E nvironment V ariables:

  ```
  v ercel - - b u ild- env {{E N V }}= {{v ar}}
  ```

- S et default regions to enable the deployment on:

  ```
  v ercel - - reg ions {{reg ion_id}}
  ```

- R emove a deployment:

  ```
  v ercel rem ov e {{proj ect_nam e}}
  ```

view

A read-only version of `vim`

This is equivalent to `vim -R`

- Open a file:

  ```
  view {{file}}
  ```

vim

Vim (Vi IMproved), a command-line text editor, provides several modes for different kinds of text manipulation.

Pressing `i` enters insert mode. `<Esc>` enters normal mode, which enables the use of Vim commands.

More information: https://www.vim.org.

- Open a file:

  ```
  vim {{path/to/file}}
  ```

- View Vim's help manual:

  ```
  :help<Enter>
  ```

- Save and Quit:

  ```
  :wq<Enter>
  ```

- Open a file at a specified line number:

  ```
  vim +{{line_number}} {{path/to/file}}
  ```

- Undo the last operation:

  ```
  u
  ```

- Search for a pattern in the file (press `n`/`N` to go to next/previous match):

  ```
  /{{search_pattern}}<Enter>
  ```

- Perform a regular expression substitution in the whole file:

  ```
  :%s/{{regular_expression}}/{{replacement}}/g<Enter>
  ```

- Display the line numbers:

  ```
  :set nu<Enter>
  ```

vimdiff

Open up two or more files in vim and show the differences between them.

See also **`vim`**

More information: https://www.vim.org.

- Open two files and show the differences:

  `vimdiff {{file1}} {{file2}}`

- Move the cursor to the window on the left|right:

  `Ctrl + w {{h|l}}`

- Jump to the next difference:

  `[c`

- Jump to the previous difference:

  `]c`

- Copy the highlighted difference from the other window to the current window:

  `do`

- Copy the highlighted difference from the current window to the other window:

  `dp`

- Update all highlights and folds:

  `:diffupdate`

- Toggle the highlighted code fold:

  `za`

vimtutor

V im tutor, teaching the basic vim commands.

- Launch the vim tutor using the given language (en, fr, de, ...):

  ```
  v im tu tor {{lang u ag e}}
  ```

- E xit the tutor:

  ```
  < E sc>  : q  < E nter>
  ```

virsh-connect

Connect to a virtual machine hypervisor.

See also: `virsh`

More information: https://manned.org/virsh.

- Connect to the default hypervisor:

  ```
  virsh connect
  ```

- Connect as root to the local QEMU/KVM hypervisor:

  ```
  virsh connect qemu:///system
  ```

- Launch a new instance of the hypervisor and connect to it as the local user:

  ```
  virsh connect qemu:///session
  ```

- Connect as root to a remote hypervisor using ssh:

  ```
  virsh connect qemu+ssh://{{user_name@host_name}}/system
  ```

virsh-domblklist

List information about block devices associated with a virtual machine.

See also: `virsh`.

More information: https://manned.org/virsh.

- List the target name and source path of the block devices:

  ```
  virsh domblklist --domain {{vm_name}}
  ```

- List the disk type and device value as well as the target name and source path:

  ```
  virsh domblklist --domain {{vm_name}} --details
  ```

virsh-help

Display information about **`virsh`** commands or command groups.

See also: **`virsh`**.

More information: https://manned.org/virsh.

- List the `virsh` commands grouped into related categories:

  ```
  virsh help
  ```

- List the command categories:

  ```
  virsh help | grep "keyword"
  ```

- List the commands in a category:

  ```
  virsh help {{category_keyword}}
  ```

- Show help for a command:

  ```
  virsh help {{command}}
  ```

virsh-list

List the ID, name, and state of virtual machines.

See also: `virsh`.

More information: https://manned.org/virsh.

- List information about running virtual machines:

  `virsh list`

- List information about virtual machines regardless of state:

  `virsh list --all`

- List information about virtual machines with autostart either enabled or disabled:

  `virsh list --all --{{autostart|no-autostart}}`

- List information about virtual machines either with or without snapshots:

  `virsh list --all --{{with-snapshot|without-snapshot}}`

virsh pool-autostart

Enable or disable autostart for a virtual machine storage pool.

See also: `virsh`.

More information: https://manned.org/virsh.

- Enable autostart for the storage pool specified by name or UUID (determine using `virsh pool-list`):

  `virsh pool-autostart --pool {{name|uuid}}`

- Disable autostart for the storage pool specified by name or UUID:

  `virsh pool-autostart --pool {{name|uuid}} --disable`

virsh pool-build

Build the underlying storage system for a virtual machine storage pool as defined in it's configuration file in **`/etc/libvirt/storage`**.

See also: **`virsh`**, **`virsh pool-define`**, **`virsh pool-start`**.

More information: https://manned.org/virsh.

- Build the storage pool specified by name or UUID (determine using `virsh pool-list`):

  `virsh pool-build --pool {{name|uuid}}`

virsh pool-define-as

Create a configuration file in `/etc/libvirt/storage` for a persistent virtual machine storage pool from the provided arguments.

See also: `virsh`, `virsh pool-build`, `virsh pool-start`.

More information: https://manned.org/virsh.

- Create the configuration file for a storage pool called pool_name using `/var/vms` as the underlying storage system:

```
virsh pool-define-as --name {{pool_name}} --type {{dir}} --target {{/var/vms}}
```

virsh pool-delete

Delete the underlying storage system of an inactive virtual machine storage pool.

See also: `virsh`, `virsh pool-destroy`, `virsh pool-undefine`.

More information: https://manned.org/virsh.

- Delete the underlying storage system for the storage pool specified by name or UUID (determine using `virsh pool-list`):

  ```
  virsh pool-delete --pool {{name|uuid}}
  ```

virsh pool-destroy

Stop an active virtual machine storage pool.

See also: `virsh`, `virsh pool-delete`

More information: https://manned.org/virsh.

- Stop a storage pool specified by name or UUID (determine using `virsh pool-list`):

  ```
  virsh pool-destroy --pool {{name|uuid}}
  ```

virsh pool-info

List information about a virtual machine storage pool.

See also: `virsh`.

More information: https://manned.org/virsh.

- List the name, UUID, state, persistence type, autostart status, capacity, space allocated, and space available for the storage pool specified by name or UUID (determine using `virsh pool-list`):

  `virsh pool-info --pool {{name|uuid}}`

virsh pool-list

List information about virtual machine storage pools.

See also: `virsh`, `virsh pool-autostart`, `virsh pool-define-as`

More information: https://manned.org/virsh.

- List the name, state, and whether autostart is enabled or disabled for active storage pools:

  ```
  virsh pool-list
  ```

- List information for active and inactive or just inactive storage pools:

  ```
  virsh pool-list --{{all|inactive}}
  ```

- List extended information about persistence, capacity, allocation, and available space for active storage pools:

  ```
  virsh pool-list --details
  ```

- List information for active storage pools with either autostart enabled or disabled:

  ```
  virsh pool-list --{{autostart|no-autostart}}
  ```

- List information for active storage pools that are either persistent or transient:

  ```
  virsh pool-list --{{persistent|transient}}
  ```

- List the name and UUID of active storage pools:

  ```
  virsh pool-list --name --uuid
  ```

virsh pool-start

Start a previously configured but inactive virtual machine storage pool.

See also: `virsh`, `virsh pool-define`, `virsh pool-destroy`.

More information: https://manned.org/virsh.

- Start the storage pool specified by name or UUID (determine using `virsh pool-list`) and create the underlying storage system if it doesn't exist:

  `virsh pool-start --pool {{name|uuid}} --build`

virsh pool-undefine

Delete the configuration file in `/ et c/ li bvi rt / s t ora ge` for a stopped virtual machine storage pool.

S ee also: `vi rs`, `vi rs h - pool- des t roy`.

More information: https://manned.org/virsh.

- Delete the configuration for the storage pool specified name or U U ID (determine using `v irsh pool- lis t`):

  ```
  v irsh pool- u ndefine - - pool {{nam e| u u id}}
  ```

virsh-undefine

Delete a virtual machine.

More information: https://manned.org/virsh.

- Delete only the virtual machine configuration file:

```
v irsh u ndefine - - dom ain {{v m _nam e}}
```

- Delete the configuration file and all associated storage volumes:

```
v irsh u ndefine - - dom ain {{v m _nam e}} - - rem ov e- all- storag e
```

- Delete the configuration file and the specified storage volumes using the target name or the source name (as obtained from the `v irsh dom b lk list` command):

```
v irsh u ndefine - - dom ain {{v m _nam e}} - - storag e {{sda, path/to/
sou rce}}
```

virsh

Manage virsh guest domains.

N O T E : ' guest_ id' can be the id, name or U U ID of the guest.

More information: https://libvirt.org/virshcmdref.html.

- Connect to a hypervisor session:

  ```
  v irsh connect {{q em u : ///system }}
  ```

- List all domains:

  ```
  v irsh list - - all
  ```

- Dump guest configuration file:

  ```
  v irsh du m pxm l {{g u est_id}} > {{path/to/g u est. xm l}}
  ```

- Create a guest from a configuration file:

  ```
  v irsh create {{path/to/config _file. xm l}}
  ```

- E dit a guest' s configuration file (editor can be changed with $ E DIT O R):

  ```
  v irsh edit {{g u est_id}}
  ```

- S tart/reboot/shutdown/suspend/resume a guest:

  ```
  v irsh {{com m and}} {{g u est_id}}
  ```

- S ave the current state of a guest to a file:

  ```
  v irsh sav e {{g u est_id}} {{filenam e}}
  ```

- Delete a running guest:

  ```
  v irsh destroy {{g u est_id}} & & v irsh u ndefine {{g u est_id}}
  ```

virt-clone

Clone a libvirt virtual machine.

More information: https://manned.org/virt-clone.

- Clone a virtual machine and automatically generate a new name, storage path, and MAC address:

```
v irt- clone - - orig inal {{v m _nam e}} - - au to- clone
```

- Clone a virtual machine and specify the new name, storage path, and MAC address:

```
v irt- clone - - orig inal {{v m _nam e}} - - nam e {{new _v m _nam e}} - -
file {{path/to/new _storag e}} - - m ac {{ff: ff: ff: ff: ff: ff|
R A N D O M }}
```

virt-install

Create virtual machines with libvirt and begin OS installation.

More information: https://virt-manager.org/.

- Create a virtual machine with 1 GiB RAM and 12 GiB storage and start Debian installation:

```
virt-install --name {{vm_name}} --memory {{1024}} --disk
path={{path/to/image.qcow2}},size={{12}} --cdrom {{path/to/
debian.iso}}
```

virt-sparsify

Make virtual machine drive images thin-provisioned.

N O T E : U se only for offl ine machines to avoid data corruption.

H ome page:https://libguestfs.org/.

- Create a sparsified compressed image without snapshots from an unsparsified one:

  ```
  v irt- sparsify - - com press {{path/to/im ag e. q cow 2 }} {{path/to/
  im ag e_new . q cow 2 }}
  ```

- S parsify an image in-place:

  ```
  v irt- sparsify - - in- place {{path/to/im ag e. im g }}
  ```

virt-sysprep

Reset, unconfigure, or customize a virtual machine image.

More information: https://manned.org/virt-sysprep.

- List all supported operations (enabled operations are indicated with asterisks):

  ```
  virt-sysprep --list-operations
  ```

- Run all enabled operations but don't actually apply the changes:

  ```
  virt-sysprep --domain {{vm_name}} --dry-run
  ```

- Run only the specified operations:

  ```
  virt-sysprep --domain {{vm_name}} --operations
  {{operation1,operation2,...}}
  ```

- Generate a new `/etc/machine-id` file and enable customizations to be able to change the host name to avoid network conflicts:

  ```
  virt-sysprep --domain {{vm_name}} --enable {{customizations}}
  --hostname {{host_name}} --operation {{machine-id}}
  ```

virtualenv

Create virtual isolated Python environments.

More information: https://virtualenv.pypa.io/.

- Create a new environment:

  `virtualenv {{path/to/venv}}`

- Customize the prompt prefix:

  `virtualenv --prompt={{prompt_prefix}} {{path/to/venv}}`

- Use a different version of Python with virtualenv:

  `virtualenv --python={{path/to/pythonbin}} {{path/to/venv}}`

- Start (select) the environment:

  `source {{path/to/venv}}/bin/activate`

- Stop the environment:

  `deactivate`

virtualenvwrapper

Group of simple wrapper commands for Python's **`virtualenv`** tool.

More information: http://virtualenvwrapper.readthedocs.org.

- Create a new Python `virtualenv` in `$WORKON_HOME`

  `mkvirtualenv {{virtualenv_name}}`

- Create a `virtualenv` for a specific Python version:

  `mkvirtualenv --python {{/usr/local/bin/python3.8}} {{virtualenv_name}}`

- Activate or use a different `virtualenv`:

  `workon {{virtualenv_name}}`

- Stop the `virtualenv`:

  `deactivate`

- List all virtual environments:

  `lsvirtualenv`

- Remove a `virtualenv`:

  `rmvirtualenv {{virtualenv_name}}`

- Get summary of all virtualenvwrapper commands:

  `virtualenvwrapper`

visudo

Safely edit the sudoers file.

- Edit sudoers file:

  ```
  sudo visudo
  ```

- Check sudoers file for errors:

  ```
  sudo visudo -c
  ```

vladimyr

Dario Vladović's personal CLI.

More information: https://github.com/vladimyr/vladimyr-cli.

- Start Dario's interactive CLI:

  ```
  vladimyr
  ```

vlc

Cross-platform multimedia player.

More information: https://www.videolan.org/vlc/.

- Play a file:

```
vlc {{path/to/file}}
```

- Play in fullscreen:

```
vlc --fullscreen {{path/to/file}}
```

- Play muted:

```
vlc --no-audio {{path/to/file}}
```

- Play repeatedly:

```
vlc --loop {{path/to/file}}
```

- Play video from a URL:

```
vlc {{https://www.youtube.com/watch?v=oHg5SJYRHA0}}
```

vsce

Extension manager for Visual Studio Code.

More information: https://github.com/microsoft/vscode-vsce.

- List all the extensions created by a publisher:

  `vsce list {{publisher}}`

- Publish an extension as major, minor or patch version:

  `vsce publish {{major|minor|patch}}`

- Unpublish an extension:

  `vsce unpublish {{extension_id}}`

- Package the current working directory as a `.vsix` file:

  `vsce package`

- Show the metadata associated with an extension:

  `vsce show {{extension_id}}`

vue build

A subcommand provided by `@vue/cli` and `@vue/cli-service-global` that enables quick prototyping.

More information: https://cli.vuejs.org/guide/prototyping.html

- Build a `.js` or `.vue` file in production mode with zero config:

  ```
  vue build {{filename}}
  ```

vue init

Legacy project initialization subcommand of the Vue.js framework.

More information: https://cli.vuejs.org/guide/creating-a-project.html#pulling-2-x-templates-legacy.

- Create a new project using one of the default templates:

  ```
  vue init {{webpack|webpack-simple|browserify|browserify-simple}} {{project name}}
  ```

- Create a new project using a local template:

  ```
  vue init {{path/to/template directory}} {{project name}}
  ```

- Create a new project using a template from GitHub:

  ```
  vue init {{username}}/{{repo}} {{project name}}
  ```

vue serve

A subcommand provided by `@vue/cli` and `@vue/cli-service-global` that enables quick prototyping.

More information: https://cli.vuejs.org/guide/prototyping.html

- Serve a `.js` or `.vue` file in development mode with zero config:

  ```
  vue serve {{filename}}
  ```

vue

Multi-purpose CLI for Vue.js.

More information: https://cli.vuejs.org

- Create a new vue project interactively:

  ```
  vue create {{project_name}}
  ```

- Create a new project with web UI:

  ```
  vue ui
  ```

W

S how who is logged on and what they are doing.

P rint user login, T T Y , remote host, login time, idle time, current process.

- S how logged-in users info:

  ```
  w
  ```

- S how logged-in users info without a header:

  ```
  w -h
  ```

w3 m

A text-based web browser.

S upports S S L and mouse input, even over S S H .

More information: [http://w3 m.sourceforge.net](http://w3m.sourceforge.net)

- O pen a U R L:

  ```
  w 3 m {{http: //exam ple. com }}
  ```

- O pen a U R L in monochrome mode:

  ```
  w 3 m {{http: //exam ple. com }} - M
  ```

- O pen a U R L without mouse support:

  ```
  w 3 m {{http: //exam ple. com }} - - no- m ou se
  ```

- O pen a new browser tab:

  ```
  S hift + T
  ```

- Display your browser history:

  ```
  C trl + H
  ```

- Q uit w3 m:

  ```
  ' q ' then ' y'
  ```

wait

Wait for a process to complete before proceeding.

- Wait for a process to finish given its process ID (P ID) and return its exit status:

  ```
  w ait {{pid}}
  ```

- Wait for all processes known to the invoking shell to finish:

  ```
  w ait
  ```

waitress-serve

Pure Python WSGI HTTP Server.

- Run a Python web app:

```
waitress-serve {{import path.wsgi_function}}
```

- Listen on port 8080 on localhost:

```
waitress-serve --listen={{localhost}}:{{8080}}
{{import path.wsgi_function}}
```

- Start waitress on a Unix socket:

```
waitress-serve --unix-socket={{path/to/socket}}
{{import path.wsgi_function}}
```

- Use 4 threads to process requests:

```
waitress-serve --threads={{4}} {{import path.wsgifunc}}
```

- Call a factory method that returns a WSGI object:

```
waitress-serve --call {{import path.wsgi_factory}}
```

- Set the URL scheme to https:

```
waitress-serve --url-scheme={{https}}
{{import path.wsgi_function}}
```

wal

A tool to create color schemes based on the dominant colors of a wallpaper.

More information: https://github.com/dylanaraps/pywal.

- Preview color scheme:

```
wal --preview {{image.png}}
```

- Create color scheme:

```
wal -i {{image.png}}
```

- Create a light color scheme:

```
wal -il {{image.png}}
```

- Skip setting the desktop wallpaper:

```
wal -in {{image.png}}
```

- Skip setting the terminal colors:

```
wal -is {{image.png}}
```

- Restore the previously generated color scheme and wallpaper:

```
wal -R
```

wapm

The WebAssembly package manager.

More information: https://wapm.io/help/reference.

- Interactively create a new `wapm.toml` file:

  `wapm init`

- Download all the packages listed as dependencies in `wapm.toml`

  `wapm install`

- Download a specific version of a package and add it to the list of dependencies in wapm.toml:

  `wapm install {{package_name}}@{{version}}`

- Download a package and install it globally:

  `wapm install --global {{package_name}}`

- Uninstall a package and remove it from the list of dependencies in `wapm.toml`

  `wapm uninstall {{package_name}}`

- Print a tree of locally-installed dependencies:

  `wapm list`

- List top-level globally installed packages:

  `wapm list --global`

- Execute a package command using the Wasmer runtime:

  `wapm run {{command_name}} {{arguments}}`

wasm-objdump

Display information from WebAssembly binaries.

- Display the section headers of a given binary:

  ```
  wasm-objdump -h {{file.wasm}}
  ```

- Display the entire disassembled output of a given binary:

  ```
  wasm-objdump -d {{file.wasm}}
  ```

- Display the details of each section:

  ```
  wasm-objdump --details {{file.wasm}}
  ```

- Display the details of a given section:

  ```
  wasm-objdump --section '{{import}}' --details {{file.wasm}
  ```

wasm-opt

Optimize WebAssembly binary files.

- Apply default optimizations and write to a given file:

  ```
  wasm-opt -O {{input.wasm}} -o {{output.wasm}}
  ```

- Apply all optimizations and write to a given file (takes more time, but generates optimal code):

  ```
  wasm-opt -O4 {{input.wasm}} -o {{output.wasm}}
  ```

- Optimize a file for size:

  ```
  wasm-opt -Oz {{input.wasm}} -o {{output.wasm}}
  ```

- Print the textual representation of the binary to console:

  ```
  wasm-opt {{input.wasm}} --print
  ```

wasm2 c

Convert a file from the WebAssembly binary format to a C source file and header.

- Convert a file to a C source file and header and display it to the console:

  ```
  w asm 2 c {{file. w asm }}
  ```

- Write the output to a given file (`file. h`gets additionally generated):

  ```
  w asm 2 c {{file. w asm }} - o {{file. c}}
  ```

wasm2wat

Convert a file from the WebAssembly binary format to the text format.

- Convert a file to the text format and display it to the console:

  ```
  wasm2wat {{file.wasm}}
  ```

- Write the output to a given file:

  ```
  wasm2wat {{file.wasm}} -o {{file.wat}}
  ```

wat2wasm

Convert a file from the WebAssembly text format to the binary format.

More information: https://github.com/WebAssembly/wabt.

- Parse and check a file for errors:

```
wat2wasm {{file.wat}}
```

- Write the output binary to a given file:

```
wat2wasm {{file.wat}} -o {{file.wasm}}
```

- Display simplified representation of every byte:

```
wat2wasm -v {{file.wat}}
```

watch

Execute a program periodically, showing output fullscreen.

- Repeatedly run a command and show the result:

  ```
  watch {{command}}
  ```

- Re-run a command every 60 seconds:

  ```
  watch -n {{60}} {{command}}
  ```

- Monitor the contents of a directory, highlighting differences as they appear:

  ```
  watch -d {{ls -l}}
  ```

watchexec

Run arbitrary commands when files change.

More information: https://github.com/watchexec/watchexec.

- Call `ls -la` when any file in the current directory changes:

  ```
  watchexec -- {{ls -la}}
  ```

- Run `make` when any JavaScript, CSS and HTML files in the current directory change:

  ```
  watchexec --exts {{js,css,html}} make
  ```

- Run `make` when any file in the `lib` or `src` subdirectories change:

  ```
  watchexec --watch {{lib}} --watch {{src}} {{make}}
  ```

- Call/restart `my_server` when any file in the current directory change, sending `SIGKILL` to stop the child process:

  ```
  watchexec --restart --signal {{SIGKILL}} {{my_server}}
  ```

watson

A wonderful CLI to track your time.

More information: https://github.com/TailorDev/Watson

- Start monitoring time in project:

  `watson start {{project}}`

- Start monitoring time in project with tags:

  `watson start {{project}} +{{tag}}`

- Stop monitoring time for the current project:

  `watson stop`

- Display latest working sessions:

  `watson log`

- Edit most recent frame:

  `watson edit`

- Remove most recent frame:

  `watson remove`

wc

Count lines, words, or bytes.

More information: https://www.gnu.org/software/coreutils/wc.

- Count lines in file:

```
wc -l {{file}}
```

- Count words in file:

```
wc -w {{file}}
```

- Count characters (bytes) in file:

```
wc -c {{file}}
```

- Count characters in file (taking multi-byte character sets into account):

```
wc -m {{file}}
```

- Use standard input to count lines, words and characters (bytes) in that order:

```
{{find .}} | wc
```

weasyprint

Render HTML to PDF or PNG.

More information: https://weasyprint.org/.

- Render a HTML file to PDF:

```
weasyprint {{path/to/input.html}} {{path/to/output}}.pdf
```

- Render a HTML file to PNG, including an additional user stylesheet:

```
weasyprint {{path/to/input.html}} {{path/to/output}}.png --stylesheet {{path/to/stylesheet.css}}
```

- Output additional debugging information when rendering:

```
weasyprint {{path/to/input.html}} {{path/to/output}}.pdf --verbose
```

- Specify a custom resolution when outputting to PNG:

```
weasyprint {{path/to/input.html}} {{path/to/output}}.png --resolution {{300}}
```

- Specify a base URL for relative URLs in the input HTML file:

```
weasyprint {{path/to/input.html}} {{path/to/output}}.png --base-url {{url_or_filename}}
```

web-ext

A command-line tool for managing web extension development.

More information: https://www.npmjs.com/package/web-ext

- Run the web extension in the current directory in Firefox:

  ```
  web-ext run
  ```

- Run a web extension from a specific directory in Firefox:

  ```
  web-ext run --source-dir {{path/to/directory}}
  ```

- Display verbose execution output:

  ```
  web-ext run --verbose
  ```

- Run a web extension in Firefox Android:

  ```
  web-ext run --target firefox-android
  ```

- Lint the manifest and source files for errors:

  ```
  web-ext lint
  ```

- Build and package the extension:

  ```
  web-ext build
  ```

- Display verbose build output:

  ```
  web-ext build --verbose
  ```

- Sign a package for self-hosting:

  ```
  web-ext sign --api-key {{api_key}} --api-secret {{api_secret}}
  ```

webpack

Bundle a web project's js files and other assets into a single output file.

More information: https://webpack.js.org

- Create a single output file from an entry point file:

  ```
  webpack {{app.js}} {{bundle.js}}
  ```

- Load css files too from the js file (this uses the css loader for `.css` files):

  ```
  webpack {{app.js}} {{bundle.js}} --module-bind '.css=.css'
  ```

- Pass a config file (with eg. the entry script and the output filename) and show compilation progress:

  ```
  webpack --config {{webpack.config.js}} --progress
  ```

- Automatically recompile on changes to project files:

  ```
  webpack --watch {{app.js}} {{bundle.js}}
  ```

webtorrent

The command-line interface for WebTorrent.

Supports magnets, urls, info hashes and .torrent files.

More information: https://github.com/webtorrent/webtorrent-cli.

- Download a torrent:

  `webtorrent download "{{torrent_id}}"`

- Stream a torrent to VLC media player:

  `webtorrent download "{{torrent_id}}" --vlc`

- Stream a torrent to a Digital Living Network Alliance (DLNA) device:

  `webtorrent download "{{torrent_id}}" --dlna`

- Display a list of files for a specific torrent:

  `webtorrent download "{{torrent_id}}" --select`

- Specify a file index from the torrent to download:

  `webtorrent download "{{torrent_id}}" --select {{index}}`

- Seed a specific file or directory:

  `webtorrent seed {{path/to/file_or_directory}}`

- Create a new torrent file for the specified file path:

  `webtorrent create {{path/to/file}}`

- Display information for a magnet URI or `torrent` file:

  `webtorrent info {{path/to/file_or_magnet}}`

wget

Download files from the Web.

S upports H T T P , H T T P S , and F T P .

More information: https://www.gnu.org/software/wget.

- Download the contents of an U R L to a file (named " foo" in this case):

```
w g et {{https: //exam ple. com /foo}}
```

- Download the contents of an U R L to a file (named " bar" in this case):

```
w g et - - ou tpu t- docu m ent {{b ar}} {{https: //exam ple. com /foo}}
```

- Download a single web page and all its resources with 3 -second intervals between req uests (scripts, stylesheets, images, etc.):

```
w g et - - pag e- req u isites - - conv ert- link s - - w ait= 3 {{https: /
exam ple. com /som epag e. htm l}}
```

- Download all listed files within a directory and its sub-directories (does not download embedded page elements):

```
w g et - - m irror - - no- parent {{https: //exam ple. com /som epath/}}
```

- Limit the download speed and the number of connection retries:

```
w g et - - lim it- rate= {{3 0 0 k }} - - tries= {{1 0 0 }} {{https: //
exam ple. com /som epath/}}
```

- Download a file from an H T T P server using B asic Auth (also works for F T P):

```
w g et - - u ser= {{u sernam e}} - - passw ord= {{passw ord}} {{https: //
exam ple. com }}
```

- Continue an incomplete download:

```
w g et - - continu e {{https: //exam ple. com }}
```

- Download all U R Ls stored in a text file to a specific directory:

```
w g et - - directory- prefix {{path/to/directory}} - - inpu t- file
{{U R L s. txt}}
```

where

Reports all known instances of a command.

It could be an executable in the PATH environment variable, an alias, or a shell builtin.

- Find all instances of a command:

  ```
  where {{command}}
  ```

which

Locate a program in the user's path.

- Search the PATH environment variable and display the location of any matching executables:

  ```
  which {{executable}}
  ```

- If there are multiple executables which match, display all:

  ```
  which -a {{executable}}
  ```

while

Simple shell loop.

- Read stdin and perform an action on every line:

  ```
  while read line; do echo "$line"; done
  ```

- Execute a command forever once every second:

  ```
  while :; do {{command}}; sleep 1; done
  ```

who

Display who is logged in and related data (processes, boot time).

More information: https://www.gnu.org/software/coreutils/who.

- Display the username, line, and time of all currently logged-in sessions:

  ```
  w ho
  ```

- Display information only for the current terminal session:

  ```
  w ho am  i
  ```

- Display all available information:

  ```
  w ho - a
  ```

- Display all available information with table headers:

  ```
  w ho - a - H
  ```

whoami

P rint the username associated with the current effective user ID.

More information: https://www.gnu.org/software/coreutils/whoami.

- Display currently logged username:

  ```
  w hoam i
  ```

- Display the username after a change in the user ID:

  ```
  su do w hoam i
  ```

whois

Command-line client for the WHOIS (RFC3912) protocol.

More information: https://github.com/rfc1036/whois

- Get information about a domain name:

  ```
  whois {{example.com}}
  ```

- Get information about an IP address:

  ```
  whois {{8.8.8.8}}
  ```

- Get abuse contact for an IP address:

  ```
  whois -b {{8.8.8.8}}
  ```

wordgrinder

Command-line word processor.

More information: https://cowlark.com/wordgrinder.

- Start wordgrinder (loads a blank document by default):

  ```
  wordgrinder
  ```

- Open a given file:

  ```
  wordgrinder {{filename}}
  ```

- Show the menu:

  ```
  Alt + M
  ```

wormhole

Get things from one computer to another, safely.

More information: https://magic-wormhole.readthedocs.io/en/latest/.

- Send a file:

  ```
  wormhole send {{path/to/file}}
  ```

- Receive a file:

  ```
  wormhole receive {{wormhole_code}}
  ```

- Send raw text:

  ```
  wormhole send
  ```

wp

The official command-line interface to manage WordPress instances.

More information: https://wp-cli.org/.

- Print information about the operating system, shell, PHP, and WP-CLI installation:

  ```
  wp --info
  ```

- Update WP-CLI:

  ```
  wp cli update
  ```

- Install and activate a WordPress plugin:

  ```
  wp plugin install {{plugin}} --activate
  ```

- Replace all instances of a string in the database:

  ```
  wp search-replace {{old_string}} {{new_string}}
  ```

- Import the contents of a WordPress Extended RSS (WXR) file:

  ```
  wp import {{path/to/file.xml}}
  ```

wpa_supplicant

Manage protected wireless networks.

- Join a protected wireless network:

```
wpa_supplicant -i {{interface}} -c {{path/to/
wpa_supplicant_conf.conf}}
```

- Join a protected wireless network and run it in a daemon:

```
wpa_supplicant -B -i {{interface}} -c {{path/to/
wpa_supplicant_conf.conf}}
```

wpscan

WordPress vulnerability scanner.

More information: https://github.com/wpscanteam/wpscan.

- Update the vulnerability database:

  ```
  wpscan --update
  ```

- Scan a WordPress website:

  ```
  wpscan --url {{url}}
  ```

- Scan a WordPress website, using random user agents and passive detection:

  ```
  wpscan --url {{url}} --stealthy
  ```

- Scan a WordPress website, checking for vulnerable plugins and specifying the path to the `wp-content` directory:

  ```
  wpscan --url {{url}} --enumerate {{vp}} --wp-content-dir {{remote/path/to/wp-content}}
  ```

- Scan a WordPress website through a proxy:

  ```
  wpscan --url {{url}} --proxy {{protocol://ip:port}} --proxy-auth {{username:password}}
  ```

- Perform user identifiers enumeration on a WordPress website:

  ```
  wpscan --url {{url}} --enumerate {{u}}
  ```

- Execute a password guessing attack on a WordPress website:

  ```
  wpscan --url {{url}} --usernames {{username|path/to/usernames.txt}} --passwords {{path/to/passwords.txt}} threads {{20}}
  ```

- Scan a WordPress website, collecting vulnerability data from the WPVulnDB (https://wpvulndb.com/):

  ```
  wpscan --url {{url}} --api-token {{token}}
  ```

wrangler

Cloudflare Workers command-line tool.

More information: https://developers.cloudflare.com/workers/

- Initialize a project with a skeleton configuration:

  ```
  wrangler init {{project_name}}
  ```

- Authenticate with Cloudflare:

  ```
  wrangler login
  ```

- Start a local development server:

  ```
  wrangler dev --host {{hostname}}
  ```

- Publish the worker script:

  ```
  wrangler publish
  ```

- Aggregate logs from the production worker:

  ```
  wrangler tail
  ```

write

Write a message on the terminal of a specified logged in user (ctrl-C to stop writing messages).

Use the `who` command to find out all terminal_ids of all active users active on the system. See also `mesg`

- Send a message to a given user on a given terminal id:

  ```
  write {{username}} {{terminal_id}}
  ```

- Send message to "testuser" on terminal `/dev/tty/5`

  ```
  write {{testuser}} {{tty/5}}
  ```

- Send message to "johndoe" on pseudo terminal `/dev/pts/5`

  ```
  write {{johndoe}} {{pts/5}}
  ```

wrk

HTTP benchmarking tool.

More information: https://github.com/wg/wrk.

- Run a benchmark for 30 seconds, using 12 threads, and keeping 400 HTTP connections open:

```
wrk -t{{12}} -c{{400}} -d{{30s}} "{{http://127.0.0.1:8
index.html}}"
```

- Run a benchmark with a custom header:

```
wrk -t{{2}} -c{{5}} -d{{5s}} -H "{{Host:
example.com}}" "{{http://example.com/index.html}}"
```

- Run a benchmark with a request timeout of 2 seconds:

```
wrk -t{{2}} -c{{5}} -d{{5s}} --timeout {{2s}} "{{http://
example.com/index.html}}"
```

wuzz

Tool to interactively inspect HTTP requests and responses.

More information: https://github.com/asciimoo/wuzz

- Start wuzz:

  ```
  wuzz
  ```

- Display help information:

  ```
  F1
  ```

- Send an HTTP request:

  ```
  Ctrl+R
  ```

- Switch to the next view:

  ```
  Ctrl+J, Tab
  ```

- Switch to the previous view:

  ```
  Ctrl+K, Shift+Tab
  ```

x1 1 docker

S ecurely run G U I applications and desktop U Is in Docker containers.

S ee also **x eph y.r**

More information: [https://github.com/mviereck/x1 1 docker](https://github.com/mviereck/x1_1_docker)

- Launch V LC in a container:

```
x1 1 dock er - - pu lseau dio - - share= {{$ H O M E /V ideos}} {{j ess/v lc
```

- Launch X fce in a window:

```
x1 1 dock er - - desk top {{x1 1 dock er/xfce}}
```

- Launch G N O ME in a window:

```
x1 1 dock er - - desk top - - g pu - - init= {{system d}} {{x1 1 dock er/
g nom e}}
```

- Launch KDE P lasma in a window:

```
x1 1 dock er - - desk top - - g pu - - init= {{system d}} {{x1 1 dock er/k
plasm a}}
```

- Display help:

```
x1 1 dock er - - help
```

X_ X

V iew E xcel and CS V files from the command-line.

More information: https://github.com/kristianperkins/x_x

- V iew an X LS X or CS V file:

```
x_x {{file.xlsx|file.csv}}
```

- V iew an X LS X or CS V file, using the first row as table headers:

```
x_x -h {{0}} {{file.xlsx|file.csv}}
```

- V iew a CS V file with unconventional delimiters:

```
x_x --delimiter={{';'}} --quotechar={{'|'}} {{file.csv}}
```

xargs

Execute a command with piped arguments coming from another command, a file, etc.

The input is treated as a single block of text and split into separate pieces on spaces, tabs, newlines and end-of-file.

- Run a command using the input data as arguments:

  ```
  {{arguments_source}} | xargs {{command}}
  ```

- Run multiple chained commands on the input data:

  ```
  {{arguments_source}} | xargs sh -c "{{command1}} && {{command2}} | {{command3}}"
  ```

- Delete all files with a `.backup` extension (`-print0` uses a null character to split file names, and `-0` uses it as delimiter):

  ```
  find . -name {{'*.backup'}} -print0 | xargs -0 rm -v
  ```

- Execute the command once for each input line, replacing any occurrences of the placeholder (here marked as _) with the input line:

  ```
  {{arguments_source}} | xargs -I _ {{command}} _ {{optional_extra_arguments}}
  ```

- Parallel runs of up to `max-procs` processes at a time; the default is 1. If `max-procs` is 0, xargs will run as many processes as possible at a time:

  ```
  {{arguments_source}} | xargs -P {{max-procs}} {{command}}
  ```

xcaddy

The custom build tool for the Caddy Web Server.

More information: https://github.com/caddyserver/xcaddy.

- Build Caddy server from source:

  ```
  xcaddy build
  ```

- Build Caddy server with a specific version (defaults to latest):

  ```
  xcaddy build {{version}}
  ```

- Build Caddy with a specific module:

  ```
  xcaddy build --with {{module_name}}
  ```

- Build Caddy and output to a specific file:

  ```
  xcaddy build --output {{path/to/file}}
  ```

- Build and run Caddy for a development plugin in the current directory:

  ```
  xcaddy run
  ```

- Build and run Caddy for a development plugin using a specific Caddy config:

  ```
  xcaddy run --config {{path/to/file}}
  ```

XCV

Cut, copy, and paste in the command-line.

More information: https://github.com/busterc/xcv.

- Cut a file:

```
xcv x {{input_file}}
```

- Copy a file:

```
xcv c {{input_file}}
```

- Paste a file:

```
xcv v {{output_file}}
```

- List files available for pasting:

```
xcv l
```

xdg-user-dirs-update

Update XDG user directories.

More information: https://manpages.ubuntu.com/manpages/bionic/man1/xdg-user-dirs-update.1.html

- Change XDG's DESKTOP directory to the specified directory (must be absolute):

```
xdg-user-dirs-update --set DESKTOP "{{path/to/directory}}"
```

- Write the result to the specified dry-run-file instead of the `user-dirs.dirs` file:

```
xdg-user-dirs-update --dummy-output "{{path/to/dry_run_file}}" --set {{xdg_user_directory}} "{{path/to/directory}}"
```

X ephyr

A nested X server that runs as an X application.

- Create a black window with display ID " :2 " :

  ```
  X ephyr - b r - ac - noreset - screen {{8 0 0 x6 0 0 }} {{: 2 }}
  ```

- S tart an X application on the new screen:

  ```
  D I S P L A Y = : 2 {{com m and_nam e}}
  ```

xgettext

Extract gettext strings from code files.

More information: https://www.gnu.org/software/gettext/manual/html_node/xgettext-Invocation.html.

- Scan file and output strings to `messages.po`

  ```
  xgettext {{path/to/input_file}}
  ```

- Use a different output filename:

  ```
  xgettext --output {{path/to/output_file}} {{path/to/input_file}}
  ```

- Append new strings to an existing file:

  ```
  xgettext --join-existing --output {{path/to/output_file}} {{path/to/input_file}}
  ```

- Don't add a header containing metadata to the output file:

  ```
  xgettext --omit-header {{path/to/input_file}}
  ```

xh

Friendly and fast tool for sending HTTP requests.

More information: https://github.com/ducaale/xh.

- Send a GET request:

```
xh {{httpbin.org/get}}
```

- Send a POST request with a JSON body (key-value pairs are added to a top-level JSON object {e.g. `{"name": "john", "age": 25}`}):

```
xh post {{httpbin.org/post}} {{name=john}} {{age:=25}}
```

- Send a GET request with query parameters (e.g. `first_param=5&second_param=true`):

```
xh get {{httpbin.org/get}} {{first_param==5}}
{{second_param==true}}
```

- Send a GET request with a custom header:

```
xh get {{httpbin.org/get}} {{header-name:header-value}}
```

- Make a GET request and save the response body to a file:

```
xh --download {{httpbin.org/json}} --output {{path/to/file}}
```

xkcdpass

A flexible and scriptable password generator which generates strong passphrases.

Inspired by XKCD 936.

More information: https://github.com/redacted/XKCD-password-generator

- Generate one passphrase with the default options:

  ```
  xkcdpass
  ```

- Generate one passphrase whose first letters of each word match the provided argument:

  ```
  xkcdpass -a {{acrostic}}
  ```

- Generate passwords interactively:

  ```
  xkcdpass -i
  ```

xkill

Kill a window interactively in a graphical session.

See also **`kill`** and **`killall`**.

- Display a cursor to kill a window when pressing the left mouse button (press any other mouse button to cancel):

  ```
  xkill
  ```

- Display a cursor to select a window to kill by pressing any mouse button:

  ```
  xkill -button any
  ```

xmllint

XML parser and linter that supports XPath, a syntax for navigating XML trees.

- Return all nodes (tags) named "foo":

  ```
  xmllint --xpath "//{{foo}}" {{source_file.xml}}
  ```

- Return the contents of the first node named "foo" as a string:

  ```
  xmllint --xpath "string(//{{foo}})" {{source_file.xml}}
  ```

- Return the href attribute of the second anchor element in an html file:

  ```
  xmllint --html --xpath "string(//a[2]/@href)" web page.xht
  ```

- Return human-readable (indented) XML from file:

  ```
  xmllint --format {{source_file.xml}}
  ```

- Check that a XML file meets the requirements of its DOCTYPE declaration:

  ```
  xmllint --valid {{source_file.xml}}
  ```

- Validate XML against DTD schema hosted online:

  ```
  xmllint --dtdvalid {{URL}} {{source_file.xml}}
  ```

xmlto

Apply an XSL stylesheet to an XML document.

More information: https://pagure.io/xmlto.

- Convert a DocBook XML document to PDF format:

  ```
  xmlto {{pdf}} {{document.xml}}
  ```

- Convert a DocBook XML document to HTML format and store the resulting files in a separate directory:

  ```
  xmlto -o {{path/to/html_files}} {{html}} {{document.xml}}
  ```

- Convert a DocBook XML document to a single HTML file:

  ```
  xmlto {{html-nochunks}} {{document.xml}}
  ```

- Specify a stylesheet to use while converting a DocBook XML document:

  ```
  xmlto -x {{stylesheet.xsl}} {{output_format}}
  {{document.xml}}
  ```

XO

A pluggable, zero-configuration linting utility for JavaScript.

More information: https://github.com/xojs/xo

- Lint files in the "src" directory:

  ```
  xo
  ```

- Lint a given set of files:

  ```
  xo {{file1}}.js {{file2}}.js
  ```

- Automatically fix any lint issues found:

  ```
  xo --fix
  ```

- Lint using spaces as indentation instead of tabs:

  ```
  xo --space
  ```

- Lint using the "prettier" code style:

  ```
  xo --prettier
  ```

xonsh

Python-powered, cross-platform, Unix-gazing shell.

Write and mix sh/Python code in Xonsh (pronounced conch).

More information: https://xon.sh.

- Start an interactive shell session:

  ```
  xonsh
  ```

- Execute a single command and then exit:

  ```
  xonsh -c "{{command}}"
  ```

- Run commands from a script file and then exit:

  ```
  xonsh {{path/to/script_file.xonsh}}
  ```

- Define environment variables for the shell process:

  ```
  xonsh -D {{name1}}={{value1}} -D {{name2}}={{value2}}
  ```

- Load the specified `.xonsh` or `.json` configuration files:

  ```
  xonsh --rc {{path/to/file1.xonsh}} {{path/to/file2.json}}
  ```

- Skip loading the `.xonshrc` configuration file:

  ```
  xonsh --no-rc
  ```

xpdf

Portable Document Format (PDF) file viewer.

More information: https://www.xpdfreader.com/xpdf-man.html.

- Open a PDF file:

  ```
  xpdf {{path/to/file.pdf}}
  ```

- Open a specific page in a PDF file:

  ```
  xpdf {{path/to/file.pdf}} :{{page_number}}
  ```

- Open a compressed PDF file:

  ```
  xpdf {{path/to/file.pdf.tar}}
  ```

- Open a PDF file in fullscreen mode:

  ```
  xpdf -fullscreen {{path/to/file.pdf}}
  ```

- Specify the initial zoom:

  ```
  xpdf -z {{75}}% {{path/to/file.pdf}}
  ```

- Specify the initial zoom at page width or full page:

  ```
  xpdf -z {{page|width}} {{path/to/file.pdf}}
  ```

xplr

Terminal-based file system explorer.

More information: https://github.com/sayanarij it/xplr

- Open a directory:

  ```
  xplr {{path/to/directory}}
  ```

- Focus on a file:

  ```
  xplr {{path/to/file}}
  ```

XSV

A CSV command-line toolkit written in Rust.

More information: https://github.com/BurntSushi/xsv

- Inspect the headers of a file:

  ```
  xsv headers {{path/to/file.csv}}
  ```

- Count the number of entries:

  ```
  xsv count {{path/to/file.csv}}
  ```

- Get an overview of the shape of entries:

  ```
  xsv stats {{path/to/file.csv}} | xsv table
  ```

- Select a few columns:

  ```
  xsv select {{column_a,column_b}} {{path/to/file.csv}}
  ```

- Show 10 random entries:

  ```
  xsv sample {{10}} {{path/to/file.csv}}
  ```

- Join a column from one file to another:

  ```
  xsv join --no-case {{column_a}} {{path/to/file/a.csv}}
  {{column_b}} {{path/to/file/b.csv}} | xsv table
  ```

xxd

Create a hexadecimal representation (hexdump) from a binary file, or vice-versa.

- Generate a hexdump from a binary file and display the output:

  ```
  xxd {{input_file}}
  ```

- Generate a hexdump from a binary file and save it as a text file:

  ```
  xxd {{input_file}} {{output_file}}
  ```

- Display a more compact output, replacing consecutive zeros (if any) with a star:

  ```
  xxd -a {{input_file}}
  ```

- Display the output with 10 columns of one octet (byte) each:

  ```
  xxd -c {{10}} {{input_file}}
  ```

- Display output only up to a length of 32 bytes:

  ```
  xxd -l {{32}} {{input_file}}
  ```

- Display the output in plain mode, without any gaps between the columns:

  ```
  xxd -p {{input_file}}
  ```

- Revert a plaintext hexdump back into binary, and save it as a binary file:

  ```
  xxd -r -p {{input_file}} {{output_file}}
  ```

xxh

Bring your shell with all of your customizations through SSH sessions.

Note: xxh does not install anything into system directories on the target machine; removing `~/.xxh` will clear all traces of xxh on the target machine.

More information: https://github.com/xxh/xxh.

- Connect to a host and run the current shell:

```
xxh "{{host}}"
```

- Install the current shell into the target machine without prompting:

```
xxh "{{host}}" ++install
```

- Run the specified shell on the target machine:

```
xxh "{{host}}" ++shell {{xonsh|zsh|fish|bash|osquery}}
```

- Use a specific xxh configuration directory on the target machine:

```
xxh "{{host}}" ++host-xxh-home {{~/.xxh}}
```

- Use the specified configuration file on the host machine:

```
xxh "{{host}}" ++xxh-config {{~/.config/xxh/config.xxhc}}
```

- Specify a password to use for the SSH connection:

```
xxh "{{host}}" ++password "{{password}}"
```

- Install an xxh package on the target machine:

```
xxh "{{host}}" ++install-xxh-packages {{package}}
```

- Set an environment variable for the shell process on the target machine:

```
xxh "{{host}}" ++env {{name}}={{value}}
```

xz

Compress or decompress .xz and .lzma files.

More information: https://tukaani.org/xz/format.html

- Compress a file to the xz file format:

  ```
  xz {{file}}
  ```

- Decompress a xz file:

  ```
  xz -d {{file.xz}}
  ```

- Compress a file to the lzma file format:

  ```
  xz --format=lzma {{file}}
  ```

- Decompress an lzma file:

  ```
  xz -d --format=lzma {{file.lzma}}
  ```

- Decompress a file and write to stdout:

  ```
  xz -dc {{file.xz}}
  ```

- Compress a file, but don't delete the original:

  ```
  xz -k {{file}}
  ```

- Compress a file using the fastest compression:

  ```
  xz -0 {{file}}
  ```

- Compress a file using the best compression:

  ```
  xz -9 {{file}}
  ```

yacas

Y et Another Computer Algebra S ystem.

More information: http://www.yacas.org.

- S tart an interactive `yacas` session:

  ```
  yacas
  ```

- While in a `yacas` session, execute a statement:

  ```
  {{I nteg rate( x) C os( x) }};
  ```

- While in a `yacas` session, display an example:

  ```
  {{E xam ple( ) }};
  ```

- Q uit from a `yacas` session:

  ```
  {{q u it}}
  ```

- E xecute one or more `yacas` scripts (without terminal or prompts), then exit:

  ```
  yacas - p - c {{path/to/script1 }} {{path/to/script2 }}
  ```

- E xecute and print the result of one statement, then exit:

  ```
  echo " {{E cho(  D eriv ( x) C os( 1 /x)  ) ; }}"  |  yacas - p - c /dev /st
  ```

yapf

Python style guide checker.

More information: https://github.com/google/yapf.

- Display a diff of the changes that would be made, without making them (dry-run):

  ```
  yapf --diff {{path/to/file}}
  ```

- Format the file in-place and display a diff of the changes:

  ```
  yapf --diff --in-place {{path/to/file}}
  ```

- Recursively format all Python files in a directory, concurrently:

  ```
  yapf --recursive --in-place --style {{pep8}} --parallel {{path/to/directory}}
  ```

yarn-why

Identifies why a Y arn package has been installed.

More information: https://www.npmj s.com/package/yarn-why

- S how why a Y arn package is installed:

```
yarn- w hy {{pack ag e_nam e}}
```

yarn

JavaScript and Node.js package manager alternative.

More information: https://yarnpkg.com.

- Install a module globally:

  ```
  yarn global add {{module_name}}
  ```

- Install all dependencies referenced in the `package.json` file (the `install` is optional):

  ```
  yarn install
  ```

- Install a module and save it as a dependency to the `package.json` file (add `--dev` to save as a dev dependency):

  ```
  yarn add {{module_name}}@{{version}}
  ```

- Uninstall a module and remove it from the `package.json` file:

  ```
  yarn remove {{module_name}}
  ```

- Interactively create a `package.json` file:

  ```
  yarn init
  ```

- Identify whether a module is a dependency and list other modules that depend upon it:

  ```
  yarn why {{module_name}}
  ```

yes

Output something repeatedly.

This command is commonly used to answer yes to every prompt by install commands (such as apt-get).

More information: https://www.gnu.org/software/coreutils/yes.

- Repeatedly output "message":

  ```
  yes {{message}}
  ```

- Repeatedly output "y":

  ```
  yes
  ```

- Accept everything prompted by the `apt-get` command:

  ```
  yes | sudo apt-get install {{program}}
  ```

yesod

Helper tool for Yesod, a Haskell-based web framework.

All Yesod commands are invoked through the `stack` project manager.

More information: https://github.com/yesodweb/yesod.

- Create a new scaffolded site, with sqlite as backend, in the `my-project` directory:

  ```
  stack new {{my-project}} {{yesod-sqlite}}
  ```

- Install the Yesod CLI tool within a Yesod scaffolded site:

  ```
  stack build yesod-bin cabal-install --install-ghc
  ```

- Start development server:

  ```
  stack exec -- yesod devel
  ```

- Touch files with altered Template Haskell dependencies:

  ```
  stack exec -- yesod touch
  ```

- Deploy application using Keter (Yesod's deployment manager):

  ```
  stack exec -- yesod keter
  ```

youtube-dl

Download videos from YouTube and other websites.

More information: http://rg3.github.io/youtube-dl/

- Download a video or playlist:

```
youtube-dl 'https://www.youtube.com/watch?v=oHg5SJYR
```

- List all formats that a video or playlist is available in:

```
youtube-dl --list-formats 'https://www.youtube.com/watc
v=MwaO_nE9H7A'
```

- Download a video or playlist at a specific quality:

```
youtube-dl --format "best[height<=480]" 'https://
www.youtube.com/watch?v=oHg5SJYRHA0'
```

- Download the audio from a video and convert it to an MP3:

```
youtube-dl -x --audio-format mp3 'url'
```

- Download the best quality audio and video and merge them:

```
youtube-dl -f bestvideo+bestaudio 'url'
```

- Download video(s) as MP4 files with custom filenames:

```
youtube-dl --format mp4 -o "%(title)s by %(uploader)
on %(upload_date)s in %(playlist)s.%(ext)s" 'url'
```

- Download a particular language's subtitles along with the video:

```
youtube-dl --sub-lang en --write-sub 'https://
www.youtube.com/watch?v=MwaO_nE9H7A'
```

- Download a playlist and extract mp3 from it:

```
youtube-dl -f "bestaudio" --continue --no-overwrites --
ignore-errors --extract-audio --audio-format mp3 -o "%
(title)s.%(ext)s" url_to_playlist
```

youtube-viewer

Command-line application for searching and playing videos from YouTube.

More information: https://github.com/triz-en/youtube-viewer

- Search for a video:

  ```
  youtube-viewer {{search_term}}
  ```

- Log in to your YouTube account:

  ```
  youtube-viewer --login
  ```

- Watch a video with a specific URL in VLC:

  ```
  youtube-viewer --player={{vlc}} {{https://youtube.com/watch?v=dQw4w9WgXcQ}}
  ```

- Display a search prompt and play the selected video in 720p:

  ```
  youtube-viewer -{{720}}
  ```

yq

A lightweight and portable command-line YAML processor.

More information: https://mikefarah.gitbook.io/yq/

- Output a YAML file, in pretty-print format (v4+):

```
yq eval {{path/to/file.yaml}}
```

- Output a YAML file, in pretty-print format (v3):

```
yq read {{path/to/file.yaml}} --colors
```

- Output the first element in a YAML file that contains only an array (v4+):

```
yq eval '.[0]' {{path/to/file.yaml}}
```

- Output the first element in a YAML file that contains only an array (v3):

```
yq read {{path/to/file.yaml}} '[0]'
```

- Set (or overwrite) a key to a value in a file (v4+):

```
yq eval '.{{key}} = "{{value}}"' --inplace {{path/to/file.yaml}}
```

- Set (or overwrite) a key to a value in a file (v3):

```
yq write --inplace {{path/to/file.yaml}} '{{key}}' '{{value}}'
```

- Merge two files and print to stdout (v4+):

```
yq eval-all 'select(filename == "{{path/to/file1.yaml}}") * select(filename == "{{path/to/file2.yaml}}")' {{path/to/file1.yaml}} {{path/to/file2.yaml}}
```

- Merge two files and print to stdout (v3):

```
yq merge {{path/to/file1.yaml}} {{path/to/file2.yaml}} --colors
```

z

Tracks the most used (by frecency) directories and enables quickly navigating to them using string patterns or regular expressions.

More information: https://github.com/rupa/z.

- Go to a directory that contains "foo" in the name:

  `z {{foo}}`

- Go to a directory that contains "foo" and then "bar":

  `z {{foo}} {{bar}}`

- Go to the highest-ranked directory matching "foo":

  `z -r {{foo}}`

- Go to the most recently accessed directory matching "foo":

  `z -t {{foo}}`

- List all directories in `z`'s database matching "foo":

  `z -l {{foo}}`

- Remove the current directory from `z`'s database:

  `z -x .`

- Restrict matches to subdirectories of the current directory:

  `z -c {{foo}}`

zbarimg

Scan and decode bar codes from image file(s).

More information: http://zbar.sourceforge.net

- Process an image file:

```
zbarimg {{image_file}}
```

z cat

Print data from gzip compressed files.

- Print the uncompressed contents of a gzipped file to the standard output:

  ```
  zcat {{file.txt.gz}}
  ```

- Print compression details of a gzipped file to the standard output:

  ```
  zcat -l {{file.txt.gz}}
  ```

zdb

ZFS debugger.

- Show detailed configuration of all mounted ZFS zpools:

  ```
  zdb
  ```

- Show detailed configuration for a specific ZFS pool:

  ```
  zdb -C {{poolname}}
  ```

- Show statistics about number, size and deduplication of blocks:

  ```
  zdb -b {{poolname}}
  ```

zeek

Passive network traffic analyser.

Any output and log files will be saved to the current working directory.

More information: https://docs.zeek.org/en/lts/quickstart.html#zeek-as-a-command-line-utility.

- Analyze live traffic from a network interface:

```
sudo zeek --iface {{interface}}
```

- Analyze live traffic from a network interface and load custom scripts:

```
sudo zeek --iface {{interface}} {{script1}} {{script2}}
```

- Analyze live traffic from a network interface, without loading any scripts:

```
sudo zeek --bare-mode --iface {{interface}}
```

- Analyze live traffic from a network interface, applying a `tcpdump` filter:

```
sudo zeek --filter {{path/to/filter}} --iface {{interface}}
```

- Analyze live traffic from a network interface using a watchdog timer:

```
sudo zeek --watchdog --iface {{interface}}
```

- Analyze traffic from a `pcap` file:

```
zeek --readfile {{path/to/file.trace}}
```

zfs

Manage ZFS filesystems.

- List all available zfs filesystems:

```
zfs list
```

- Create a new ZFS filesystem:

```
zfs create {{pool_name/filesystem_name}}
```

- Delete a ZFS filesystem:

```
zfs destroy {{pool_name/filesystem_name}}
```

- Create a Snapshot of a ZFS filesystem:

```
zfs snapshot {{pool_name/filesystem_name}}@{{snapshot_name}}
```

- Enable compression on a filesystem:

```
zfs set compression=on {{pool_name/filesystem_name}}
```

- Change mountpoint for a filesystem:

```
zfs set mountpoint={{/my/mount/path}} {{pool_name/
filesystem_name}}
```

zip

Package and compress (archive) files into zip file.

- Package and compress files and directories [r]ecursively:

```
zip -r {{compressed.zip}} {{path/to/file}} {{path/to/directory1}} {{path/to/directory2}}
```

- Exclude unwanted files from being added to the compressed archive:

```
zip -r {{compressed.zip}} {{path/to/directory}} -x {{path/to/exclude}}
```

- Archive a directory and its contents with the highest level [9] of compression:

```
zip -r -{{9}} {{compressed.zip}} {{path/to/directory}}
```

- Create an encrypted archive (user will be prompted for a password):

```
zip -e -r {{compressed.zip}} {{path/to/directory}}
```

- Add files to an existing zip file:

```
zip {{compressed.zip}} {{path/to/file}}
```

- Delete files from an existing zip file:

```
zip -d {{compressed.zip}} "{{foo/*.tmp}}"
```

- Archive a directory and its contents to a multi-part [s]plit zip file (e.g. 3GB parts):

```
zip -r -s {{3g}} {{compressed.zip}} {{path/to/directory}}
```

- List files within a specified archive (without extracting them):

```
zip -sf {{compressed.zip}}
```

zipalign

Zip archive alignment tool.

Part of the Android SDK build tools.

More information: https://developer.android.com/studio/command-line/zipalign

- Align the data of a ZIP file on 4-byte boundaries:

  ```
  zipalign {{4}} {{path/to/input.zip}} {{path/to/output.zip}}
  ```

- Check that a ZIP file is correctly aligned on 4-byte boundaries and display the results in a verbose manner:

  ```
  zipalign -v -c {{4}} {{path/to/input.zip}}
  ```

z less

View compressed files.

- Page through a compressed archive with `less`:

```
zless {{file.txt.gz}}
```

zmv

Move or rename files matching a specified extended glob pattern.

See also `zcp` and `zln`.

More information: http://zsh.sourceforge.net/Doc/Release/User-Contributions.html.

- Move files using a regular expression-like pattern:

```
zmv '{{(*).log}}' '{{$1.txt}}'
```

- Preview the result of a move, without making any actual changes:

```
zmv -n '{{(*).log}}' '{{$1.txt}}'
```

- Interactively move files, with a prompt before every change:

```
zmv -i '{{(*).log}}' '{{$1.txt}}'
```

- Verbosely print each action as it's being executed:

```
zmv -v '{{(*).log}}' '{{$1.txt}}'
```

z ola

A static site generator in a single binary with everything built-in.

More information: https://www.getz ola.org/documentation/getting-started/cli-usage/

- Create the directory structure used by Z ola at the given directory:

  `zola init {{m y_site}}`

- B uild the whole site in the `pu b li` directory after deleting it:

  `zola b u ild`

- B uild the whole site into a different directory:

  `zola b u ild - - ou tpu t- dir {{path/to/ou tpu t_directory/}}`

- B uild and serve the site using a local server (default is `l 2 7 . 0 . 0 . 1`):`: 1 1 1 1`

  `zola serv e`

- B uild all pages j ust like the build command would, but without writing any of the results to disk:

  `zola check`

z opfl ipng

P N G image compression utility.

More information: https://github.com/google/z opfl. i

- O ptimiz e a P N G image:

```
zopflipng {{inpu t. png }} {{ou tpu t. png }}
```

- O ptimiz e several P N G images and save with given prefix:

```
zopflipng - - prefix= {{prefix}} {{im ag e1 . png }} {{im ag e2 . png }}
{{im ag e3 . png }}
```

zoxide

Keep track of the most frequently used directories.

Uses a ranking algorithm to navigate to the best match.

More information: https://github.com/ajeetdsouza/zoxide

- Go to the highest-ranked directory that contains "foo" in the name:

  `zoxide query {{foo}}`

- Go to the highest-ranked directory that contains "foo" and then "bar":

  `zoxide query {{foo}} {{bar}}`

- Start an interactive directory search (requires `fzf`):

  `zoxide query --interactive`

- Add a directory or increment its rank:

  `zoxide add {{path/to/directory}}`

- Remove a directory from `zoxide`'s database:

  `zoxide remove {{path/to/directory}}`

- Generate shell configuration for command aliases (`z`, `za`, `zi`, `zq`, `zr`):

  `zoxide init {{bash|fish|zsh}}`

zpool

Manage ZFS pools.

- Show the configuration and status of all ZFS zpools:

  ```
  zpool status
  ```

- Check a ZFS pool for errors (verifies the checksum of EVERY block). Very CPU and disk intensive:

  ```
  zpool scrub {{pool_name}}
  ```

- List zpools available for import:

  ```
  zpool import
  ```

- Import a zpool:

  ```
  zpool import {{pool_name}}
  ```

- Export a zpool (unmount all filesystems):

  ```
  zpool export {{pool_name}}
  ```

- Show the history of all pool operations:

  ```
  zpool history {{pool_name}}
  ```

- Create a mirrored pool:

  ```
  zpool create {{pool_name}} mirror {{disk1}} {{disk2}} mirror
  {{disk3}} {{disk4}}
  ```

- Add a cache (L2ARC) device to a zpool:

  ```
  zpool add {{pool_name}} cache {{cache_disk}}
  ```

zsh

Z S H ell, a B ash-compatible command-line interpreter.

S ee also `history expand` for history expansion.

More information: https://www.zsh.org

- S tart an interactive shell session:

  ```
  zsh
  ```

- E xecute a command and then exit:

  ```
  zsh -c "{{command}}"
  ```

- E xecute a script:

  ```
  zsh {{path/to/script.zsh}}
  ```

- E xecute a script, printing each command before executing it:

  ```
  zsh --xtrace {{path/to/script.zsh}}
  ```

- S tart an interactive shell session in verbose mode, printing each command before executing it:

  ```
  zsh --verbose
  ```

z std

Compress or decompress files with Z standard compression.

More information: https://github.com/facebook/z std

- Compress a file into a new file with the `. zst` suffix:

  ```
  zstd {{file}}
  ```

- Decompress a file:

  ```
  zstd - d {{file}}. zst
  ```

- Decompress to stdout:

  ```
  zstd - dc {{file}}. zst
  ```

- Compress a file specifying the compression level, where 1 = fastest, 1 9 = slowest and 3 = default:

  ```
  zstd - {{lev el}} {{file}}
  ```

- U nlock higher compression levels (up to 2 2) using more memory (both for compression and decompression):

  ```
  zstd - - u ltra - {{lev el}} {{file}}
  ```

zsteg

Steganography detection tool for PNG and BMP file formats.

It detects LSB steganography, ZLIB-compressed data, OpenStego, Camouflage and LSB with the Eratosthenes set.

More information: https://github.com/zed-0xff/zsteg

- Detect embedded data in a PNG image:

  ```
  zsteg {{path/to/image.png}}
  ```

- Detect embedded data in a BMP image, using all known methods:

  ```
  zsteg --all {{path/to/image.bmp}}
  ```

- Detect embedded data in a PNG image, iterating pixels vertically and using MSB first:

  ```
  zsteg --msb --order yx {{path/to/image.png}}
  ```

- Detect embedded data in a BMP image, specifying the bits to consider:

  ```
  zsteg --bits {{1,2,3|1-3}} {{path/to/image.bmp}}
  ```

- Detect embedded data in a PNG image, extracting only prime pixels and inverting bits:

  ```
  zsteg --prime --invert {{path/to/image.png}}
  ```

- Detect embedded data in a BMP image, specifying the minimum length of the strings to be found and the find mode:

  ```
  zsteg --min-str-len {{10}} --strings {{first|all|longest|none}} {{path/to/image.bmp}}
  ```

L inu x

a2 disconf

Disable an Apache configuration file on Debian-based O S es.

More information: https://manpages.debian.org/latest/apache2 /a2 disconf.8 .en.html

- Disable a configuration file:

  ```
  su do a2 disconf {{config u ration_file}}
  ```

- Don' t show informative messages:

  ```
  su do a2 disconf - - q u iet {{config u ration_file}}
  ```

a2 dismod

Disable an Apache module on Debian-based O S es.

More information: https://manpages.debian.org/latest/apache2 /a2 dismod.8 .en.html

- Disable a module:

  ```
  su do a2 dism od {{m odu le}}
  ```

- Don' t show informative messages:

  ```
  su do a2 dism od - - q u iet {{m odu le}}
  ```

a2 dissite

Disable an Apache virtual host on Debian-based O S es.

More information: https://manpages.debian.org/latest/apache2 /a2 dissite.8 .en.html

- Disable a virtual host:

```
su do a2 dissite {{v irtu al_host}}
```

- Don' t show informative messages:

```
su do a2 dissite - - q u iet {{v irtu al_host}}
```

a2 enconf

E nable an Apache configuration file on Debian-based O S es.

More information: https://manpages.debian.org/latest/apache2 /a2 enconf.8 .en.html

- E nable a configuration file:

  ```
  su do a2 enconf {{config u ration_file}}
  ```

- Don' t show informative messages:

  ```
  su do a2 enconf - - q u iet {{config u ration_file}}
  ```

a2enmod

Enable an Apache module on Debian-based OSes.

More information: https://manpages.debian.org/latest/apache2-/a2enmod.8-.en.html

- Enable a module:

  ```
  sudo a2enmod {{module}}
  ```

- Don't show informative messages:

  ```
  sudo a2enmod --quiet {{module}}
  ```

a2ensite

Enable an Apache virtual host on Debian-based OSes.

More information: https://manpages.debian.org/latest/apache2-/a2ensite.8-.en.html

- Enable a virtual host:

  ```
  sudo a2ensite {{virtual_host}}
  ```

- Don't show informative messages:

  ```
  sudo a2ensite --quiet {{virtual_host}}
  ```

a2 q uery

R etrieve runtime configuration from Apache on Debian-based O S es.

More information: https://manpages.debian.org/buster/apache2 /a2 q uery.1 .en.html

- List enabled Apache modules:

```
su do a2 q u ery - m
```

- Check if a specific module is installed:

```
su do a2 q u ery - m {{m odu le_nam e}}
```

- List enabled virtual hosts:

```
su do a2 q u ery - s
```

- Display the currently enabled Multi P rocessing Module:

```
su do a2 q u ery - M
```

- Display the Apache version:

```
su do a2 q u ery - v
```

ac

Print statistics on how long users have been connected.

More information: https://www.gnu.org/software/acct/manual/accounting.html#_ac

- Print how long the current user has been connected in hours:

```
ac
```

- Print how long users have been connected in hours:

```
ac --individual-totals
```

- Print how long a particular user has been connected in hours:

```
ac --individual-totals {{username}}
```

- Print how long a particular user has been connected in hours per day (with total):

```
ac --daily-totals --individual-totals {{username}}
```

- Also display additional details:

```
ac --compatibility
```

acpi

Shows battery status or thermal information.

More information: https://sourceforge.net/projects/acpiclient/files/acpiclient/

- Show battery information:

  `acpi`

- Show thermal information:

  `acpi -t`

- Show cooling device information:

  `acpi -c`

- Show thermal information in Fahrenheit:

  `acpi -tf`

- Show all information:

  `acpi -V`

- Extract information from `/proc` instead of `/sys`:

  `acpi -p`

add-apt-repository

Manages apt repository definitions.

More information: https://manned.org/apt-add-repository.

- Add a new apt repository:

  ```
  add-apt-repository {{repository_spec}}
  ```

- Remove an apt repository:

  ```
  add-apt-repository --remove {{repository_spec}}
  ```

- Update the package cache after adding a repository:

  ```
  add-apt-repository --update {{repository_spec}}
  ```

- Enable source packages:

  ```
  add-apt-repository --enable-source {{repository_spec}}
  ```

addpart

Tells the Linux kernel about the existence of the specified partition.

The command is a simple wrapper around the `addpartition`.

More information: https://manned.org/addpart.

- Tell the kernel about the existence of the specified partition:

```
addpart {{device}} {{partition}} {{start}} {{length}}
```

addr2 line

Convert addresses of a binary into file names and line numbers.

More information: https://manned.org/addr2 line

- Display the filename and line number of the source code from an instruction address of an executable:

```
addr2 line - - exe= {{path/to/execu tab le}} {{address}}
```

- Display the function name, filename and line number:

```
addr2 line - - exe= {{path/to/execu tab le}} - - fu nctions
{{address}}
```

- Demangle the function name for C+ + code:

```
addr2 line - - exe= {{path/to/execu tab le}} - - fu nctions - - dem ang le
{{address}}
```

adduser

U ser addition utility.

More information: https://manpages.debian.org/latest/adduser/adduser.html.

- Create a new user with a default home directory and prompt the user to set a password:

  ```
  addu ser {{u sernam e}}
  ```

- Create a new user without a home directory:

  ```
  addu ser - - no- create- hom e {{u sernam e}}
  ```

- Create a new user with a home directory at the specified path:

  ```
  addu ser - - hom e {{path/to/hom e}} {{u sernam e}}
  ```

- Create a new user with the specified shell set as the login shell:

  ```
  addu ser - - shell {{path/to/shell}} {{u sernam e}}
  ```

- Create a new user belonging to the specified group:

  ```
  addu ser - - ing rou p {{g rou p}} {{u sernam e}}
  ```

- Add an existing user to the specified group:

  ```
  addu ser {{u sernam e}} {{g rou p}}
  ```

alpine

An email client and U senet newsgroup program with a pico/nano-inspired interface.

S upports most modern email services through IMAP .

- O pen alpine normally:

  ```
  alpine
  ```

- O pen alpine directly to the message composition screen to send an email to a given email address:

  ```
  alpine {{em ail@ exam ple. net}}
  ```

- Q uit alpine:

  ```
  ' q '  then ' y'
  ```

amixer

Mixer for ALSA soundcard driver.

More information: https://manned.org/amixer.

- Turn up the master volume by 10%:

```
amixer -D pulse sset Master {{10%+}}
```

- Turn down the master volume by 10%:

```
amixer -D pulse sset Master {{10%-}}
```

anbox

R un Android applications on any G N U /Linux operating system.

More information: https://manned.org/anbox.

- Launch Anbox into the app manager:

```
anb ox lau nch - - pack ag e= {{org . anb ox. appm g r}} - -
com ponent= {{org . anb ox. appm g r. A ppV iew A ctiv ity}}
```

apache2ctl

The CLI tool to administrate HTTP web server Apache.

This command comes with Debian based OSes, for RHEL based ones see `httpd`

More information: https://manpages.debian.org/latest/apache2/apache2ctl.8.en.html

- Start the Apache daemon. Throw a message if it is already running:

  ```
  sudo apache2ctl start
  ```

- Stop the Apache daemon:

  ```
  sudo apache2ctl stop
  ```

- Restart the Apache daemon:

  ```
  sudo apache2ctl restart
  ```

- Test syntax of the configuration file:

  ```
  sudo apache2ctl -t
  ```

- List loaded modules:

  ```
  sudo apache2ctl -M
  ```

apk

Alpine Linux package management tool.

More information: https://wiki.alpinelinux.org/wiki/Alpine_Linux_package_management

- Update repository indexes from all remote repositories:

  ```
  apk update
  ```

- Install a new package:

  ```
  apk add {{package}}
  ```

- Remove a package:

  ```
  apk del {{package}}
  ```

- Repair package or upgrade it without modifying main dependencies:

  ```
  apk fix {{package}}
  ```

- Search package via keyword:

  ```
  apk search {{keyword}}
  ```

- Get info about a specific package:

  ```
  apk info {{package}}
  ```

aplay

Command-line sound player for ALSA soundcard driver.

More information: https://manned.org/aplay.

- Play a specific file (sampling rate, bit depth, etc. will be automatically determined for the file format):

```
aplay {{path/to/file}}
```

- Play the first 10 seconds of a specific file at 2500 Hz:

```
aplay --duration={{10}} --rate={{2500}} {{path/to/file}}
```

- Play the raw file as a 22050 Hz, mono, 8-bit, Mu-law:

```
aplay --channels={{1}} --file-type {{raw}} --rate={{22050}}
--format={{mu_law}} {{path/to/file}}
```

apport-bug

File a bug report on Ubuntu.

More information: https://wiki.ubuntu.com/Apport.

- Report a bug about the whole system:

  ```
  apport-bug
  ```

- Report a bug about a specific package:

  ```
  apport-bug {{package}}
  ```

- Report a bug about a specific executable:

  ```
  apport-bug {{path/to/executable}}
  ```

- Report a bug about a specific process:

  ```
  apport-bug {{PID}}
  ```

apt-add-repository

Manages apt repository definitions.

More information: https://manpages.debian.org/latest/software-properties-common/apt-add-repository.1.html

- Add a new apt repository:

  ```
  apt-add-repository {{repository_spec}}
  ```

- Remove an apt repository:

  ```
  apt-add-repository --remove {{repository_spec}}
  ```

- Update the package cache after adding a repository:

  ```
  apt-add-repository --update {{repository_spec}}
  ```

- Enable source packages:

  ```
  apt-add-repository --enable-source {{repository_spec}}
  ```

apt-cache

Debian and Ubuntu package query tool.

More information: https://manpages.debian.org/latest/apt/apt-cache.8.html

- Search for a package in your current sources:

  ```
  apt-cache search {{query}}
  ```

- Show information about a package:

  ```
  apt-cache show {{package}}
  ```

- Show whether a package is installed and up to date:

  ```
  apt-cache policy {{package}}
  ```

- Show dependencies for a package:

  ```
  apt-cache depends {{package}}
  ```

- Show packages that depend on a particular package:

  ```
  apt-cache rdepends {{package}}
  ```

apt-file

Search for files in apt packages, including ones not yet installed.

More information: https://manpages.debian.org/latest/apt-file/apt-file.1.html

- Update the metadata database:

  ```
  sudo apt update
  ```

- Search for packages that contain the specified file or path:

  ```
  apt-file {{search|find}} {{path/to/filename}}
  ```

- List the contents of a specific package:

  ```
  apt-file {{show|list}} {{package_name}}
  ```

- Search for packages that match the regular expression given in `pattern`:

  ```
  apt-file {{search|find}} --regexp {{regular_expression}}
  ```

apt-get

Debian and Ubuntu package management utility.

Search for packages using `apt-cache`

More information: https://manpages.debian.org/latest/apt/apt-get.8.html

- Update the list of available packages and versions (it's recommended to run this before other `apt-get` commands):

  ```
  apt-get update
  ```

- Install a package, or update it to the latest available version:

  ```
  apt-get install {{package}}
  ```

- Remove a package:

  ```
  apt-get remove {{package}}
  ```

- Remove a package and its configuration files:

  ```
  apt-get purge {{package}}
  ```

- Upgrade all installed packages to their newest available versions:

  ```
  apt-get upgrade
  ```

- Clean the local repository - removing package files (`.deb`) from interrupted downloads that can no longer be downloaded:

  ```
  apt-get autoclean
  ```

- Remove all packages that are no longer needed:

  ```
  apt-get autoremove
  ```

- Upgrade installed packages (like `upgrade`), but remove obsolete packages and install additional packages to meet new dependencies:

  ```
  apt-get dist-upgrade
  ```

apt-key

Key management utility for the APT Package Manager on Debian and Ubuntu.

Note: `apt-key` is now deprecated (except for the use of `apt-key del` in maintainer scripts).

More information: https://manpages.debian.org/latest/apt/apt-key.8.html

- List trusted keys:

  ```
  apt-key list
  ```

- Add a key to the trusted keystore:

  ```
  apt-key add {{public_key_file.asc}}
  ```

- Delete a key from the trusted keystore:

  ```
  apt-key del {{key_id}}
  ```

- Add a remote key to the trusted keystore:

  ```
  wget -qO - {{https://host.tld/filename.key}} | apt-key add -
  ```

- Add a key from keyserver with only key id:

  ```
  apt-key adv --keyserver {{pgp.mit.edu}} --recv {{KEYID}}
  ```

apt-mark

Utility to change the status of installed packages.

More information: https://manpages.debian.org/latest/apt/apt-mark.8.html

- Mark a package as automatically installed:

```
sudo apt-mark auto {{package_name}}
```

- Hold a package at its current version and prevent updates to it:

```
sudo apt-mark hold {{package_name}}
```

- Allow a package to be updated again:

```
sudo apt-mark unhold {{package_name}}
```

- Show manually installed packages:

```
apt-mark showmanual
```

- Show held packages that aren't being updated:

```
apt-mark showhold
```

apt

Package management utility for Debian based distributions.

Recommended replacement for apt-get when used interactively in Ubuntu versions 16.04 and later.

More information: https://manpages.debian.org/latest/apt/apt.8.html

- Update the list of available packages and versions (it's recommended to run this before other `apt` commands):

  ```
  sudo apt update
  ```

- Search for a given package:

  ```
  apt search {{package}}
  ```

- Show information for a package:

  ```
  apt show {{package}}
  ```

- Install a package, or update it to the latest available version:

  ```
  sudo apt install {{package}}
  ```

- Remove a package (using `purge` instead also removes its configuration files):

  ```
  sudo apt remove {{package}}
  ```

- Upgrade all installed packages to their newest available versions:

  ```
  sudo apt upgrade
  ```

- List all packages:

  ```
  apt list
  ```

- List installed packages:

  ```
  apt list --installed
  ```

aptitude

Debian and Ubuntu package management utility.

More information: https://manpages.debian.org/latest/aptitude/aptitude.8.html

- Synchronize list of packages and versions available. This should be run first, before running subsequent aptitude commands:

  ```
  aptitude update
  ```

- Install a new package and its dependencies:

  ```
  aptitude install {{package}}
  ```

- Search for a package:

  ```
  aptitude search {{package}}
  ```

- Search for an installed package (`?installed` is an aptitude search term):

  ```
  aptitude search '?installed({{package}})'
  ```

- Remove a package and all packages depending on it:

  ```
  aptitude remove {{package}}
  ```

- Upgrade installed packages to newest available versions:

  ```
  aptitude upgrade
  ```

- Upgrade installed packages (like `aptitude upgrade`) including removing obsolete packages and installing additional packages to meet new package dependencies:

  ```
  aptitude full-upgrade
  ```

- Put an installed package on hold to prevent it from being automatically upgraded:

  ```
  aptitude hold '?installed({{package}})'
  ```

arch-chroot

Enhanced `chroot` command to help in the Arch Linux installation process.

More information: https://man.archlinux.org/man/arch-chroot.8.

- Start an interactive shell (`bash`, by default) in a new root directory:

  ```
  arch-chroot {{path/to/new/root}}
  ```

- Specify the user (other than the current user) to run the shell as:

  ```
  arch-chroot -u {{user}} {{path/to/new/root}}
  ```

- Run a custom command (instead of the default `bash`) in the new root directory:

  ```
  arch-chroot {{path/to/new/root}} {{command}}
  {{command_arguments}}
  ```

- Specify the shell, other than the default `bash` (in this case, the `zsh` package should have been installed in the target system):

  ```
  arch-chroot {{path/to/new/root}} {{zsh}}
  ```

archey

S imple tool for stylishly displaying system information.

More information: https://lclarkmichalek.github.io/archey3 /

- S how system information:

  ```
  archey
  ```

archinstall

G uided Arch Linux installer with a twist.

More information: https://archinstall.readthedocs.io.

- S tart the interactive installer:

  ```
  archinstall
  ```

- S tart a preset installer:

  ```
  archinstall {{m inim al| u nattended}}
  ```

archlinux-j ava

A helper script that provides functionalities for J ava environments.

More information: https://github.com/michaellass/archlinux-j ava-run

- List installed J ava environments:

  ```
  archlinu x- j av a statu s
  ```

- S et the default J ava environment:

  ```
  archlinu x- j av a set {{j av a_env ironm ent}}
  ```

- U nset the default J ava environment:

  ```
  archlinu x- j av a u nset
  ```

- S et the default J ava environment automatically:

  ```
  archlinu x- j av a fix
  ```

arecord

Sound recorder for ALSA soundcard driver.

More information: https://manned.org/arecord.

- Record a snippet in "CD" quality (finish with Ctrl-C when done):

```
arecord -vv --format=cd {{path/to/file.wav}}
```

- Record a snippet in "CD" quality, with a fixed duration of 10 seconds:

```
arecord -vv --format=cd --duration={{10}} {{path/to/file.wav}}
```

- Record a snippet and save it as mp3 (finish with Ctrl-C when done):

```
arecord -vv --format=cd --file-type raw | lame -r - {{path/to/file.mp3}}
```

- List all sound cards and digital audio devices:

```
arecord --list-devices
```

- Allow interactive interface (e.g. use space-bar or enter to play or pause):

```
arecord --interactive
```

arithmetic

Quiz on simple arithmetic problems.

More information: https://manpages.debian.org/bsdgames/arithmetic.6.en.html

- Start an arithmetic quiz:

  ```
  arithmetic
  ```

- Specify one or more arithmetic [o]peration symbols to get problems on them:

  ```
  arithmetic -o {{+|-|x|/}}
  ```

- Specify a range. Addition and multiplication problems would feature numbers between 0 and range, inclusive. Subtraction and division problems would have required result and number to be operated on, between 0 and range:

  ```
  arithmetic -r {{7}}
  ```

ark

KDE archiving tool.

More information: https://docs.kde.org/stable5/en/ark/ark/

- Extract an archive into the current directory:

  ```
  ark --batch {{archive}}
  ```

- Change extraction directory:

  ```
  ark --batch --destination {{path/to/directory}} {{archive}}
  ```

- Create an archive if it does not exist and add files to it:

  ```
  ark --add-to {{archive}} {{file1}} {{file2}}
  ```

arp-scan

Send ARP packets to hosts (specified as IP addresses or hostnames) to scan the local network.

More information: https://github.com/royhills/arp-scan.

- Scan the current local network:

```
arp-scan --localnet
```

- Scan an IP network with a custom bitmask:

```
arp-scan {{192.168.1.1}}/{{24}}
```

- Scan an IP network within a custom range:

```
arp-scan {{127.0.0.0}}-{{127.0.0.31}}
```

- Scan an IP network with a custom net mask:

```
arp-scan {{10.0.0.0}}:{{255.255.255.0}}
```

as

Portable GNU assembler.

Primarily intended to assemble output from `gcc` to be used by `ld`.

More information: https://manned.org/as.

- Assemble a file, writing the output to `a.out`:

  ```
  as {{file.s}}
  ```

- Assemble the output to a given file:

  ```
  as {{file.s}} -o {{out.o}}
  ```

- Generate output faster by skipping whitespace and comment preprocessing. (Should only be used for trusted compilers):

  ```
  as -f {{file.s}}
  ```

- Include a given path to the list of directories to search for files specified in `.include` directives:

  ```
  as -I {{path/to/directory}} {{file.s}}
  ```

ascii

Show ASCII character aliases.

More information: http://www.catb.org/~esr/ascii/

- Show ASCII aliases of a character:

  ```
  ascii {{a}}
  ```

- Show ASCII aliases in short, script-friendly mode:

  ```
  ascii -t {{a}}
  ```

- Show ASCII aliases of multiple characters:

  ```
  ascii -s {{tldr}}
  ```

- Show ASCII table in decimal:

  ```
  ascii -d
  ```

- Show ASCII table in hexadecimal:

  ```
  ascii -x
  ```

- Show ASCII table in octal:

  ```
  ascii -o
  ```

- Show ASCII table in binary:

  ```
  ascii -b
  ```

- Show options summary and complete ASCII table:

  ```
  ascii
  ```

asciiart

Convert images to ASCII.

More information: https://github.com/nodanaonlyzuul/asciiart

- Read an image from a file and print in ASCII:

  ```
  asciiart {{path/to/image.jpg}}
  ```

- Read an image from a URL and print in ASCII:

  ```
  asciiart {{www.example.com/image.jpg}}
  ```

- Choose the output width (default is 100):

  ```
  asciiart -width {{50}} {{path/to/image.jpg}}
  ```

- Colorize the ASCII output:

  ```
  asciiart --color {{path/to/image.jpg}}
  ```

- Choose the output format (default format is text):

  ```
  asciiart --format {{text|html}} {{path/to/image.jpg}}
  ```

- Invert the character map:

  ```
  asciiart --invert-chars {{path/to/image.jpg}}
  ```

aspell

Interactive spell checker.

More information: http://aspell.net/.

- Spell check a single file:

  ```
  aspell check {{path/to/file}}
  ```

- List misspelled words from standard input:

  ```
  cat {{file}} | aspell list
  ```

- Show available dictionary languages:

  ```
  aspell dicts
  ```

- Run aspell with different language (takes two letter ISO 639 language code):

  ```
  aspell --lang={{cs}}
  ```

- List misspelled words from standard input and ignore words from personal word list:

  ```
  cat {{file}} | aspell --personal={{personal-word-list.pws}} {{list}}
  ```

asterisk

Telephone and exchange (phone) server.

Used for running the server itself, and managing an already running instance.

More information: https://wiki.asterisk.org/wiki/display/AST/Home

- [-r] Reconnect to a running server, and turn on logging 3 levels of [-v] erbosity:

  ```
  asterisk -r -vvv
  ```

- [-r] Reconnect to a running server, run a single command, and return:

  ```
  asterisk -r -x "{{command}}"
  ```

- Show chan_SIP clients (phones):

  ```
  asterisk -r -x "sip show peers"
  ```

- Show active calls and channels:

  ```
  asterisk -r -x "core show channels"
  ```

- Show voicemail mailboxes:

  ```
  asterisk -r -x "voicemail show users"
  ```

- Terminate a channel:

  ```
  asterisk -r -x "hangup request {{channel_ID}}"
  ```

- Reload chan_SIP configuration:

  ```
  asterisk -r -x "sip reload"
  ```

at

Executes commands at a specified time.

More information: https://man.archlinux.org/man/at.1.

- Open an `at` prompt to create a new set of scheduled commands, press `Ctrl + D` to save and exit:

  ```
  at {{hh:mm}}
  ```

- Execute the commands and email the result using a local mailing program such as sendmail:

  ```
  at {{hh:mm}} -m
  ```

- Execute a script at the given time:

  ```
  at {{hh:mm}} -f {{path/to/file}}
  ```

- Display a system notification at 11 pm on February 18th:

  ```
  echo "notify-send '{{Wake up!}}'" | at {{11pm}} {{Feb 18}}
  ```

auracle

Command-line tool used to interact with Arch Linux's User Repository, commonly referred to as the AUR.

More information: https://github.com/falconindy/auracle.

- Display AUR packages that match a regular expression:

  `auracle search '{{regular_expression}}'`

- Display package information for a space-separated list of AUR packages:

  `auracle info {{package1}} {{package2}}`

- Display the `PKGBUILD` file (build information) for a space-separated list of AUR packages:

  `auracle show {{package1}} {{package2}}`

- Display updates for installed AUR packages:

  `auracle outdated`

aurman

An Arch Linux utility to build and install packages from the Arch User Repository.

See also `pacman`.

More information: https://github.com/polygamma/aurman.

- Synchronize and update all packages:

  `aurman --sync --refresh --sysupgrade`

- Synchronize and update all packages without show changes of `PKGBUILD` files:

  `aurman --sync --refresh --sysupgrade --noedit`

- Install a new package:

  `aurman --sync {{package_name}}`

- Install a new package without show changes of `PKGBUILD` files:

  `aurman --sync --noedit {{package_name}}`

- Install a new package without prompting:

  `aurman --sync --noedit --noconfirm {{package_name}}`

- Search the package database for a keyword from the official repositories and AUR:

  `aurman --sync --search {{keyword}}`

- Remove a package and its dependencies:

  `aurman --remove --recursive --nosave {{package_name}}`

- Clear the package cache (use two `--clean` flags to clean all packages):

  `aurman --sync --clean`

authconfig

A CLI interface for configuring system authentication resources.

More information: https://access.redhat.com/documentation/en-us/red_hat_enterprise_linux/7/html/system-level_authentication_guide/authconfig-install

- Display the current configuration (or dry run):

  ```
  authconfig --test
  ```

- Configure the server to use a different password hashing algorithm:

  ```
  authconfig --update --passalgo={{algorithm}}
  ```

- Enable LDAP authentication:

  ```
  authconfig --update --enableldapauth
  ```

- Disable LDAP authentication:

  ```
  authconfig --update --disableldapauth
  ```

- Enable Network Information Service (NIS):

  ```
  authconfig --update --enablenis
  ```

- Enable Kerberos:

  ```
  authconfig --update --enablekrb5
  ```

- Enable Winbind (Active Directory) authentication:

  ```
  authconfig --update --enablewinbindauth
  ```

- Enable local authorization:

  ```
  authconfig --update --enablelocauthorize
  ```

autorandr

Automatically change screen layout.

More information: https://github.com/phillipberndt/autorandr.

- S ave the current screen layout:

  ```
  au torandr - s {{profile_nam e}}
  ```

- S how the saved profiles:

  ```
  au torandr
  ```

- Change the profile:

  ```
  au torandr - l {{profile_nam e}}
  ```

- S et the default profile:

  ```
  au torandr - d {{profile_nam e}}
  ```

avahi-browse

Displays services and hosts exposed on the local network via mDNS/DNS-SD.

Avahi is compatible with Bonjour (Zeroconf) found in Apple devices.

More information: https://www.avahi.org/.

- List all services available on the local network along with their addresses and ports while ignoring local ones:

  ```
  avahi-browse --all --resolve --ignore-local
  ```

- List all domains:

  ```
  avahi-browse --browse-domains
  ```

- Limit the search to a particular domain:

  ```
  avahi-browse --all --domain={{domain}}
  ```

balooctl

File indexing and searching framework for KDE Plasma.

More information: https://wiki.archlinux.org/index.php/Baloo

- Display help:

  ```
  baloctl
  ```

- Display the status of the indexer:

  ```
  baloctl status
  ```

- Enable/Disable the file indexer:

  ```
  baloctl {{enable|disable}}
  ```

- Clean the index database:

  ```
  baloctl purge
  ```

- Suspend the file indexer:

  ```
  baloctl suspend
  ```

- Resume the file indexer:

  ```
  baloctl resume
  ```

- Display the disk space used by Baloo:

  ```
  baloctl indexSize
  ```

- Check for any unindexed files and index them:

  ```
  baloctl check
  ```

beep

A utility to beep the PC speaker.

More information: https://github.com/spkr-beep/beep.

- Play a beep:

```
beep
```

- Play a beep that repeats:

```
beep -r {{repetitions}}
```

- Play a beep at a specified frequency (Hz) and duration (milliseconds):

```
beep -f {{frequency}} -l {{duration}}
```

- Play each new frequency and duration as a distinct beep:

```
beep -f {{frequency}} -l {{duration}} -n -f {{frequency}} -l
{{duration}}
```

- Play the C major scale:

```
beep -f 262 -n -f 294 -n -f 330 -n -f 349 -n -f 392 -
-n -f 494 -n -f 523
```

betterlockscreen

S imple, minimal lock screen.

More information: https://github.com/pavanj adhaw/betterlockscreen

- Lock the screen:

  ```
  b etterlock screen - - lock
  ```

- Change the lock screen background:

  ```
  b etterlock screen - u {{path/to/im ag e. png }}
  ```

- Lock the screen, showing some custom text:

  ```
  b etterlock screen - l pixel - t " {{cu stom lock screen text}}"
  ```

- Lock the screen, with a custom monitor off timeout in seconds:

  ```
  b etterlock screen - - off {{5 }} - l
  ```

binwalk

Firmware Analysis Tool.

More information: https://github.com/ReFirmLabs/binwalk

- Scan a binary file:

```
binwalk {{path/to/binary}}
```

- Extract files from a binary, specifying the output directory:

```
binwalk --extract --directory {{output_directory}} {{path/to/binary}}
```

- Recursively extract files from a binary limiting the recursion depth to 2:

```
binwalk --extract --matryoshka --depth {{2}} {{path/to/binary}}
```

- Extract files from a binary with the specified file signature:

```
binwalk --dd '{{png image:png}}' {{path/to/binary}}
```

- Analyze the entropy of a binary, saving the plot with the same name as the binary and `.png` extension appended:

```
binwalk --entropy --save {{path/to/binary}}
```

- Combine entropy, signature and opcodes analysis in a single command:

```
binwalk --entropy --signature --opcodes {{path/to/binary}}
```

bitwise

Multi base interactive calculator supporting dynamic base conversion and bit manipulation.

More information: https://github.com/mellowcandle/bitwise.

- R un using interactive mode:

  ```
  b itw ise
  ```

- Convert from decimal:

  ```
  b itw ise {{1 2 3 4 5 }}
  ```

- Convert from hexadecimal:

  ```
  b itw ise {{0 x5 6 3 d}}
  ```

- Convert a C-style calculation:

  ```
  b itw ise " {{0 x1 2 3  +  0 x2 0  -  3 0  / 5 0 }}"
  ```

blkdiscard

Discards device sectors on storage devices. Useful for SSDs.

More information: https://manned.org/blkdiscard.

- Discard all sectors on a device, removing all data:

```
blkdiscard /dev/{{device}}
```

- Securely discard all blocks on a device, removing all data:

```
blkdiscard --secure /dev/{{device}}
```

- Discard the first 100 MB of a device:

```
blkdiscard --length {{100MB}} /dev/{{device}}
```

blkid

Lists all recognized partitions and their Universally Unique Identifier (UUID).

More information: https://manned.org/blkid.

- List all partitions:

  ```
  sudo blkid
  ```

- List all partitions in a table, including current mountpoints:

  ```
  sudo blkid -o list
  ```

bluetoothctl

Handling bluetooth devices from the shell.

More information: https://www.npmjs.com/package/bluetoothctl

- Enter the bluetoothctl shell:

  ```
  bluetoothctl
  ```

- List devices:

  ```
  bluetoothctl --devices
  ```

- Pair a device:

  ```
  bluetoothctl --pair {{mac_address}}
  ```

- Remove a device:

  ```
  bluetoothctl --remove {{mac_address}}
  ```

- Connect a paired device:

  ```
  bluetoothctl --connect {{mac_address}}
  ```

- Disconnect a paired device:

  ```
  bluetoothctl --disconnect {{mac_address}}
  ```

bluetoothd

Daemon to manage bluetooth devices.

More information: https://manned.org/bluetoothd.

- Start the daemon:

```
bluetoothd
```

- Start the daemon, logging to stdout:

```
bluetoothd --nodetach
```

- Start the daemon with a specific configuration file (defaults to `/etc/bluetooth/main.conf`):

```
bluetoothd --config file {{path/to/file}}
```

- Start the daemon with verbose output to stderr:

```
bluetoothd --debug
```

- Start the daemon with verbose output coming from specific files in the bluetoothd or plugins source:

```
bluetoothd --debug={{path/to/file1}}:{{path/to/file2}}:{{path/to/file3}}
```

bmon

Monitor bandwidth and capture network related statistics.

More information: https://github.com/tgraf/bmon.

- Display the list of all the interfaces:

```
bmon -a
```

- Display data transfer rates in bits per second:

```
bmon -b
```

- Set policy to define which network interface(s) is/are displayed:

```
bmon -p {{interface_1,interface_2,interface_3}}
```

- Set interval (in seconds) in which rate per counter is calculated:

```
bmon -R {{2.0}}
```

boltctl

Control thunderbolt devices.

More information: https://manned.org/boltctl.

- List connected (and authoriz ed) devices:

  ```
  b oltctl
  ```

- List connected devices, including unauthoriz ed ones:

  ```
  b oltctl list
  ```

- Authoriz e a device temporarily:

  ```
  b oltctl au thorize {{dev ice_u u id}}
  ```

- Authoriz e and remember a device:

  ```
  b oltctl enroll {{dev ice_u u id}}
  ```

- R evoke a previously authoriz ed device:

  ```
  b oltctl forg et {{dev ice_u u id}}
  ```

- S how more information about a device:

  ```
  b oltctl info {{dev ice_u u id}}
  ```

bpftrace

High-level tracing language for Linux eBPF.

More information: https://github.com/iovisor/bpftrace.

- Display bpftrace version:

```
bpftrace -V
```

- List all available probes:

```
sudo bpftrace -l
```

- Run a one-liner program (e.g syscall count by program):

```
sudo bpftrace -e '{{tracepoint:raw_syscalls:sys_enter {
@[comm] = count();}}}'
```

- Run a program from a file:

```
sudo bpftrace {{path/to/file}}
```

- Trace a program by PID:

```
sudo bpftrace -e '{{tracepoint:raw_syscalls:sys_enter /pid==
123/ {@[comm] = count();}}}'
```

- Do a dry run and display the output in eBPF format:

```
sudo bpftrace -d -e '{{one_line_program}}'
```

bpytop

Display dynamic real-time information about running processes with graphs. Similar to **`gtop`** and **`htop`**.

More information: https://github.com/aristocratos/bpytop.

- Start bpytop:

  `bpytop`

- Start in minimal mode without memory and networking boxes:

  `bpytop -m`

- Show version:

  `bpytop -v`

- Toggle minimal mode:

  `m`

- Search for running programs or processes:

  `f`

- Change settings:

  `M`

brctl

Ethernet bridge administration.

More information: https://manned.org/brctl.

- Show a list with information about currently existing ethernet bridges:

  ```
  sudo brctl show
  ```

- Create a new ethernet bridge interface:

  ```
  sudo brctl add {{bridge_name}}
  ```

- Delete an existing ethernet bridge interface:

  ```
  sudo brctl del {{bridge_name}}
  ```

- Add an interface to an existing bridge:

  ```
  sudo brctl addif {{bridge_name}} {{interface_name}}
  ```

- Remove an interface from an existing bridge:

  ```
  sudo brctl delif {{bridge_name}} {{interface_name}}
  ```

btrfs device

Manage devices in a btrfs filesystem.

More information: https://btrfs.wiki.kernel.org/index.php/Manpage/btrfs-device.

- Add one or more devices to a btrfs filesystem:

```
sudo btrfs device add {{path/to/block_device1}} [{{path/to/block_device2}}] {{path/to/btrfs_filesystem}}
```

- Remove a device from a btrfs filesystem:

```
sudo btrfs device remove {{path/to/device|device_id}} [{{...}}]
```

- Display error statistics:

```
sudo btrfs device stats {{path/to/btrfs_filesystem}}
```

- Scan all disks and inform the kernel of all detected btrfs filesystems:

```
sudo btrfs device scan --all-devices
```

- Display detailed per-disk allocation statistics:

```
sudo btrfs device usage {{path/to/btrfs_filesystem}}
```

btrfs filesystem

Manage btrfs filesystems.

More information: https://btrfs.wiki.kernel.org/index.php/Manpage/btrfs-filesystem.

- Show filesystem usage (optionally run as root to show detailed information):

  ```
  btrfs filesystem usage {{path/to/btrfs_mount}}
  ```

- Show usage by individual devices:

  ```
  sudo btrfs filesystem show {{path/to/btrfs_mount}}
  ```

- Defragment a single file on a btrfs filesystem (avoid while a deduplication agent is running):

  ```
  sudo btrfs filesystem defragment -v {{path/to/file}}
  ```

- Defragment a directory recursively (does not cross subvolume boundaries):

  ```
  sudo btrfs filesystem defragment -v -r {{path/to/directory}}
  ```

- Force syncing unwritten data blocks to disk(s):

  ```
  sudo btrfs filesystem sync {{path/to/btrfs_mount}}
  ```

- Summarize disk usage for the files in a directory recursively:

  ```
  sudo btrfs filesystem du --summarize {{path/to/directory}}
  ```

btrfs scrub

Scrub btrfs filesystems to verify data integrity.

It is recommended to run a scrub once a month.

More information: https://btrfs.wiki.kernel.org/index.php/Manpage/btrfs-scrub.

- Start a scrub:

  ```
  sudo btrfs scrub start {{path/to/btrfs_mount}}
  ```

- Show the status of an ongoing or last completed scrub:

  ```
  sudo btrfs scrub status {{path/to/btrfs_mount}}
  ```

- Cancel an ongoing scrub:

  ```
  sudo btrfs scrub cancel {{path/to/btrfs_mount}}
  ```

- Resume a previously cancelled scrub:

  ```
  sudo btrfs scrub resume {{path/to/btrfs_mount}}
  ```

- Start a scrub, but wait until the scrub finishes before exiting:

  ```
  sudo btrfs scrub start -B {{path/to/btrfs_mount}}
  ```

- Start a scrub in quiet mode (does not print errors or statistics):

  ```
  sudo btrfs scrub start -q {{path/to/btrfs_mount}}
  ```

btrfs subvolume

Manage btrfs subvolumes and snapshots.

More information: https://btrfs.wiki.kernel.org/index.php/Manpage/btrfs-subvolume.

- Create a new empty subvolume:

```
su do b trfs su b v olu m e create {{path/to/new _su b v olu m e}}
```

- List all subvolumes and snapshots in the specified filesystem:

```
su do b trfs su b v olu m e list {{path/to/b trfs_filesystem }}
```

- Delete a subvolume:

```
su do b trfs su b v olu m e delete {{path/to/su b v olu m e}}
```

- Create a read-only snapshot of an existing subvolume:

```
su do b trfs su b v olu m e snapshot - r {{path/to/sou rce_su b v olu m e}}
{{path/to/targ et}}
```

- Create a read-write snapshot of an existing subvolume:

```
su do b trfs su b v olu m e snapshot {{path/to/sou rce_su b v olu m e}}
{{path/to/targ et}}
```

- S how detailed information about a subvolume:

```
su do b trfs su b v olu m e show {{path/to/su b v olu m e}}
```

btrfs

A filesystem based on the copy-on-write (COW) principle for Linux.

More information: https://btrfs.wiki.kernel.org/index.php/Manpage/btrfs.

- Create subvolume:

  ```
  sudo btrfs subvolume create {{path/to/subvolume}}
  ```

- List subvolumes:

  ```
  sudo btrfs subvolume list {{path/to/mount_point}}
  ```

- Show space usage information:

  ```
  sudo btrfs filesystem df {{path/to/mount_point}}
  ```

- Enable quota:

  ```
  sudo btrfs quota enable {{path/to/subvolume}}
  ```

- Show quota:

  ```
  sudo btrfs qgroup show {{path/to/subvolume}}
  ```

cal

Prints calendar information, with the current day highlighted.

More information: https://manned.org/cal.

- Display a calendar for the current month:

  ```
  cal
  ```

- Display previous, current and next month:

  ```
  cal -3
  ```

- Use monday as the first day of the week:

  ```
  cal --monday
  ```

- Display a calendar for a specific year (4 digits):

  ```
  cal {{year}}
  ```

- Display a calendar for a specific month and year:

  ```
  cal {{month}} {{year}}
  ```

calc

An interactive arbitrary-precision calculator on the terminal.

More information: https://github.com/lcn2/calc

- Start calc in interactive mode:

  ```
  calc
  ```

- Perform a calculation in non-interactive mode:

  ```
  calc -p '{{85 * (36 / 4)}}'
  ```

calcurse

A text-based calendar and scheduling application for the command-line.

More information: https://calcurse.org.

- Start calcurse on interactive mode:

  ```
  calcurse
  ```

- Print the appointments and events for the current day and exit:

  ```
  calcurse --appointment
  ```

- Remove all local calcurse items and import remote objects:

  ```
  calcurse-caldav --init=keep-remote
  ```

- Remove all remote objects and push local calcurse items:

  ```
  calcurse-caldav --init=keep-local
  ```

- Copy local objects to the CalDAV server and vice versa:

  ```
  calcurse-caldav --init=two-way
  ```

ceph

A unified storage system.

More information: https://ceph.io.

- Check cluster health status:

  ```
  ceph statu s
  ```

- Check cluster usage stats:

  ```
  ceph df
  ```

- G et the statistics for the placement groups in a cluster:

  ```
  ceph pg  du m p - - form at {{plain}}
  ```

- Create a storage pool:

  ```
  ceph osd pool create {{pool_nam e}} {{pag e_nu m b er}}
  ```

- Delete a storage pool:

  ```
  ceph osd pool delete {{pool_nam e}}
  ```

- R ename a storage pool:

  ```
  ceph osd pool renam e {{cu rrent_nam e}} {{new _nam e}}
  ```

- S elf-repair pool storage:

  ```
  ceph pg  repair {{pool_nam e}}
  ```

certbot

The Let's Encrypt Agent for automatically obtaining and renewing TLS certificates.

Successor to `letsencrypt`

More information: https://certbot.eff.org/docs/using.html.

- Obtain a new certificate via webroot authorization, but do not install it automatically:

  ```
  sudo certbot certonly --webroot --webroot-path {{path/to/webroot}} --domain {{subdomain.example.com}}
  ```

- Obtain a new certificate via nginx authorization, installing the new certificate automatically:

  ```
  sudo certbot --nginx --domain {{subdomain.example.com}}
  ```

- Obtain a new certificate via apache authorization, installing the new certificate automatically:

  ```
  sudo certbot --apache --domain {{subdomain.example.com}}
  ```

- Renew all Let's Encrypt certificates that expire in 30 days or less (don't forget to restart any servers that use them afterwards):

  ```
  sudo certbot renew
  ```

- Simulate the obtaining of a new certificate, but don't actually save any new certificates to disk:

  ```
  sudo certbot --webroot --webroot-path {{path/to/webroot}} --domain {{subdomain.example.com}} --dry-run
  ```

- Obtain an untrusted test certificate instead:

  ```
  sudo certbot --webroot --webroot-path {{path/to/webroot}} --domain {{subdomain.example.com}} --test-cert
  ```

cewl

URL spidering tool for making a cracking wordlist from web content.

More information: https://digi.ninja/projects/cewl.php

- Create a wordlist file from the given URL up to 2 links depth:

```
cewl --depth {{2}} --write {{path/to/wordlist.txt}} {{url}}
```

- Output an alphanumeric wordlist from the given URL with words of minimum 5 characters:

```
cewl --with-numbers --min_word_length {{5}} {{url}}
```

- Output a wordlist from the given URL in debug mode including email addresses:

```
cewl --debug --email {{url}}
```

- Output a wordlist from the given URL using HTTP Basic or Digest authentication:

```
cewl --auth_type {{basic|digest}} --auth_user {{username}} --
auth_pass {{password}} {{url}}
```

- Output a wordlist from the given URL through a proxy:

```
cewl --proxy_host {{host}} --proxy_port {{port}} {{url}}
```

cfdisk

A program for managing partition tables and partitions on a hard disk using a curses U I.

More information: https://manned.org/cfdisk.

- S tart the partition manipulator with a specific device:

  ```
  cfdisk {{/dev /sdX }}
  ```

- Create a new partition table for a specific device and manage it:

  ```
  cfdisk - - zero {{/dev /sdX }}
  ```

chage

Change user account and password expiry information.

More information: https://manned.org/chage.

- List password information for the user:

```
chage -l {{username}}
```

- Enable password expiration in 10 days:

```
sudo chage -M {{10}} {{username}}
```

- Disable password expiration:

```
sudo chage -M -1 {{username}}
```

- Set account expiration date:

```
sudo chage -E {{YYYY-MM-DD}}
```

- Force user to change password on next log in:

```
sudo chage -d 0
```

chattr

Change attributes of files or directories.

More information: https://manned.org/chattr.

- Make a file or directory immutable to changes and deletion, even by superuser:

  ```
  chattr + i {{path/to/file_or_directory}}
  ```

- Make a file or directory mutable:

  ```
  chattr - i {{path/to/file_or_directory}}
  ```

- R ecursively make an entire directory and contents immutable:

  ```
  chattr - R  + i {{path/to/directory}}
  ```

chcpu

Enable/disable a system's CPUs.

More information: https://manned.org/chcpu.

- Disable CPUs via a list of CPU ID numbers:

```
chcpu -d {{1,3}}
```

- Enable a set of CPUs via a range of CPU ID numbers:

```
chcpu -e {{1-10}}
```

check-support-status

Identify installed Debian packages for which support has had to be limited or prematurely ended.

More information: https://manpages.debian.org/buster/debian-security-support/check-support-status.1.en.html

- Display packages whose support is limited, has already ended or will end earlier than the distribution's end of life:

  ```
  check-support-status
  ```

- Display only packages whose support has ended:

  ```
  check-support-status --type {{ended}}
  ```

- Skip printing a headline:

  ```
  check-support-status --no-heading
  ```

chfn

Update `finger` info for a user.

More information: https://manned.org/chfn.

- Update a user's "Name" field in the output of `finger`:

  ```
  chfn -f {{new_display_name}} {{username}}
  ```

- Update a user's "Office Room Number" field for the output of `finger`:

  ```
  chfn -o {{new_office_room_number}} {{username}}
  ```

- Update a user's "Office Phone Number" field for the output of `finger`:

  ```
  chfn -p {{new_office_telephone_number}} {{username}}
  ```

- Update a user's "Home Phone Number" field for the output of `finger`:

  ```
  chfn -h {{new_home_telephone_number}} {{username}}
  ```

chkconfig

Manage the runlevel of services on CentOS 6.

More information: https://manned.org/chkconfig.

- List services with runlevel:

```
chkconfig --list
```

- Show a service's runlevel:

```
chkconfig --list {{ntpd}}
```

- Enable service at boot:

```
chkconfig {{sshd}} on
```

- Enable service at boot for runlevels 2, 3, 4, and 5:

```
chkconfig --level {{2 3 4 5}} {{sshd}} on
```

- Disable service at boot:

```
chkconfig {{ntpd}} off
```

- Disable service at boot for runlevel 3:

```
chkconfig --level {{3}} {{ntpd}} off
```

chronyc

Query the Chrony NTP daemon.

More information: https://chrony.tuxfamily.org/doc/4.0/chronyc.html

- Start chronyc in interactive mode:

  ```
  chronyc
  ```

- Display tracking stats for the Chrony daemon:

  ```
  chronyc tracking
  ```

- Print the time sources that Chrony is currently using:

  ```
  chronyc sources
  ```

- Display stats for sources currently used by chrony daemon as a time source:

  ```
  chronyc sourcestats
  ```

- Step the system clock immediately, bypassing any slewing:

  ```
  chronyc makestep
  ```

- Display verbose information about each NTP source:

  ```
  chronyc ntpdata
  ```

chrt

Manipulate the real-time attributes of a process.

More information: https://man7.org/linux/man-pages/man1/chrt.1.html

- Display attributes of a process:

  ```
  chrt --pid {{PID}}
  ```

- Display attributes of all threads of a process:

  ```
  chrt --all-tasks --pid {{PID}}
  ```

- Display the min/max priority values that can be used with `chrt`:

  ```
  chrt --max
  ```

- Set the scheduling policy for a process:

  ```
  chrt --pid {{PID}} --{{deadline|idle|batch|rr|fifo|other}}
  ```

clamav

Open-source anti-virus program.

Designed especially for e-mail scanning on mail gateways, but can be used in other contexts.

More information: https://www.clamav.net.

- Update virus definitions:

  ```
  freshclam
  ```

- Scan a file for viruses:

  ```
  clamscan {{path/to/file}}
  ```

- Scan directories recursively and print out infected files:

  ```
  clamscan --recursive --infected {{path/to/directory}}
  ```

- Scan directories recursively and move them into quarantine:

  ```
  clamscan --recursive --move={{directory}}
  ```

cmus

Command-line Music Player.

Use arrow keys to navigate, `<enter/return>` to select, and numbers 1-8 switch between different views.

More information: https://cmus.github.io.

- Open cmus into the specified directory (this will become your new working directory):

  ```
  cmus {{path/to/directory}}
  ```

- Add file/directory to library:

  ```
  :add {{path/to/file_or_directory}}
  ```

- Pause/unpause current song:

  ```
  c
  ```

- Toggle shuffle mode on/off:

  ```
  s
  ```

- Quit cmus:

  ```
  q
  ```

collectd

System statistics collection daemon.

More information: https://collectd.org/.

- Show usage help, including the program version:

  ```
  collectd -h
  ```

- Test the configuration file and then exit:

  ```
  collectd -t
  ```

- Test plugin data collection functionality and then exit:

  ```
  collectd -T
  ```

- Start collectd:

  ```
  collectd
  ```

- Specify a custom configuration file location:

  ```
  collectd -C {{path/to/file}}
  ```

- Specify a custom PID file location:

  ```
  collectd -P {{path/to/file}}
  ```

- Don't fork into the background:

  ```
  collectd -f
  ```

colrm

Remove columns from stdin.

More information: https://manned.org/colrm.

- Remove first column of stdin:

```
colrm {{1 1}}
```

- Remove from 3rd column till the end of each line:

```
colrm {{3}}
```

- Remove from the 3rd column till the 5th column of each line:

```
colrm {{3 5}}
```

compgen

A built-in command for auto-completion in bash, which is called on pressing TAB key twice.

More information: https://www.gnu.org/software/bash/manual/bash.html#_index-compgen.

- List all commands that you could run:

```
compgen -c
```

- List all aliases:

```
compgen -a
```

- List all functions that you could run:

```
compgen -A function
```

- Show shell reserved key words:

```
compgen -k
```

- See all available commands/aliases starting with 'ls':

```
compgen -ac {{ls}}
```

compose

An alias to a `run-mailcap` action compose.

Originally `run-mailcap` is used to mime-type/file.

More information: https://manned.org/compose.

- Compose action can be used to compose any existing file or new on default mailcap edit tool:

  ```
  compose {{filename}}
  ```

- With `run-mailcap`

  ```
  run-mailcap --action=compose {{filename}}
  ```

compsize

Calculate the compression ratio of a set of files on a btrfs filesystem.

See also `btrfs filesystem` for compressing a file by defragmenting it.

More information: https://github.com/kilobyte/compsize

- Calculate the current compression ratio for a file or directory:

  `sudo compsize {{path/to/file_or_directory}}`

- Don't traverse filesystem boundaries:

  `sudo compsize --one-file-system {{path/to/file_or_directory}}`

- Show raw byte counts instead of human-readable sizes:

  `sudo compsize --bytes {{path/to/file_or_directory}}`

conky

Light-weight system monitor for X .

More information: https://github.com/brndnmtthws/conky.

- Launch with default, built-in config:

  ```
  conky
  ```

- Create a new default config:

  ```
  conky -C > ~/.conkyrc
  ```

- Launch conky with a given config file:

  ```
  conky -c {{path/to/config}}
  ```

- Start in the background (daemonize):

  ```
  conky -d
  ```

- Align conky on the desktop:

  ```
  conky -a {{{top,bottom,middle}_{left,right,middle}}}
  ```

- Pause for 5 seconds at startup before launching:

  ```
  conky -p {{5}}
  ```

coredumpctl

Retrieve and process saved core dumps and metadata.

More information: https://www.freedesktop.org/software/systemd/man/coredumpctl.html.

- List all captured core dumps:

```
coredumpctl list
```

- List captured core dumps for a program:

```
coredumpctl list {{program}}
```

- Show information about the core dumps matching a program with PID:

```
coredumpctl info {{PID}}
```

- Invoke debugger using the last core dump of a program:

```
coredumpctl debug {{program}}
```

- Extract the last core dump of a program to a file:

```
coredumpctl --output={{path/to/file}} dump {{program}}
```

cp

Copy files and directories.

More information: https://www.gnu.org/software/coreutils/cp.

- Copy a file to another location:

```
cp {{path/to/sou rce_file. ext}} {{path/to/targ et_file. ext}}
```

- Copy a file into another directory, keeping the filename:

```
cp {{path/to/sou rce_file. ext}} {{path/to/
targ et_parent_directory}}
```

- R ecursively copy a directory' s contents to another location (if the destination exists, the directory is copied inside it):

```
cp - r {{path/to/sou rce_directory}} {{path/to/
targ et_directory}}
```

- Copy a directory recursively, in verbose mode (shows files as they are copied):

```
cp - v r {{path/to/sou rce_directory}} {{path/to/
targ et_directory}}
```

- Copy text files to another location, in interactive mode (prompts user before overwriting):

```
cp - i {{* . txt}} {{path/to/targ et_directory}}
```

- F ollow symbolic links before copying:

```
cp - L  {{link }} {{path/to/targ et_directory}}
```

- U se the full path of source files, creating any missing intermediate directories when copying:

```
cp - - parents {{sou rce/path/to/file}} {{path/to/targ et_file}}
```

cpufreq-aperf

Calculate the average CPU frequency over a time period.

Requires root privileges.

More information: https://manned.org/cpufreq-aperf

- Start calculating, defaulting to all CPU cores and 1 second refresh interval:

  ```
  sudo cpufreq-aperf
  ```

- Start calculating for CPU 1 only:

  ```
  sudo cpufreq-aperf -c {{1}}
  ```

- Start calculating with a 3 seconds refresh interval for all CPU cores:

  ```
  sudo cpufreq-aperf -i {{3}}
  ```

- Calculate only once:

  ```
  sudo cpufreq-aperf -o
  ```

cpufreq-info

A tool to show CPU frequency information.

More information: https://manned.org/cpufreq-info

- Show CPU frequency information for all CPUs:

  ```
  cpufreq-info
  ```

- Show CPU frequency information for the specified CPU:

  ```
  cpufreq-info -c {{cpu_number}}
  ```

- Show the allowed minimum and maximum CPU frequency:

  ```
  cpufreq-info -l
  ```

- Show the current minimum and maximum CPU frequency and policy in table format:

  ```
  cpufreq-info -o
  ```

- Show available CPU frequency policies:

  ```
  cpufreq-info -g
  ```

- Show current CPU work frequency in a human-readable format, according to the cpufreq kernel module:

  ```
  cpufreq-info -f -m
  ```

- Show current CPU work frequency in a human-readable format, by reading it from hardware (only available to root):

  ```
  sudo cpufreq-info -w -m
  ```

cpufreq-set

A tool to modify CPU frequency settings.

The frequency value should range between the output of command `cpufreq-info -l`.

More information: https://manned.org/cpufreq-set

- Set the CPU frequency policy of CPU 1 to "userspace":

```
sudo cpufreq-set -c {{1}} -g {{userspace}}
```

- Set the current minimum CPU frequency of CPU 1:

```
sudo cpufreq-set -c {{1}} --min {{min_frequency}}
```

- Set the current maximum CPU frequency of CPU 1:

```
sudo cpufreq-set -c {{1}} --max {{max_frequency}}
```

- Set the current work frequency of CPU 1:

```
sudo cpufreq-set -c {{1}} -f {{work_frequency}}
```

cpuid

Display detailed information about all CPUs.

More information: http://etallen.com/cpuid.html.

- Display information for all CPUs:

  ```
  cpu id
  ```

- Display information only for the current CPU:

  ```
  cpu id -1
  ```

- Display raw hex information with no decoding:

  ```
  cpu id -r
  ```

cpulimit

A tool to throttle the CPU usage of other processes.

More information: http://cpulimit.sourceforge.net/.

- Limit an existing process with PID 1234 to only use 25% of the CPU:

```
cpulimit --pid {{1234}} --limit {{25%}}
```

- Limit an existing program by its executable name:

```
cpulimit --exe {{program}} --limit {{25}}
```

- Launch a given program and limit it to only use 50% of the CPU:

```
cpulimit --limit {{50}} --  {{program arg1 arg2 ...}}
```

- Launch a program, limit its CPU usage to 50% and run cpulimit in the background:

```
cpulimit --limit {{50}} --background --  {{program}}
```

- Kill its process if the program's CPU usage goes over 50%:

```
cpulimit --limit 50  --kill --  {{program}}
```

- Throttle both it and its child processes so that none go about 25% CPU:

```
cpulimit --limit {{25}} --monitor-forks --  {{program}}
```

create_ ap

Create an AP (Access P oint) at any channel.

More information: https://github.com/obliq ue/create_ ap

- Create an open network with no passphrase:

  ```
  create_ap {{w lan0 }} {{eth0 }} {{access_point_ssid}}
  ```

- U se a WP A + WP A2 passphrase:

  ```
  create_ap {{w lan0 }} {{eth0 }} {{access_point_ssid}} {{passphrase}}
  ```

- Create an access point without Internet sharing:

  ```
  create_ap - n {{w lan0 }} {{acces_point_ssid}} {{passphrase}}
  ```

- Create a bridged network with Internet sharing:

  ```
  create_ap - m b ridg e {{w lan0 }} {{eth0 }} {{access_point_ssid}} {{passphrase}}
  ```

- Create a bridged network with Internet sharing and a pre-configured bridge interface:

  ```
  create_ap - m b ridg e {{w lan0 }} {{b r0 }} {{access_point_ssid}} {{passphrase}}
  ```

- Create an access port for Internet sharing from the same WiF i interface:

  ```
  create_ap {{w lan0 }} {{w lan0 }} {{access_point_ssid}} {{passphrase}}
  ```

- Choose a different WiF i adapter driver:

  ```
  create_ap - - driv er {{w ifi_adapter}} {{w lan0 }} {{eth0 }} {{access_point_ssid}} {{passphrase}}
  ```

cryptsetup

Manage plain dm-crypt and LUKS (Linux Unified Key Setup) encrypted volumes.

More information: https://gitlab.com/cryptsetup/cryptsetup/.

- Initialize a LUKS volume (overwrites all data on the partition):

  ```
  cryptsetup luksFormat {{/dev/sda1}}
  ```

- Open a LUKS volume and create a decrypted mapping at `/dev/mapper/{{target}}`:

  ```
  cryptsetup luksOpen {{/dev/sda1}} {{target}}
  ```

- Remove an existing mapping:

  ```
  cryptsetup luksClose {{target}}
  ```

- Change the LUKS volume's passphrase:

  ```
  cryptsetup luksChangeKey {{/dev/sda1}}
  ```

csplit

Split a file into pieces.

This generates files named "xx00", "xx01", and so on.

More information: https://www.gnu.org/software/coreutils/csplit.

- Split a file at lines 5 and 23:

```
csplit {{file}} {{5}} {{23}}
```

- Split a file every 5 lines (this will fail if the total number of lines is not divisible by 5):

```
csplit {{file}} {{5}} {*}
```

- Split a file every 5 lines, ignoring exact-division error:

```
csplit -k {{file}} {{5}} {*}
```

- Split a file at line 5 and use a custom prefix for the output files:

```
csplit {{file}} {{5}} -f {{prefix}}
```

- Split a file at a line matching a regular expression:

```
csplit {{file}} /{{regular_expression}}/
```

ctr

Manage **`containerd`** containers and images.

More information: https://containerd.io.

- List all containers (running and stopped):

  ```
  ctr containers list
  ```

- List all images:

  ```
  ctr images list
  ```

- Pull an image:

  ```
  ctr images pull {{image}}
  ```

- Tag an image:

  ```
  ctr images tag {{source_image}}:{{source_tag}} {{target_image}}:{{target_tag}}
  ```

ctrlaltdel

Utility to control what happens when CTRL+ALT+DEL is pressed.

- Get current setting:

  ```
  ctrlaltdel
  ```

- Set CTRL+ALT+DEL to reboot immediately, without any preparation:

  ```
  sudo ctrlaltdel hard
  ```

- Set CTRL+ALT+DEL to reboot "normally", giving processes a chance to exit first (send SIGINT to PID1):

  ```
  sudo ctrlaltdel soft
  ```

cuyo

T etris like game.

More information: https://www.karimmi.de/cuyo/.

- S tart a new game:

  `cu yo`

- N avigate the piece horiz ontally:

  `{{A | D }} O R {{L eft| R ig ht}} arrow k ey`

- T urn the piece:

  `{{W | U p arrow k ey}}`

- H ard drop the piece:

  `{{S | D ow n arrow k ey}}`

daemonize

Run a command (that does not daemonize itself) as a Unix daemon.

More information: http://software.clapper.org/daemonize/

- Run a command as a daemon:

  ```
  daemonize {{command}} {{command_arguments}}
  ```

- Write the pid to the specified file:

  ```
  daemonize -p {{path/to/pidfile}} {{command}}
  {{command_arguments}}
  ```

- Use a lock file to ensure that only one instance runs at a time:

  ```
  daemonize -l {{path/to/lockfile}} {{command}}
  {{command_arguments}}
  ```

- Use the specified user account:

  ```
  sudo daemonize -u {{user}} {{command}} {{command_arguments}}
  ```

datamash

Tool to perform basic numeric, textual and statistical operations on input textual data files.

More information: http://www.gnu.org/software/datamash/.

- Get max, min, mean and median of a single column of numbers:

```
seq 3 | datamash max 1 min 1 mean 1 median 1
```

- Get the mean of a single column of float numbers (floats must use "," and not "."):

```
echo -e '1.0\n2.5\n3.1\n4.3\n5.6\n5.7' | tr '.' ',' | datamash mean 1
```

- Get the mean of a single column of numbers with a given decimal precision:

```
echo -e '1\n2\n3\n4\n5\n5' | datamash -R {{number_of_decimals_wanted}} mean 1
```

- Get the mean of a single column of numbers ignoring "Na" and "NaN" (literal) strings:

```
echo -e '1\n2\nNa\n3\nNaN' | datamash --narm mean 1
```

dbus-daemon

The D-Bus message daemon, allowing multiple programs to exchange messages.

More information: https://www.freedesktop.org/wiki/Software/dbus/

- Run the daemon with a configuration file:

```
dbus-daemon --config-file {{path/to/file}}
```

- Run the daemon with the standard per-login-session message bus configuration:

```
dbus-daemon --session
```

- Run the daemon with the standard systemwide message bus configuration:

```
dbus-daemon --system
```

- Set the address to listen on and override the configuration value for it:

```
dbus-daemon --address {{address}}
```

- Output the process id to stdout:

```
dbus-daemon --print-pid
```

- Force the message bus to write to the system log for messages:

```
dbus-daemon --syslog
```

dconf write

Write a value to a dconf database path.

More information: https://developer.gnome.org/dconf.

- Write a string to a dconf path (note the nested q uotes):

```
dconf w rite {{/exam ple/dconf/path}} " ' {{E xam ple V alu e}}' "
```

- Write a boolean to a dconf path:

```
dconf w rite {{/exam ple/dconf/path}} {{tru e| false}}
```

- Write an integer to a dconf path:

```
dconf w rite {{/exam ple/dconf/path}} {{1 6 }}
```

- Write an array to a dconf path:

```
dconf w rite {{/exam ple/dconf/path}} " [ {{' M y F irst V alu e' ,  ' M y
S econd V alu e' }}] "
```

- Write an empty array to a dconf path:

```
dconf w rite {{/exam ple/dconf/path}} " @ as [ ] "
```

dconf

Simple tool for manipulating dconf databases.

See also `dconf write`

More information: https://developer.gnome.org/dconf.

- Print the value of a dconf path:

  `dconf read {{/example/dconf/path}}`

- List contents of a dconf path:

  `dconf list {{/example/dconf/path}}`

- Watch for dconf database changes in a path and subpaths:

  `dconf watch {{/example/dconf/path}}`

ddrescue

Data recovery tool that reads data from damaged block devices.

More information: https://www.gnu.org/software/ddrescue/.

- Take an image of a device, creating a log file:

```
sudo ddrescue {{/dev/sdb}} {{path/to/image.dd}} {{path/to/log.txt}}
```

- Clone Disk A to Disk B, creating a log file:

```
sudo ddrescue --force --no-scrape {{/dev/sdX}} {{/dev/sdY}} {{path/to/log.txt}}
```

debchange

T ool for maintenance of the debian/changelog file in a Debian source package.

More information: https://manpages.debian.org/debchange.

- Add a new version for a non-maintainer upload to the changelog:

  ```
  deb chang e - - nm u
  ```

- Add a changelog entry to the current version:

  ```
  deb chang e - - append
  ```

- Add a changelog entry to close the bug with specified ID:

  ```
  deb chang e - - closes {{b u g _id}}
  ```

debman

Read man pages from uninstalled packages.

More information: https://manpages.debian.org/latest/debian-goodies/debman.1.html

- Read a man page for a command that is provided by a specified package name:

  `debman -p {{package_name}} {{command_name}}`

- Specify a package version to download:

  `debman -p {{package_name}}={{version}} {{command_name}}`

- Read a man page in a `.deb` file:

  `debman -f {{path/to/filename.deb}} {{command_name}}`

debootstrap

Create a basic Debian system.

More information: https://wiki.debian.org/Debootstrap.

- Create a Debian stable release system inside the `deb ian- root` directory:

  ```
  su do deb ootstrap stab le {{path/to/deb ian- root/}} http: //
  deb . deb ian. org /deb ian
  ```

- Create an U buntu 2 0 .0 4 system inside the `foc al- root` directory with a local mirror:

  ```
  su do deb ootstrap focal {{path/to/focal- root/}} {{file: ///
  path/to/m irror/}}
  ```

- S witch to a bootstrapped system:

  ```
  su do chroot {{path/to/root}}
  ```

- List available releases:

  ```
  ls /u sr/share/deb ootstrap/scripts/
  ```

debugfs

An interactive ext2 /ext3 /ext4 filesystem debugger.

More information: https://manned.org/debugfs.

- Open the filesystem in read only mode:

  ```
  debugfs {{/dev/sdXN}}
  ```

- Open the filesystem in read write mode:

  ```
  debugfs -w {{/dev/sdXN}}
  ```

- Read commands from a specified file, execute them and then exit:

  ```
  debugfs -f {{path/to/cmd_file}} {{/dev/sdXN}}
  ```

- View the filesystem stats in debugfs console:

  ```
  stats
  ```

- Close the filesystem:

  ```
  close -a
  ```

- List all available commands:

  ```
  lr
  ```

debuild

Tool to build a Debian package from source.

More information: https://manpages.debian.org/debuild.

- Build the package in the current directory:

  ```
  debuild
  ```

- Build a binary package only:

  ```
  debuild -b
  ```

- Do not run lintian after building the package:

  ```
  debuild --no-lintian
  ```

deluser

Delete a user from the system.

N ote: all commands must be executed as root.

More information: https://manpages.debian.org/latest/adduser/deluser.html.

- R emove a user:

  ```
  delu ser {{u sernam e}}
  ```

- R emove a user and their home directory:

  ```
  delu ser - - rem ov e- hom e {{u sernam e}}
  ```

- R emove a user and their home, but backup their files into a `. tar. gz` file in the specified directory:

  ```
  delu ser - - b ack u p- to {{path/to/b ack u p_directory}} - - rem ov e-
  hom e {{u sernam e}}
  ```

- R emove a user, and all files owned by them:

  ```
  delu ser - - rem ov e- all- files {{u sernam e}}
  ```

dex

DesktopEntry Execution is a program to generate and execute DesktopEntry files of the Application type.

More information: https://github.com/jceb/dex

- Execute all programs in the autostart folders:

```
dex --autostart
```

- Execute all programs in the specified folders:

```
dex --autostart --search-paths {{path/to/directory1}}:{{path/to/directory2}}:{{path/to/directory3}}:
```

- Preview the programs would be executed in a GNOME specific autostart:

```
dex --autostart --environment {{GNOME}}
```

- Preview the programs would be executed in a regular autostart:

```
dex --autostart --dry-run
```

- Preview the value of the DesktopEntry property `Name`:

```
dex --property {{Name}} {{path/to/file.desktop}}
```

- Create a DesktopEntry for a program in the current directory:

```
dex --create {{path/to/file.desktop}}
```

- Execute a single program (with `Terminal=true` in the desktop file) in the given terminal:

```
dex --term {{terminal}} {{path/to/file.desktop}}
```

dget

Download Debian packages.

More information: https://manpages.debian.org/dget.

- Download a binary package:

```
dg et {{pack ag e_nam e}}
```

- Download and extract a package source from its `. dsc`file:

```
dg et {{http: //deb . deb ian. org /deb ian/pool/m ain/h/hask ell- tldr/
hask ell- tldr_0 . 4 . 0 - 2 . dsc}}
```

- Download a package source tarball from its `. dsc`file but don' t extract it:

```
dg et - d {{http: //deb . deb ian. org /deb ian/pool/m ain/h/hask ell-
tldr/hask ell- tldr_0 . 4 . 0 - 2 . dsc}}
```

diff3

Compare three files line by line.

More information: https://www.gnu.org/software/diffutils/manual/html_node/Invoking-diff3.html

- Compare files:

```
diff3 {{file1}} {{file2}} {{file3}}
```

- Show all changes, outlining conflicts:

```
diff3 --show-all {{file1}} {{file2}} {{file3}}
```

disown

Allow sub-processes to live beyond the shell that they are attached to.

See also the `jobs` command.

More information: https://www.gnu.org/software/bash/manual/bash.html#_index-disown.

- Disown the current job:

  ```
  disown
  ```

- Disown a specific job:

  ```
  disown %{{job_number}}
  ```

- Disown all jobs:

  ```
  disown -a
  ```

- Keep job (do not disown it), but mark it so that no future SIGHUP is received on shell exit:

  ```
  disown -h %{{job_number}}
  ```

dkms

A framework that allows for dynamic building of kernel modules.

More information: https://github.com/dell/dkms.

- List currently installed modules:

  ```
  dkms status
  ```

- Rebuild all modules for the currently running kernel:

  ```
  dkms autoinstall
  ```

- Install version 1.2.1 of the acpi_call module for the currently running kernel:

  ```
  dkms install -m {{acpi_call}} -v {{1.2.1}}
  ```

- Remove version 1.2.1 of the acpi_call module from all kernels:

  ```
  dkms remove -m {{acpi_call}} -v {{1.2.1}} --all
  ```

dmenu

Dynamic menu.

Creates a menu from a text input with each item on a new line.

More information: https://manned.org/dmenu.

- Display a menu of the output of the `ls` command:

```
{{ls}} | dmenu
```

- Display a menu with custom items separated by a new line (`\n`):

```
echo -e "{{red}}\n{{green}}\n{{blue}}" | dmenu
```

- Let the user choose between multiple items and save the selected one to a file:

```
echo -e "{{red}}\n{{green}}\n{{blue}}" | dmenu > {{color.txt}}
```

- Launch dmenu on a specific monitor:

```
ls | dmenu -m {{1}}
```

- Display dmenu at the bottom of the screen:

```
ls | dmenu -b
```

dmesg

Write the kernel messages to standard output.

More information: https://manned.org/dmesg.

- Show kernel messages:

```
dmesg
```

- Show kernel error messages:

```
dmesg --level=err
```

- Show kernel messages and keep reading new ones, similar to `tail -f` (available in kernels 3.5.0 and newer):

```
dmesg -w
```

- Show how much physical memory is available on this system:

```
dmesg | grep -i memory
```

- Show kernel messages 1 page at a time:

```
dmesg | less
```

- Show kernel messages with a timestamp (available in kernels 3.5.0 and newer):

```
dmesg -T
```

- Show kernel messages in human-readable form (available in kernels 3.5.0 and newer):

```
dmesg -H
```

- Colorize output (available in kernels 3.5.0 and newer):

```
dmesg -L
```

dmidecode

Display the DMI (alternatively known as SMBIOS) table contents in a human-readable format.

Requires root privileges.

More information: https://manned.org/dmidecode.

- Show all DMI table contents:

```
sudo dmidecode
```

- Show the BIOS version:

```
sudo dmidecode -s bios-version
```

- Show the system's serial number:

```
sudo dmidecode -s system-serial-number
```

- Show BIOS information:

```
sudo dmidecode -t bios
```

- Show CPU information:

```
sudo dmidecode -t processor
```

- Show memory information:

```
sudo dmidecode -t memory
```

dnf

Package management utility for RHEL, Fedora, and CentOS (replaces yum).

More information: https://dnf.readthedocs.io/.

- Upgrade installed packages to the newest available versions:

  ```
  sudo dnf upgrade
  ```

- Search packages via keywords:

  ```
  dnf search {{keywords}}
  ```

- Display details about a package:

  ```
  dnf info {{package}}
  ```

- Install a new package:

  ```
  sudo dnf install {{package}}
  ```

- Install a new package and assume yes to all questions:

  ```
  sudo dnf -y install {{package}}
  ```

- Remove a package:

  ```
  sudo dnf remove {{package}}
  ```

- List installed packages:

  ```
  dnf list --installed
  ```

- Find which packages provide a given file:

  ```
  dnf provides {{file}}
  ```

dnsrecon

DNS enumeration tool.

More information: https://github.com/darkoperator/dnsrecon.

- Scan a domain and save the results to a SQLite database:

  ```
  dnsrecon --domain {{example.com}} --db {{path/to/database.sqlite}}
  ```

- Scan a domain, specifying the nameserver and performing a zone transfer:

  ```
  dnsrecon --domain {{example.com}} --name_server {{nameserver.example.com}} --type axfr
  ```

- Scan a domain, using a brute-force attack and a dictionary of subdomains and hostnames:

  ```
  dnsrecon --domain {{example.com}} --dictionary {{path/to/dictionary.txt}} --type brt
  ```

- Scan a domain, performing a reverse lookup of IP ranges from the SPF record and saving the results to a JSON file:

  ```
  dnsrecon --domain {{example.com}} -s --json
  ```

- Scan a domain, performing a Google enumeration and saving the results to a CSV file:

  ```
  dnsrecon --domain {{example.com}} -g --csv
  ```

- Scan a domain, performing DNS cache snooping:

  ```
  dnsrecon --domain {{example.com}} --type snoop --name_server {{nameserver.example.com}} --dictionary {{path/to/dictionary.txt}}
  ```

- Scan a domain, performing zone walking:

  ```
  dnsrecon --domain {{example.com}} --type zonewalk
  ```

do-release-upgrade

T he U buntu release upgrader.

More information: https://ubuntu.com/server/docs/upgrade-introduction.

- U pgrade to the latest release:

  ```
  su do do- release- u pg rade
  ```

- U pgrade to the latest development release:

  ```
  su do do- release- u pg rade - - dev el- release
  ```

- U pgrade to the latest proposed release:

  ```
  su do do- release- u pg rade - - proposed
  ```

dockerd

A persistent process to start and manage docker containers.

More information: https://docs.docker.com/engine/reference/commandline/dockerd/.

- Run docker daemon:

  ```
  dockerd
  ```

- Run docker daemon and config it to listen to specific sockets(unix,tcp):

  ```
  dockerd --host unix://{{path/to/tmp.sock}} --host tcp://{{ip}}
  ```

- Run with specific daemon PID file:

  ```
  dockerd --pidfile {{path/to/pid_file}}
  ```

- Run in debug mode:

  ```
  dockerd --debug
  ```

- Run and set a specific log level:

  ```
  dockerd --log-level={{debug|info|warn|error|fatal}}
  ```

dolphin

KDE file manager.

More information: https://apps.kde.org/dolphin/.

- Launch Dolphin:

  ```
  dolphin
  ```

- Launch Dolphin and display a specific directory:

  ```
  dolphin {{path/to/directory}}
  ```

- Launch Dolphin with a specific file or directory selected:

  ```
  dolphin --select {{path/to/file_or_directory}}
  ```

- Launch Dolphin in a separated window:

  ```
  dolphin --new-window
  ```

- Launch Dolphin in split view:

  ```
  dolphin --split
  ```

- Start the Dolphin daemon (only required to use the DBus interface):

  ```
  dolphin --daemon
  ```

- Display help:

  ```
  dolphin --help
  ```

dos2unix

Change DOS-style line endings to Unix-style.

Replaces CRLF with CR.

More information: https://manned.org/dos2unix

- Change the line endings of a file:

  ```
  dos2unix {{filename}}
  ```

- Create a copy with Unix-style line endings:

  ```
  dos2unix -n {{filename}} {{new_filename}}
  ```

dpkg-deb

Pack, unpack and provide information about Debian archives.

More information: https://manpages.debian.org/buster/dpkg/dpkg-deb.1.en.html

- Display information about a package:

```
dpkg-deb --info {{path/to/file.deb}}
```

- Display the package's name and version on one line:

```
dpkg-deb --show {{path/to/file.deb}}
```

- List the package's contents:

```
dpkg-deb --contents {{path/to/file.deb}}
```

- Extract package's contents into a directory:

```
dpkg-deb --extract {{path/to/file.deb}} {{path/to/directory}}
```

- Create a package from a specified directory:

```
dpkg-deb --build {{path/to/directory}}
```

dpkg-q uery

A tool that shows information about installed packages.

More information: https://manpages.debian.org/latest/dpkg/dpkg-q uery.1 .html

- List all installed packages:

```
dpk g - q u ery - l
```

- List installed packages matching a pattern:

```
dpk g - q u ery - l ' {{pattern}}'
```

- List all files installed by a package:

```
dpk g - q u ery - L {{pack ag e_nam e}}
```

- S how information about a package:

```
dpk g - q u ery - s {{pack ag e_nam e}}
```

dpkg

Debian package manager.

More information: https://manpages.debian.org/buster/dpkg/dpkg.1.en.html

- Install a package:

```
dpkg -i {{path/to/file.deb}}
```

- Remove a package:

```
dpkg -r {{package_name}}
```

- List installed packages:

```
dpkg -l {{pattern}}
```

- List a package's contents:

```
dpkg -L {{package_name}}
```

- List contents of a local package file:

```
dpkg -c {{path/to/file.deb}}
```

- Find out which package owns a file:

```
dpkg -S {{filename}}
```

dstat

Versatile tool for generating system resource statistics.

More information: http://dag.wieers.com/home-made/dstat.

- Display CPU, disk, net, paging and system statistics:

```
dstat
```

- Display statistics every 5 seconds and 4 updates only:

```
dstat {{5}} {{4}}
```

- Display CPU and memory statistics only:

```
dstat --cpu --mem
```

- List all available dstat plugins:

```
dstat --list
```

- Display the process using the most memory and most CPU:

```
dstat --top-mem --top-cpu
```

- Display battery percentage and remaining battery time:

```
dstat --battery --battery-remain
```

dumpe2fs

Print the superblock and blocks group information for ext2/ext3/ext4 filesystems.

Unmount the partition before running this command using `umount {{device}}`

More information: https://manned.org/dumpe2fs

- Display ext2, ext3 and ext4 filesystem information:

  `dumpe2fs {{/dev/sdXN}}`

- Display the blocks which are reserved as bad in the filesystem:

  `dumpe2fs -b {{/dev/sdXN}}`

- Force display filesystem information even with non-recognisable feature flags:

  `dumpe2fs -f {{/dev/sdXN}}`

- Only display the superblock information and not any of the block group descriptor detail information:

  `dumpe2fs -h {{/dev/sdXN}}`

- Print the detailed group information block numbers in hexadecimal format:

  `dumpe2fs -x {{/dev/sdXN}}`

dunstify

A notification tool that is an extension of notify-send, but has more features based around dunst.

Works with all options that work for notify-send.

More information: https://wiki.archlinux.org/title/Dunst#_Dunstify

- S how a notification with a given title and message:

  ```
  du nstify " {{T itle}}"  " {{M essag e}}"
  ```

- S how a notification with specified urgency:

  ```
  du nstify " {{T itle}}"  " {{M essag e}}"  - u  {{low | norm al| critical}
  ```

- S pecify a message ID (overwrites any previous messages with the same ID):

  ```
  du nstify " {{T itle}}"  " {{M essag e}}"  - r {{1 2 3 }}
  ```

- T o see other possible options:

  ```
  notify- send - - help
  ```

duperemove

Finds duplicate filesystem extents and optionally schedule them for deduplication.

An extent is small part of a file inside the filesystem.

On some filesystems one extent can be referenced multiple times, when parts of the content of the files are identical.

More information: https://markfasheh.github.io/duperemove/.

- Search for duplicate extents in a directory and show them:

  ```
  duperemove -r {{path/to/directory}}
  ```

- Deduplicate duplicate extents on a Btrfs or XFS (experimental) filesystem:

  ```
  duperemove -r -d {{path/to/directory}}
  ```

- Use a hash file to store extent hashes (less memory usage and can be reused on subsequent runs):

  ```
  duperemove -r -d --hashfile={{path/to/hashfile}} {{path/to/directory}}
  ```

- Limit I/O threads (for hashing and dedupe stage) and CPU threads (for duplicate extent finding stage):

  ```
  duperemove -r -d --hashfile={{path/to/hashfile}} --io-threads={{N}} --cpu-threads={{N}} {{path/to/directory}}
  ```

e2 freefrag

Print the free space fragmentation information for ext2/ext3/ext4 filesystems.

More information: https://manned.org/e2freefrag

- Check how many free blocks are present as contiguous and aligned free space:

  ```
  e2freefrag {{/dev/sdX N}}
  ```

- Specify chunk size in kilobytes to print how many free chunks are available:

  ```
  e2freefrag -c {{chunk_size_in_kb}} {{/dev/sdX N}}
  ```

e2 fsck

Check a Linux ext2 /ext3 /ext4 filesystem. The filesystem should be unmounted at the time the command is run.

More information: https://manned.org/e2_fsck

- Check filesystem, reporting any damaged blocks:

```
e2 fsck {{/dev /sdX N }}
```

- Check filesystem and automatically repair any damaged blocks:

```
e2 fsck -p {{/dev /sdX N }}
```

- Check filesystem in read only mode:

```
e2 fsck -c {{/dev /sdX N }}
```

e2image

Save critical ext2/ext3/ext4 filesystem metadata to a file.

More information: https://manned.org/e2image

- Write metadata located on device to a specific file:

```
e2image {{/dev/sdX}} {{path/to/image_file}}
```

- Print metadata located on device to stdout:

```
e2image {{/dev/sdX}} -
```

- Restore the filesystem metadata back to the device:

```
e2image -I {{/dev/sdX}} {{path/to/image_file}}
```

- Create a large raw sparse file with metadata at proper offsets:

```
e2image -r {{/dev/sdX}} {{path/to/image_file}}
```

- Create a QCOW2 image file instead of a normal or raw image file:

```
e2image -Q {{/dev/sdX}} {{path/to/image_file}}
```

e2 label

Change the label on an ext2 /ext3 /ext4 filesystem.

More information: https://manned.org/e2 label

- Change the volume label on a specific ext partition:

```
e2 lab el {{/dev /sda1 }} " {{lab el_nam e}}"
```

e2 undo

Replay undo logs for an ext2/ext3/ext4 filesystem.

This can be used to undo a failed operation by an e2fsprogs program.

More information: https://man7.org/linux/man-pages/man8/e2undo.8.html

- Display information about a specific undo file:

  ```
  e2undo -h {{path/to/undo_file}} {{/dev/sdXN}}
  ```

- Perform a dry-run and display the candidate blocks for replaying:

  ```
  e2undo -nv {{path/to/undo_file}} {{/dev/sdXN}}
  ```

- Perform an undo operation:

  ```
  e2undo {{path/to/undo_file}} {{/dev/sdXN}}
  ```

- Perform an undo operation and display verbose information:

  ```
  e2undo -v {{path/to/undo_file}} {{/dev/sdXN}}
  ```

- Write the old contents of the block to an undo file before overwriting a file system block:

  ```
  e2undo -z {{path/to/file.e2undo}} {{path/to/undo_file}} {{/dev/sdXN}}
  ```

e4defrag

Defragment an ext4 filesystem.

More information: https://manned.org/e4defrag

- Defragment the filesystem:

```
e4defrag {{/dev/sdX N}}
```

- See how fragmented a filesystem is:

```
e4defrag -c {{/dev/sdX N}}
```

- Print errors and the fragmentation count before and after each file:

```
e4defrag -v {{/dev/sdX N}}
```

ebuild

A low level interface to the Gentoo Portage system.

More information: https://wiki.gentoo.org/wiki/Ebuild

- Create or update the package manifest:

  ```
  ebuild {{path/to/file.ebuild}} manifest
  ```

- Clean the temporary build directories for the build file:

  ```
  ebuild {{path/to/file.ebuild}} clean
  ```

- Fetch sources if they do not exist:

  ```
  ebuild {{path/to/file.ebuild}} fetch
  ```

- Extract the sources to a temporary build directory:

  ```
  ebuild {{path/to/file.ebuild}} unpack
  ```

- Compile the extracted sources:

  ```
  ebuild {{path/to/file.ebuild}} compile
  ```

- Install the package to a temporary install directory:

  ```
  ebuild {{path/to/file.ebuild}} install
  ```

- Install the temporary files to the live filesystem:

  ```
  ebuild {{path/to/file.ebuild}} qmerge
  ```

- Fetch, unpack, compile, install and qmerge the specified ebuild file:

  ```
  ebuild {{path/to/file.ebuild}} merge
  ```

edit

An alias to a `run-mailcap` action edit.

Originally `run-mailcap` is used to process/edit mime-type/file.

More information: https://www.computerhope.com/unix/uedit.htm.

- Edit action can be used to view any file on default mailcap explorer:

  ```
  edit {{filename}}
  ```

- With `run-mailcap`

  ```
  run-mailcap --action=edit {{filename}}
  ```

edquota

Edit quotas for a user or group. By default it operates on all filesystems with quotas.

Quota information is stored permanently in the `quota.user` and `quota.group` files in the root of the filesystem.

More information: https://manned.org/edquota

- Edit quota of the current user:

```
edquota --user $(whoami)
```

- Edit quota of a specific user:

```
sudo edquota --user {{username}}
```

- Edit quota for a group:

```
sudo edquota --group {{group}}
```

- Restrict operations to a given filesystem (by default edquota operates on all filesystems with quotas):

```
sudo edquota --filesystem {{filesystem}}
```

- Edit the default grace period:

```
sudo edquota -t
```

- Duplicate a quota to other users:

```
sudo edquota -p {{reference_user}} {{destination_user1}} {{destination_user2}}
```

efibootmgr

Manipulate the U E F I B oot Manager.

More information: https://manned.org/efibootmgr.

- List the current settings / bootnums:

```
efib ootm g r
```

- List the filepaths:

```
efib ootm g r - v
```

- Add U E F I S hell v2 as a boot option:

```
su do efib ootm g r - c - d {{/dev /sda1 }} - l
{{\ E F I \ tools\ S hell. efi}} - L " {{U E F I S hell}}"
```

- Change the current boot order:

```
su do efib ootm g r - o {{0 0 0 2 , 0 0 0 8 , 0 0 0 1 , 0 0 0 5 }}
```

- Delete a boot option:

```
su do efib ootm g r - b {{0 0 0 8 }} - - delete- b ootnu m
```

eix

Utilities for searching local Gentoo packages.

Update local package cache using `eix-update`

More information: https://wiki.gentoo.org/wiki/Eix

- Search for a package:

  ```
  eix {{package_name}}
  ```

- Search for installed packages:

  ```
  eix --installed {{package_name}}
  ```

- Search in package descriptions:

  ```
  eix --description "{{description}}"
  ```

- Search by package license:

  ```
  eix --license {{license}}
  ```

- Exclude results from search:

  ```
  eix --not --license {{license}}
  ```

ej ect

E j ect cds, fl oppy disks and tape drives.

More information: https://manned.org/ej ect

- Display the default device:

```
ej ect - d
```

- E j ect the default device:

```
ej ect
```

- E j ect a specific device (the default order is cd-rom, scsi, fl oppy and tape):

```
ej ect {{/dev /cdrom }}
```

- T oggle whether a device' s tray is open or closed:

```
ej ect - T {{/dev /cdrom }}
```

- E j ect a cd drive:

```
ej ect - r {{/dev /cdrom }}
```

- E j ect a fl oppy drive:

```
ej ect - f {{/m nt/floppy}}
```

- E j ect a tape drive:

```
ej ect - q {{/m nt/tape}}
```

emerge

Gentoo Linux package manager utility.

More information: https://wiki.gentoo.org/wiki/Portage#emerge

- Synchronize all packages:

```
emerge --sync
```

- Update all packages, including dependencies:

```
emerge -uDNav @world
```

- Resume a failed updated, skipping the failing package:

```
emerge --resume --skipfirst
```

- Install a new package, with confirmation:

```
emerge -av {{package_name}}
```

- Remove a package, with confirmation:

```
emerge -Cav {{package_name}}
```

- Remove orphaned packages (that were installed only as dependencies):

```
emerge -avc
```

- Search the package database for a keyword:

```
emerge -S {{keyword}}
```

enum4linux

Tool for enumerating Windows and Samba information from remote systems.

It attempts to offer similar functionality to enum.exe formerly available from www.bindview.com.

More information: https://labs.portcullis.co.uk/tools/enum4linux/

- Try to enumerate using all methods:

  ```
  enum4linux -a {{remote_host}}
  ```

- Enumerate using given login credentials:

  ```
  enum4lix -u {{user_name}} -p {{password}} {{remote_host}}
  ```

- List usernames from a given host:

  ```
  enum4lix -U {{remote_host}}
  ```

- List shares:

  ```
  enum4lix -S {{remote_host}}
  ```

- Get OS information:

  ```
  enum4lix -o {{remote_host}}
  ```

eq uery

View information about Portage packages.

More information: https://wiki.gentoo.org/wiki/Equery

- List all installed packages:

  ```
  equery list '*'
  ```

- Search for installed packages in the Portage tree and in overlays:

  ```
  equery list -po {{package name}}
  ```

- List all packages that depend on a given package:

  ```
  equery depends {{package name}}
  ```

- List all packages that a given package depends on:

  ```
  equery depgraph {{package name}}
  ```

- List all files installed by a package:

  ```
  equery files --tree {{package name}}
  ```

etckeeper

Track system configuration files in Git.

More information: http://etckeeper.branchable.com/.

- Set up a Git repo and perform various setup tasks (run from `/etc`):

  ```
  sudo etckeeper init
  ```

- Commit all changes in `/etc`:

  ```
  sudo etckeeper commit {{message}}
  ```

- Run arbitrary Git commands:

  ```
  sudo etckeeper vcs {{status}}
  ```

- Check if there are uncommitted changes (only returns an exit code):

  ```
  sudo etckeeper unclean
  ```

- Destroy existing repo and stop tracking changes:

  ```
  sudo etckeeper uninit
  ```

ethtool

Display and modify N etwork Interface Controller (N IC) parameters.

More information: [http://man7 .org/linux/man-pages/man8 /ethtool.8 .html](http://man7.org/linux/man-pages/man8/ethtool.8.html)

- Display the current settings for an interface:

```
ethtool {{eth0 }}
```

- Display the driver information for an interface:

```
ethtool - - driv er {{eth0 }}
```

- Display the network usage statistics for an interface:

```
ethtool - - statistics {{eth0 }}
```

- B link one or more LE Ds on an interface for 1 0 seconds:

```
ethtool - - identify {{eth0 }} {{1 0 }}
```

- S et the link speed, duplex mode, and parameter auto-negotiation for a given interface:

```
ethtool - s {{eth0 }} speed {{1 0 | 1 0 0 | 1 0 0 0 }} du plex {{half|
fu ll}} au toneg {{on| off}}
```

eval

Execute arguments as a single command in the current shell and return its result.

More information: https://manned.org/eval.

- Call `echo` with the "foo" argument:

  ```
  eval "{{echo foo}}"
  ```

- Set a variable in the current shell:

  ```
  eval "{{foo=bar}}"
  ```

exif

Show and change EXIF information in JPEG files.

More information: https://github.com/libexif/exif/.

- Show all recognized EXIF information in an image:

  ```
  exif {{path/to/image.jpg}}
  ```

- Show a table listing known EXIF tags and whether each one exists in an image:

  ```
  exif --list-tags --no-fixup {{image.jpg}}
  ```

- Extract the image thumbnail into the file `thumbnail.jpg`

  ```
  exif --extract-thumbnail --output={{thumbnail.jpg}} {{image.jpg}}
  ```

- Show the raw contents of the "Model" tag in the given image:

  ```
  exif --ifd={{0}} --tag={{Model}} --machine-readable {{image.jpg}}
  ```

- Change the value of the "Artist" tag to John Smith and save to `new.jpg`

  ```
  exif --output={{new.jpg}} --ifd={{0}} --tag="{{Artist}}" --set-value="{{John Smith}}" --no-fixup {{image.jpg}}
  ```

expect

Script executor that interacts with other programs that require user input.

More information: https://manned.org/expect.

- Execute an expect script from a file:

  ```
  expect {{path/to/file}}
  ```

- Execute a specified expect script:

  ```
  expect -c "{{commands}}"
  ```

- Enter an interactive REPL (use `exit` or Ctrl+D to exit):

  ```
  expect -i
  ```

export

Command to mark shell variables in the current environment to be exported with any newly forked child processes.

More information: https://www.gnu.org/software/bash/manual/bash.html#_index-export.

- Set a new environment variable:

```
export {{VARIABLE}}={{value}}
```

- Remove an environment variable:

```
export -n {{VARIABLE}}
```

- Mark a shell function for export:

```
export -f {{FUNCTION_NAME}}
```

- Append something to the PATH variable:

```
export PATH=$PATH:{{path/to/append}}
```

extrace

Trace exec() calls.

More information: https://github.com/chneukirchen/extrace.

- Trace all program executions occurring on the system:

```
sudo extrace
```

- Run a command and only trace descendants of this command:

```
sudo extrace {{command}}
```

- Print the current working directory of each process:

```
sudo extrace -d
```

- Resolve the full path of each executable:

```
sudo extrace -l
```

- Display the user running each process:

```
sudo extrace -u
```

extundelete

Recover deleted files from ext3 or ext4 partitions by parsing the journal.

See also **date** for Unix time information and **umount** for unmounting partitions.

More information: http://extundelete.sourceforge.net.

- Restore all deleted files inside partition N on device X:

  ```
  sudo extundelete {{/dev/sdXN}} --restore-all
  ```

- Restore a file from a path relative to root (Do not start the path with **/**):

  ```
  extundelete {{/dev/sdXN}} --restore-file {{path/to/file}}
  ```

- Restore a directory from a path relative to root (Do not start the path with **/**):

  ```
  extundelete {{/dev/sdXN}} --restore-directory {{path/to/directory}}
  ```

- Restore all files deleted after January 1st, 2020 (in Unix time):

  ```
  extundelete {{/dev/sdXN}} --restore-all --after {{1577840400}}
  ```

eyeD3

Read and manipulate metadata of MP3 files.

More information: https://eyed3.readthedocs.io/en/latest/

- View information about an MP3 file:

  ```
  eyeD3 {{filename.mp3}}
  ```

- Set the title of an MP3 file:

  ```
  eyeD3 --title "{{Title}}" {{filename.mp3}}
  ```

- Set the album of all the MP3 files in a directory:

  ```
  eyeD3 --album "{{Album Name}}" {{*.mp3}}
  ```

- Set the front cover art for an MP3 file:

  ```
  eyeD3 --add-image {{front_cover.jpeg}}:FRONT_COVER:{{filename.mp3}}
  ```

f5 fpc

A proprietry commercial SSL VPN client by BIG-IP Edge.

More information: https://techdocs.f5.com/kb/en-us/products/big-ip_apm/manuals/product/apm-client-configuration-11-4-0/4.html

- Open a new VPN connection:

  ```
  sudo f5 fpc --start
  ```

- Open a new VPN connection to a specific host:

  ```
  sudo f5 fpc --start --host {{host.example.com}}
  ```

- Specify a username (user will be prompted for a password):

  ```
  sudo f5 fpc --start --host {{host.example.com}} --username {{user}}
  ```

- Show the current VPN status:

  ```
  sudo f5 fpc --info
  ```

- Shutdown the VPN connection:

  ```
  sudo f5 fpc --stop
  ```

fail2ban-client

Configure and control fail2ban server.

More information: https://github.com/fail2ban/fail2ban

- Retrieve current status of the jail service:

  ```
  fail2ban-client status {{jail}}
  ```

- Remove the specified IP from the jail service's ban list:

  ```
  fail2ban-client set {{jail}} unbanip {{ip}}
  ```

- Verify fail2ban server is alive:

  ```
  fail2ban-client ping
  ```

faketime

Fake the system time for a given command.

More information: https://manpages.ubuntu.com/manpages/trusty/man1/faketime.1.html

- Fake the time to this evening, before printing the result of `date`:

  ```
  faketime '{{today 23:30}}' {{date}}
  ```

- Open a new `bash` shell, which uses yesterday as the current date:

  ```
  faketime '{{yesterday}}' {{bash}}
  ```

- Simulate how any program would act next friday night:

  ```
  faketime '{{next Friday 1am}}' {{path/to/any/program}}
  ```

fallocate

Reserve or deallocate disk space to files.

The utility allocates space without zeroing.

More information: https://manned.org/fallocate.

- Reserve a file taking up 700 MB of disk space:

```
fallocate --length {{700M}} {{path/to/file}}
```

- Shrink an already allocated file by 200 MB:

```
fallocate --collapse-range --length {{200M}} {{path/to/file}}
```

- Shrink 20 MB of space after 100 MB in a file:

```
fallocate --collapse-range --offset {{100M}} --length {{20M
{{path/to/file}}
```

fatlabel

Sets or gets the label of a FAT32 partition.

More information: https://manned.org/fatlabel.

- Get the label of a FAT32 partition:

  ```
  fatlabel {{/dev/sda1}}
  ```

- Set the label of a FAT32 partition:

  ```
  fatlabel {{/dev/sdc3}} "{{new_label}}"
  ```

fc-cache

Scan font directories in order to build font cache files.

More information: https://manned.org/fc-cache.

- Generate font cache files:

```
fc-cache
```

- Force a rebuild of all font cache files, without checking if cache is up-to-date:

```
fc-cache -f
```

- Erase font cache files, then generate new font cache files:

```
fc-cache -r
```

fc-list

List available fonts installed on the system.

More information: https://manned.org/fc-list.

- Return a list of installed fonts in your system:

```
fc-list
```

- Return a list of installed fonts with given name:

```
fc-list | grep '{{DejaVu Serif}}'
```

- Return the number of installed fonts in your system:

```
fc-list | wc -l
```

fc-match

Match available fonts.

More information: https://manned.org/fc-match.

- Return a sorted list of best matching fonts:

```
fc-match -s '{{DejaVu Serif}}'
```

fc-pattern

S hows information about a font matching a pattern.

More information: https://manned.org/fc-pattern.

- Display default information about a font:

```
fc- pattern - - defau lt ' {{D ej aV u  S erif}}'
```

- Display config information about a font:

```
fc- pattern - - config  ' {{D ej aV u  S erif}}'
```

fc

O pen the most recent command and edit it.

More information: https://manned.org/fc.

- O pen in the default system editor:

  ```
  fc
  ```

- S pecify an editor to open with:

  ```
  fc -e {{'emacs'}}
  ```

- List recent commands from history:

  ```
  fc -l
  ```

fcrackzip

Zip archive password cracking utility.

More information: https://manned.org/fcrackzip

- Brute-force a password with a length of 4 to 8 characters, and contains only alphanumeric characters (order matters):

```
fcrackzip --brute-force --length 4-8 --charset aA1 {{archive}}
```

- Brute-force a password in verbose mode with a length of 3 characters that only contains lowercase characters, `$` and `%`:

```
fcrackzip -v --brute-force --length 3 --charset a:$% {{archive}}
```

- Brute-force a password that contains only lowercase and special characters:

```
fcrackzip --brute-force --length 4 --charset a! {{archive}}
```

- Brute-force a password containing only digits, starting from the password `12345`:

```
fcrackzip --brute-force --length 5 --charset 1 --init-password 12345 {{archive}}
```

- Crack a password using a wordlist:

```
fcrackzip --use-unzip --dictionary --init-password {{wordlist}} {{archive}}
```

- Benchmark cracking performance:

```
fcrackzip --benchmark
```

fdisk

A program for managing partition tables and partitions on a hard disk.

See also: `partprobe`.

More information: https://manned.org/fdisk.

- List partitions:

  ```
  sudo fdisk -l
  ```

- Start the partition manipulator:

  ```
  sudo fdisk {{/dev/sdX}}
  ```

- Once partitioning a disk, create a partition:

  ```
  n
  ```

- Once partitioning a disk, select a partition to delete:

  ```
  d
  ```

- Once partitioning a disk, view the partition table:

  ```
  p
  ```

- Once partitioning a disk, write the changes made:

  ```
  w
  ```

- Once partitioning a disk, discard the changes made:

  ```
  q
  ```

- Once partitioning a disk, open a help menu:

  ```
  m
  ```

feedreader

A G U I desktop R S S client.

More information: https://j angernert.github.io/F eedR eader/

- P rint the count of unread articles:

  ```
  feedreader - - u nreadC ou nt
  ```

- Add a U R L for a feed to follow:

  ```
  feedreader - - addF eed= {{feed_u rl}}
  ```

- G rab a specific article using its U R L:

  ```
  feedreader - - g rab A rticle= {{article_u rl}}
  ```

- Download all images from a specific article:

  ```
  feedreader - - u rl= {{feed_u rl}} - - g rab I m ag es= {{article_path}}
  ```

- P lay media from a U R L:

  ```
  feedreader - - playM edia= {{article_u rl}}
  ```

feh

Lightweight image viewing utility.

More information: https://feh.finalrewind.org.

- View images locally or using a URL:

  ```
  feh {{path/to/images}}
  ```

- View images recursively:

  ```
  feh --recursive {{path/to/images}}
  ```

- View images without window borders:

  ```
  feh --borderless {{path/to/images}}
  ```

- Exit after the last image:

  ```
  feh --cycle-once {{path/to/images}}
  ```

- Set the slideshow cycle delay:

  ```
  feh --slideshow-delay {{seconds}} {{path/to/images}}
  ```

- Set your wallpaper (centered, filled, maximized, scaled or tiled):

  ```
  feh --bg-{{center|fill|max|scale|tile}} {{path/to/image}}
  ```

- Create a montage of all images within a directory. Outputs as a new image:

  ```
  feh --montage --thumb-height {{150}} --thumb-width {{15
  index-info "{{%nn%wx%h}}" --output {{path/to/
  montage_image.png}}
  ```

ffuf

Subdomain and directory discovery tool.

More information: https://github.com/ffuf/ffuf.

- Discover directories using a [w]ordlist on a target [u]rl with [c]oloriz ed and [v]erbose ouput:

```
ffuf -w {{path/to/wordlist}} -u {{https://target/FUZZ}} -c -v
```

- Fuzz host-[H]eaders with a host file on a traget website and [m]atch HTTP 200 [c]ode responses:

```
ffuf -w {{hosts.txt}} -u {{https://example.org}} -H "{{Host:
FUZZ}}" -mc {{200}}
```

- Discover directories using a [w]ordlist on a target website with a max individual j ob time of 60 seconds and recursion discovery depth of 2 levels:

```
ffuf -w {{path/to/wordlist}} -u {{https://target/FUZZ}} -
maxtime-job {{60}} -recursion -recursion-depth {{2}}
```

- Fuzz GET parameter on a target website and [f]ilter out message [s]iz e response of 4242 bytes:

```
ffuf -w {{path/to/param_names.txt}} -u {{https://target/
script.php?FUZZ=test_value}} -fs {{4242}}
```

- Fuzz POST method with POST [d]ata of password on a target website and [f]ilter out HTTP response [c]ode 401:

```
ffuf -w {{path/to/postdata.txt}} -X {{POST}} -d
"{{username=admin\&password=FUZZ}}" -u {{https://target/
login.php}} -fc {{401}}
```

- Discover subdomains using a subdoomain list on a target website:

```
ffuf -w {{subdomains.txt}} -u {{https://website.com}} -H
"{{Host: FUZZ.website.com}}"
```

rename

Rename multiple files.

NOTE: this page refers to the command from the `file-rename` Debian package.

More information: https://manned.org/file-rename.

- Rename files using a Perl Common Regular Expression (substitute 'foo' with 'bar' wherever found):

  `rename {{'s/foo/bar/'}} {{*}}`

- Dry-run - display which renames would occur without performing them:

  `rename -n {{'s/foo/bar/'}} {{*}}`

- Force renaming even if the operation would remove existing destination files:

  `rename -f {{'s/foo/bar/'}} {{*}}`

- Convert filenames to lower case (use `-f` in case-insensitive filesystems to prevent "already exists" errors):

  `rename 'y/A-Z/a-z/' {{*}}`

- Replace whitespace with underscores:

  `rename 's/\s+/_/g' {{*}}`

file

Determine file type.

More information: https://manned.org/file.

- G ive a description of the type of the specified file. Works fine for files with no file extension:

  ```
  file {{filenam e}}
  ```

- Look inside a z ipped file and determine the file type(s) inside:

  ```
  file - z {{foo. zip}}
  ```

- Allow file to work with special or device files:

  ```
  file - s {{filenam e}}
  ```

- Don' t stop at first file type match; keep going until the end of the file:

  ```
  file - k  {{filenam e}}
  ```

- Determine the mime encoding type of a file:

  ```
  file - i {{filenam e}}
  ```

filefrag

Report how badly fragmented a particular file might be.

More information: https://manned.org/filefrag.

- Display a report for a specific file:

```
filefrag {{path/to/file}}
```

- Display a report for space-separated list of files:

```
filefrag {{path/to/file1}} {{path/to/file2}}
```

- Display a report using a 1024 byte blocksize:

```
filefrag -b {{path/to/file}}
```

- Sync the file before requesting the mapping:

```
filefrag -s {{path/to/files}}
```

- Display mapping of extended attributes:

```
filefrag -x {{path/to/files}}
```

- Display a report with verbose information:

```
filefrag -v {{path/to/files}}
```

finch

Console-based modular messaging client.

More information: https://developer.pidgin.im/wiki/U sing% 2 0 F inch

- Launch finch:

  ```
  finch
  ```

- Q uit:

  ```
  alt + q or ctrl + c
  ```

- S how actions menu:

  ```
  alt + a
  ```

- J ump to n-th window:

  ```
  alt + {{nu m b er_k ey}}
  ```

- Close current window:

  ```
  alt + c
  ```

- S tart moving a window, use arrow keys to move, press escape when done:

  ```
  alt + m
  ```

- S tart resiz ing a window, use arrow keys to resiz e, press escape when done:

  ```
  alt + r
  ```

findfs

Finds a filesystem by label or UUID.

More information: https://mirrors.edge.kernel.org/pub/linux/utils/util-linux.

- Search block devices by filesystem label:

  ```
  findfs LABEL={{label}}
  ```

- Search by filesystem UUID:

  ```
  findfs UUID={{uuid}}
  ```

- Search by partition label (GPT or MAC partition table):

  ```
  findfs PARTLABEL={{partition_label}}
  ```

- Search by partition UUID (GPT partition table only):

  ```
  findfs PARTUUID={{partition_uuid}}
  ```

findmnt

F ind your filesystem.

More information: https://manned.org/findmnt.

- List all mounted filesystems:

  ```
  findm nt
  ```

- S earch for a device:

  ```
  findm nt {{/dev /sdb 1 }}
  ```

- S earch for a mountpoint:

  ```
  findm nt {{/}}
  ```

- F ind filesystems in specific type:

  ```
  findm nt - t {{ext4 }}
  ```

- F ind filesystems with specific label:

  ```
  findm nt L A B E L = {{B ig S torag e}}
  ```

firej ail

S ecurely sandboxes processes to containers using built-in Linux capabilities.

More information: https://manned.org/firej ail

- Integrate firej ail with your desktop environment:

```
su do firecfg
```

- O pen a restricted Moz illa F irefox:

```
firej ail {{firefox}}
```

- S tart a restricted Apache server on a known interface and address:

```
firej ail - - net= {{eth0 }} - - ip= {{1 9 2 . 1 6 8 . 1 . 2 4 4 }} {{/etc/in
apache2 }} {{start}}
```

- List running sandboxes:

```
firej ail - - list
```

- List network activity from running sandboxes:

```
firej ail - - netstats
```

- S hutdown a running sandbox:

```
firej ail - - shu tdow n= {{7 7 7 7 }}
```

firewall-cmd

The firewalld command-line client.

More information: https://firewalld.org/documentation/man-pages/firewall-cmd.

- View the available firewall zones:

  ```
  firewall-cmd --get-active-zones
  ```

- View the rules which are currently applied:

  ```
  firewall-cmd --list-all
  ```

- Permanently move the interface into the block zone, effectively blocking all communication:

  ```
  firewall-cmd --permanent --zone={{block}} --change-interface={{enp1s0}}
  ```

- Permanently open the port for a service in the specified zone (like port 443 when in the `public` zone):

  ```
  firewall-cmd --permanent --zone={{public}} --add-service={{https}}
  ```

- Permanently close the port for a service in the specified zone (like port 80 when in the `public` zone):

  ```
  firewall-cmd --permanent --zone={{public}} --remove-service={{http}}
  ```

- Permanently open two arbitrary ports in the specified zone:

  ```
  firewall-cmd --permanent --zone={{public}} --add-port={{25565/tcp}} --add-port={{19132/udp}}
  ```

- Reload firewalld to force rule changes to take effect:

  ```
  firewall-cmd --reload
  ```

flameshot

Screenshot utility with a gui interface.

Supports basic image editing, such as text, shapes, colors, and imgur.

More information: https://flameshot.js.org

- Launch flameshot in gui mode:

  ```
  flameshot launcher
  ```

- Take a screenshot by clicking and dragging:

  ```
  flameshot gui
  ```

- Take a full screen screenshot:

  ```
  flameshot full
  ```

- Set the save path to write screenshots to:

  ```
  flameshot full --path {{path/to/directory}}
  ```

- Delay the screenshot for N milliseconds and output to clipboard:

  ```
  flameshot full --delay {{2000}} --clipboard
  ```

flash

Flashcards in the terminal.

More information: https://github.com/tallguyjenks/flash.sh

- Open a menu of available flashcard decks for selection:

  `flash`

- Display the program version:

  `flash -v`

- Display information about the flashcard system:

  `flash -i`

- Display a list of available commands:

  `flash -h`

- Change the previewer from default `bat` to `cat`:

  `flash -p {{cat}}`

flashrom

Read, write, verify and erase flash chips.

More information: https://manned.org/flashrom

- Probe the chip, ensuring the wiring is correct:

```
flashrom --programmer {{programmer}}
```

- Read flash and save it to a file:

```
flashrom -p {{programmer}} --read {{path/to/file}}
```

- Write a file to the flash:

```
flashrom -p {{programmer}} --write {{path/to/file}}
```

- Verify the flash against a file:

```
flashrom -p {{programmer}} --verify {{path/to/file}}
```

- Probe the chip using RaspberryPi:

```
flashrom -p {{linux_spi:dev=/dev/spidev0.0}}
```

flatpak-builder

Help build application dependencies.

More information: https://docs.flatpak.org/en/latest/flatpak-builder-command-reference.html.

- Build a Flatpak and export it to a new repository:

```
flatpak-builder {{path/to/build_directory}} {{path/to/manifest}}
```

- Build a Flatpak and export it to the specified repository:

```
flatpak-builder --repo={{repository_name}} {{path/to/build_directory}} {{path/to/manifest}}
```

- Build a Flatpak and install it locally:

```
flatpak-builder --install {{path/to/build_directory}} {{path/to/manifest}}
```

- Build and sign a Flatpak and export it to the specified repository:

```
flatpak-builder --gpg-sign={{key_id}} --repo={{repository_name}} {{path/to/manifest}}
```

- Run a shell inside of an application sandbox without installing it:

```
flatpak-builder --run {{path/to/build_directory}} {{path/to/manifest}} {{sh}}
```

flatpak

Build, install and run flatpak applications and runtimes.

More information: https://docs.flatpak.org/en/latest/flatpak-command-reference.html#flatpak

- Run an installed application:

```
flatpak run {{name}}
```

- Install an application from a remote source:

```
flatpak install {{remote}} {{name}}
```

- List all installed applications and runtimes:

```
flatpak list
```

- Update all installed applications and runtimes:

```
flatpak update
```

- Add a remote source:

```
flatpak remote-add --if-not-exists {{remote_name}} {{remote_url}}
```

- List all configured remote sources:

```
flatpak remote-list
```

- Remove an installed application:

```
flatpak remove {{name}}
```

- Show information about an installed application:

```
flatpak info {{name}}
```

foreman

Manage Procfile-based applications.

More information: https://manned.org/foreman.

- Start an application with the Procfile in the current directory:

  ```
  foreman start
  ```

- Start an application with a specified Procfile:

  ```
  foreman start -f {{Procfile}}
  ```

- Start a specific application:

  ```
  foreman start {{process}}
  ```

- Validate Procfile format:

  ```
  foreman check
  ```

- Run one-off commands with the process's environment:

  ```
  foreman run {{command}}
  ```

- Start all processes except the one named "worker":

  ```
  foreman start -m all=1,{{worker}}=0
  ```

free

Display amount of free and used memory in the system.

More information: https://manned.org/free.

- Display system memory:

```
free
```

- Display memory in Bytes/KB/MB/GB:

```
free -{{b|k|m|g}}
```

- Display memory in human readable units:

```
free -h
```

- Refresh the output every 2 seconds:

```
free -s {{2}}
```

fsck

Check the integrity of a filesystem or repair it. T he filesystem should be unmounted at the time the command is run.

More information: https://manned.org/fsck.

- Check filesystem `/dev /sdX`, reporting any damaged blocks:

  ```
  fsck {{/dev /sdX }}
  ```

- Check filesystem `/dev /sdX`, reporting any damaged blocks and interactively letting the user choose to repair each one:

  ```
  fsck - r {{/dev /sdX }}
  ```

- Check filesystem `/dev /sdX`, reporting any damaged blocks and automatically repairing them:

  ```
  fsck - a {{/dev /sdX }}
  ```

fstrim

Discard unused blocks on a mounted filesystem.

Only supported by flash memory devices such as SSDs and microSD cards.

More information: https://manned.org/fstrim.

- Trim unused blocks on all mounted partitions that support it:

```
sudo fstrim --all
```

- Trim unused blocks on a specified partition:

```
sudo fstrim {{/}}
```

- Display statistics after trimming:

```
sudo fstrim --verbose {{/}}
```

fuser

Display process IDs currently using files or sockets.

More information: https://manned.org/fuser.

- Find which processes are accessing a file or directory:

```
fuser {{path/to/file_or_directory}}
```

- Show more fields (`USER`, `PID`, `ACCESS` and `COMMAND`):

```
fuser --verbose {{path/to/file_or_directory}}
```

- Identify processes using a TCP socket:

```
fuser --namespace tcp {{port}}
```

- Kill all processes accessing a file or directory (sends the `SIGKILL` signal):

```
fuser --kill {{path/to/file_or_directory}}
```

- Find which processes are accessing the filesystem containing a specific file or directory:

```
fuser --mount {{path/to/file_or_directory}}
```

gcov

Code coverage analysis and profiling tool that discovers untested parts of a program.

Also displays a copy of source code annotated with execution freq uencies of code segments.

More information: https://gcc.gnu.org/onlinedocs/gcc/Invoking-G cov.html

- G enerate a coverage report named `file. cpp. g cov`

  ```
  g cov {{path/to/file. cpp}}
  ```

- Write individual execution counts for every basic block:

  ```
  g cov - - all- b lock s {{path/to/file. cpp}}
  ```

- Write branch freq uencies to the output file and print summary information to stdout as a percentage:

  ```
  g cov - - b ranch- prob ab ilities {{path/to/file. cpp}}
  ```

- Write branch freq uencies as the number of branches taken, rather than the percentage:

  ```
  g cov - - b ranch- cou nts {{path/to/file. cpp}}
  ```

- Do not create a `g cov` output file:

  ```
  g cov - - no- ou tpu t {{path/to/file. cpp}}
  ```

- Write file level as well as function level summaries:

  ```
  g cov - - fu nction- su m m aries {{path/to/file. cpp}}
  ```

gdebi

Simple tool to install `.deb` files.

More information: https://www.commandlinux.com/man-page/man1/gdebi.1.html

- Install local `.deb` packages resolving and installing its dependencies:

  ```
  gdebi {{path/to/package.deb}}
  ```

- Display the program version:

  ```
  gdebi --version
  ```

- Do not show progress information:

  ```
  gdebi {{path/to/package.deb}} --quiet
  ```

- Set an APT configuration option:

  ```
  gdebi {{path/to/package.deb}} --option={{APT_OPTS}}
  ```

- Use alternative root dir:

  ```
  gdebi {{path/to/package.deb}} --root={{path/to/root_dir}}
  ```

gedit

Text editor of the GNOME Desktop project.

More information: https://help.gnome.org/users/gedit/stable/.

- Open a text file:

  ```
  gedit {{path/to/file}}
  ```

- Open multiple text files:

  ```
  gedit {{file1 file2 ...}}
  ```

- Open a text file with a specific encoding:

  ```
  gedit --encoding={{UTF-8}} {{path/to/file}}
  ```

- Display a list of supported encodings:

  ```
  gedit --list-encodings
  ```

genfstab

Arch Linux install script to generate output suitable for addition to an fstab file.

More information: https://man.archlinux.org/man/extra/arch-install-scripts/genfstab.8.

- Display an fstab compatible output based on a volume label:

```
g enfstab  - L  {{path/to/m ou nt_point}}
```

- Display an fstab compatible output based on a volume U U ID:

```
g enfstab  - U  {{path/to/m ou nt_point}}
```

- A usual way to generate an fstab file, req uires root permissions:

```
g enfstab  - U  {{/m nt}} > >  {{/m nt/etc/fstab }}
```

- Append a volume into an fstab file to mount it automatically:

```
g enfstab  - U  {{path/to/m ou nt_point}} |  su do  tee  - a  /etc/fstab
```

genid

Generate IDs, such as snowflakes, UUIDs, and a new GAID.

More information: https://github.com/bleonard252/genid

- Generate a UUIDv4:

  ```
  genid uuid
  ```

- Generate a UUIDv5 using a namespace UUID and a specific name:

  ```
  genid uuidv5 {{{ce59 8 8 faa-8 dd0-49 ee-8 52 5-9 e24 fff7 1 dca}}} {{name}}
  ```

- Generate a Discord Snowflake, without a trailing newline (useful in shell scripts):

  ```
  genid --script snowflake
  ```

- Generate a Generic Anonymous ID with a specific "real ID":

  ```
  genid gaid {{real_id}}
  ```

- Generate a Snowflake with the epoch set to a specific date:

  ```
  genid snowflake --epoch={{unix_epoch_time}}
  ```

genie

Set up and use a "bottle" namespace to run systemd under WSL (Windows Subsystem for Linux).

To run these from Windows rather than an already-running distribution, precede them with `wsl`.

More information: https://github.com/arkane-systems/genie.

- Initialize the bottle (run once, at start):

```
genie -i
```

- Run a login shell inside the bottle:

```
genie -s
```

- Run a specified command inside the bottle:

```
genie -c {{command}}
```

genkernel

G entoo Linux utility to compile and install kernels.

More information: https://wiki.gentoo.org/wiki/G enkernel

- Automatically compile and install a generic kernel:

  ```
  su do g enk ernel all
  ```

- B uild and install the bz Image| initramfs| kernel| ramdisk only:

  ```
  su do g enk ernel {{b zI m ag e| initram fs| k ernel| ram disk }}
  ```

- Apply changes to the kernel configuration before compiling and installing:

  ```
  su do g enk ernel - - m enu config all
  ```

- G enerate a kernel with a custom name:

  ```
  su do g enk ernel - - k ernnam e= {{cu stom _nam e}} all
  ```

- U se a kernel source outside of the default directory `/u sr/src/linu x`:

  ```
  su do g enk ernel - - k erneldir= {{path/to/directory}} all
  ```

getent

Get entries from Name Service Switch libraries.

More information: https://manned.org/getent.

- Get list of all groups:

  ```
  getent group
  ```

- See the members of a group:

  ```
  getent group {{group_name}}
  ```

- Get list of all services:

  ```
  getent services
  ```

- Find a username by UID:

  ```
  getent passwd 1000
  ```

- Perform a reverse DNS lookup:

  ```
  getent hosts {{host}}
  ```

getfacl

G et file access control lists.

More information: https://manned.org/getfacl.

- Display the file access control list:

```
g etfacl {{path/to/file_or_directory}}
```

- Display the file access control list with numeric user and group IDs:

```
g etfacl - n {{path/to/file_or_directory}}
```

- Display the file access control list with tabular output format:

```
g etfacl - t {{path/to/file_or_directory}}
```

gnome-extensions

Manage gnome extensions from the terminal.

More information: https://wiki.gnome.org/Projects/GnomeShell/Extensions

- Display the version:

  ```
  gnome-extensions version
  ```

- List all the installed extensions:

  ```
  gnome-extensions list
  ```

- Display information about a specific extension:

  ```
  gnome-extensions info "{{extension_id}}"
  ```

- Display help for a subcommand (like `list`):

  ```
  gnome-extensions help {{subcommand}}
  ```

- Enable a specific extension:

  ```
  gnome-extensions enable "{{extension_id}}"
  ```

- Disable a specific extension:

  ```
  gnome-extension disable "{{extension_id}}"
  ```

- Uninstall a specific extension:

  ```
  gnome-extension uninstall "{{extension_id}}"
  ```

gnome-terminal

The GNOME Terminal emulator.

More information: https://help.gnome.org/users/gnome-terminal/stable/.

- Open a new GNOME terminal window:

  ```
  gnome-terminal
  ```

- Run a specific command in a new terminal window:

  ```
  gnome-terminal --  {{command}}
  ```

- Open a new tab in the last opened window instead:

  ```
  gnome-terminal --tab
  ```

- Set the title of the new tab:

  ```
  gnome-terminal --tab  --title "{{title}}"
  ```

google-chrome

The web browser from Google.

More information: https://chrome.google.com.

- Run with a custom profile directory:

```
google-chrome --user-data-dir={{path/to/directory}}
```

- Run without CORS validation, useful to test an API:

```
google-chrome --user-data-dir={{path/to/directory}} --disable-web-security
```

gpasswd

Administer `/etc/group` and `/etc/gshadow`.

More information: https://manned.org/gpasswd.

- Define group administrators:

```
sudo gpasswd -A {{user1,user2}} {{group}}
```

- Set the list of group members:

```
sudo gpasswd -M {{user1,user2}} {{group}}
```

- Create a password for the named group:

```
gpasswd {{group}}
```

- Add a user to the named group:

```
gpasswd -a {{user}} {{group}}
```

- Remove a user from the named group:

```
gpasswd -d {{user}} {{group}}
```

groupadd

Add user groups to the system.

More information: https://manned.org/groupadd.

- Create a new Linux group:

  ```
  g rou padd {{g rou p_nam e}}
  ```

- Create new group with a specific groupid:

  ```
  g rou padd {{g rou p_nam e}} - g  {{g rou p_id}}
  ```

groupdel

Delete existing user groups from the system.

More information: https://manned.org/groupdel.

- Delete an existing group:

```
g rou pdel {{g rou p_nam e}}
```

groupmod

Modify existing user groups in the system.

More information: https://manned.org/groupmod.

- Change the group name:

```
g rou pm od - n {{new _g rou p_nam e}} {{old_g rou p_nam e}}
```

- Change the group id:

```
g rou pm od - g  {{new _g rou p_id}} {{old_g rou p_nam e}}
```

grub-install

Install G R U B to a device.

More information: https://www.gnu.org/software/grub/manual/grub/html_node/Installing-G R U B -using-grub_0 0 2 dinstall.html

- Install G R U B on a B IO S system:

```
g ru b - install - - targ et= {{i3 8 6 - pc}} {{path/to/dev ice}}
```

- Install G R U B on an U E F I system:

```
g ru b - install - - targ et= {{x8 6 _6 4 - efi}} - - efi- directory= {{path
to/efi_directory}} - - b ootloader- id= {{G R U B }}
```

- Install G R U B pre-loading specific modules:

```
g ru b - install - - targ et= {{x8 6 _6 4 - efi}} - - efi- directory= {{path
to/efi_directory}} - - m odu les= " {{part_g pt part_m sdos}}"
```

grub-mkconfig

Generate a GRUB configuration file.

More information: https://www.gnu.org/software/grub/manual/grub/html_node/Invoking-grub_002dmkconfig.html

- Do a dry run and print the configuration to stdout:

```
sudo grub-mkconfig
```

- Generate the configuration file:

```
sudo grub-mkconfig --output={{/boot/grub/grub.cfg}}
```

- Print the help page:

```
grub-mkconfig --help
```

gs

GhostScript is a PDF and PostScript interpreter.

More information: https://manned.org/gs.

- To view a file:

```
gs -dQUIET -dBATCH {{file.pdf}}
```

- Reduce PDF file size to 150 dpi images for reading on a e-book device:

```
gs -dNOPAUSE -dQUIET -dBATCH -sDEVICE=pdfwrite -
dPDFSETTINGS=/ebook -sOutputFile={{output.pdf}} {{input
```

- Convert PDF file (pages 1 through 3) to an image with 150 dpi resolution:

```
gs -dQUIET -dBATCH -dNOPAUSE -sDEVICE=jpeg -r150
dFirstPage={{1}} -dLastPage={{3}} -
sOutputFile={{output_%d.jpg}} {{input.pdf}}
```

- Extract pages from a PDF file:

```
gs -dQUIET -dBATCH -dNOPAUSE -sDEVICE=pdfwrite -
sOutputFile={{output.pdf}} {{input.pdf}}
```

- Merge PDF files:

```
gs -dQUIET -dBATCH -dNOPAUSE -sDEVICE=pdfwrite -
sOutputFile={{output.pdf}} {{input1.pdf}} {{input2.pdf}}
```

- Convert from PostScript file to PDF file:

```
gs -dQUIET -dBATCH -dNOPAUSE -sDEVICE=pdfwrite -
sOutputFile={{output.pdf}} {{input.ps}}
```

gsettings

Q uery and modify dconf settings with schema validation.

More information: https://access.redhat.com/documentation/en-us/red_hat_enterprise_linux/8/html/using_the_desktop_environment_in_rhel_8/configuring-gnome-at-low-level_using-the-desktop-environment-in-rhel-8#using-gsettings-command_configuring-gnome-at-low-level

- S et the value of a key. F ails if the key doesn' t exist or the value is out of range:

```
g setting s set {{org . exam ple. schem a}} {{exam ple- k ey}} {{v alu e}}
```

- P rint the value of a key or the schema-provided default if the key has not been set in `dconf`:

```
g seting s g et {{org . exam ple. schem a}} {{exam ple- k ey}}
```

- U nset a key, so that its schema default value will be used:

```
g setting s reset {{org . exam ple. schem a}} {{exam ple- k ey}}
```

- Display all (non-relocatable) schemas, keys, and values:

```
g setting s list- recu rsiv ely
```

- Display all keys and values (default if not set) from one schema:

```
g setting s list- recu rsiv ely {{org . exam ple. schem a}}
```

- Display schema-allowed values for a key (helpful with enum keys):

```
g setting s rang e {{org . exam ple. schem a}} {{exam ple- k ey}}
```

- Display the human-readable description of a key:

```
g setting s describ e {{org . exam ple. schem a}} {{exam ple- k ey}}
```

guake

A drop-down terminal for G N O ME .

More information: http://guake-proj ect.org/

- T oggle G uake visibility:

  `F 1 2`

- T oggle fullscreen mode:

  `F 1 1`

- O pen a new tab:

  `C trl+ S hift+ T`

- Close the terminal:

  `S u per+ X`

- G o to the previous tab:

  `C trl+ P ag eU p`

- S earch the selected text in the browser:

  `S hift+ C trl+ L`

guix package

Install, upgrade and remove G uix packages, or rollback to previous configurations.

More information: https://guix.gnu.org/manual/html_node/Invoking-guix-package.html.

- Install a new package:

```
g u ix pack ag e - i {{pack ag e_nam e}}
```

- R emove a package:

```
g u ix pack ag e - r {{pack ag e_nam e}}
```

- S earch the package database for a regular expression:

```
g u ix pack ag e - s " {{search_pattern}}"
```

- List installed packages:

```
g u ix pack ag e - I
```

- List generations:

```
g u ix pack ag e - l
```

- R oll back to the previous generation:

```
g u ix pack ag e - - roll- b ack
```

halt

Halt the system.

More information: https://www.man7.org/linux/man-pages/man8/halt.8.html

- Halt the system:

  ```
  halt
  ```

- Power off the system (same as `poweroff`):

  ```
  halt --poweroff
  ```

- Reboot the system (same as `reboot`):

  ```
  halt --reboot
  ```

- Halt immediately without contacting the system manager:

  ```
  halt --force --force
  ```

- Write the wtmp shutdown entry without halting the system:

  ```
  halt --wtmp-only
  ```

hardinfo

Show hardware information in GUI window.

- Start hardinfo:

  ```
  hardinfo
  ```

- Print report to standard output:

  ```
  hardinfo -r
  ```

- Save report to HTML file:

  ```
  hardinfo -r -f html > hardinfo.html
  ```

hashcat

Fast and advanced password recovery tool.

More information: https://manned.org/hashcat.

- Perform a brute-force attack (mode 3) with the default hashcat mask:

```
hashcat --hash-type {{hash_type_id}} --attack-mode {{3}} {{hash_value}}
```

- Perform a brute-force attack (mode 3) with a known pattern of 4 digits:

```
hashcat --hash-type {{hash_type_id}} --attack-mode {{3}} {{hash_value}} "{{?d?d?d?d}}"
```

- Perform a brute-force attack (mode 3) using at most 8 of all printable ASCII characters:

```
hashcat --hash-type {{hash_type_id}} --attack-mode {{3}} --increment {{hash_value}} "{{?a?a?a?a?a?a?a?a}}"
```

- Perform a dictionary attack (mode 0) using the RockYou wordlist of a Kali Linux box:

```
hashcat --hash-type {{hash_type_id}} --attack-mode {{0}} {{hash_value}} {{/usr/share/wordlists/rockyou.txt}}
```

- Perform a rule based dictionary attack (mode 0) using the RockYou wordlist mutated with common password variations:

```
hashcat --hash-type {{hash_type_id}} --attack-mode {{0}} --rules-file {{/usr/share/hashcat/rules/best64.rule}} {{hash_value}} {{/usr/share/wordlists/rockyou.txt}}
```

- Perform a combination attack (mode 1) using the concatenation of words from two different custom dictionaries:

```
hashcat --hash-type {{hash_type_id}} --attack-mode {{1}} {{hash_value}} {{/path/to/dictionary1.txt}} {{/path/to/dictionary2.txt}}
```

- Show result of an already cracked hash:

```
hashcat --show {{hash_value}}
```

hdparm

Get and set SATA and IDE hard drive parameters.

- Request the identification info of a given device:

```
sudo hdparm -I /dev/{{device}}
```

- Get the Advanced Power Management level:

```
sudo hdparm -B /dev/{{device}}
```

- Set the Advanced Power Management value (values 1-127 permit spin-down, and values 128-254 do not):

```
sudo hdparm -B {{1}} /dev/{{device}}
```

- Display the device's current power mode status:

```
sudo hdparm -C /dev/{{device}}
```

- Force a drive to immediately enter standby mode (usually causes a drive to spin down):

```
sudo hdparm -y /dev/{{device}}
```

- Put the drive into idle (low-power) mode, also setting its standby timeout:

```
sudo hdparm -S {{standby_timeout}} {{device}}
```

hello

Print "Hello, world!", "hello, world" or a customizable text.

More information: https://www.gnu.org/software/hello/.

- Print "Hello, world!":

  ```
  hello
  ```

- Print "hello, world", the traditional type:

  ```
  hello --traditional
  ```

- Print a text message:

  ```
  hello --greeting="{{greeting_text}}"
  ```

hexdump

An ASCII, decimal, hexadecimal, octal dump.

- Print the hexadecimal representation of a file:

  ```
  hexdump {{file}}
  ```

- Display the input offset in hexadecimal and its ASCII representation in two columns:

  ```
  hexdump -C {{file}}
  ```

- Display the hexadecimal representation of a file, but interpret only n bytes of the input:

  ```
  hexdump -C -n{{number_of_bytes}} {{file}}
  ```

hlint

T ool for suggesting improvements to H askell code.

More information: http://hackage.haskell.org/package/hlint.

- Display suggestions for a given file:

  ```
  hlint {{path/to/file}} options
  ```

- Check all H askell files and generate a report:

  ```
  hlint {{path/to/directory}} - - report
  ```

- Automatically apply most suggestions:

  ```
  hlint {{path/to/file}} - - refactor
  ```

- Display additional options:

  ```
  hlint {{path/to/file}} - - refactor- options
  ```

- G enerate a settings file ignoring all outstanding hints:

  ```
  hlint {{path/to/file}} - - defau lt >  {{. hlint. yam l}}
  ```

homectl

Create, remove, change or inspect home directories using the systemd-homed service.

More information: https://man.archlinux.org/man/homectl.1.

- List user accounts and their associated home directories:

```
homectl list
```

- Create a user account and their associated home directory:

```
sudo homectl create {{username}}
```

- Remove a specific user and the associated home directory:

```
sudo homectl remove {{username}}
```

- Change the password for a specific user:

```
sudo homectl passwd {{username}}
```

- Run a shell or a command with access to a specific home directory:

```
sudo homectl with {{username}} -- {{command}} {{command_arguments}}
```

- Lock or unlock a specific home directory:

```
sudo homectl {{lock|unlock}} {{username}}
```

- Change the disk space assigned to a specific home directory to 100 GiB:

```
sudo homectl resize {{username}} {{100G}}
```

- Display help:

```
homectl --help
```

homeshick

Synchronize Git dotfiles.

More information: https://github.com/andsens/homeshick/wiki.

- Create a new castle:

  `homeshick generate {{castle_name}}`

- Add a file to your castle:

  `homeshick track {{castle_name}} {{path/to/file}}`

- Go to a castle:

  `homeshick cd {{castle_name}}`

- Clone a castle:

  `homeshick clone {{github_username}}/{{repository_name}}`

- Symlink all files from a castle:

  `homeshick link {{castle_name}}`

hostname

S how or set the system' s host name.

- S how current host name:

  ```
  hostnam e
  ```

- S how the network address of the host name:

  ```
  hostnam e - i
  ```

- S how all network addresses of the host:

  ```
  hostnam e - I
  ```

- S how the F Q DN (F ully Q ualified Domain N ame):

  ```
  hostnam e - - fq dn
  ```

- S et current host name:

  ```
  hostnam e {{new _hostnam e}}
  ```

hostnamectl

Get or set the hostname of the computer.

- Get the hostname of the computer:

  ```
  hostnamectl
  ```

- Set the hostname of the computer:

  ```
  sudo hostnamectl set-hostname "{{hostname}}"
  ```

- Set a pretty hostname for the computer:

  ```
  sudo hostnamectl set-hostname --static
  "{{hostname.example.com}}" && sudo hostnamectl set-hostname
  --pretty "{{hostname}}"
  ```

- Reset hostname to its default value:

  ```
  sudo hostnamectl set-hostname --pretty ""
  ```

htpdate

Synchronize local date and time via HTTP headers from web servers.

More information: http://www.vervest.org/htp/.

- Synchronize date and time:

  ```
  sudo htpdate {{host}}
  ```

- Perform simulation of synchronization, without any action:

  ```
  htpdate -q {{host}}
  ```

- Compensate the systematisch clock drift:

  ```
  sudo htpdate -x {{host}}
  ```

- Set time immediate after the synchronization:

  ```
  sudo htpdate -s {{host}}
  ```

http-prompt

An interactive command-line HTTP client featuring autocomplete and syntax highlighting.

- Launch a session targeting the default URL of http://localhost:8000 or the previous session:

  ```
  http-prompt
  ```

- Launch a session with a given URL:

  ```
  http-prompt {{http://example.com}}
  ```

- Launch a session with some initial options:

  ```
  http-prompt {{localhost:8000/api}} --auth
  {{username:password}}
  ```

http_load

A HTTP benchmarking tool.

Runs multiple HTTP fetches in parallel to test the throughput of a web server.

More information: http://www.acme.com/software/http_load/

- Emulate 20 requests based on a given URL list file per second for 60 seconds:

  ```
  http_load -rate {{20}} -seconds {{60}} {{path/to/urls.txt}}
  ```

- Emulate 5 concurrent requests based on a given URL list file for 60 seconds:

  ```
  http_load -parallel {{5}} -seconds {{60}} {{path/to/
  urls.txt}}
  ```

- Emulate 1000 requests at 20 requests per second, based on a given URL list file:

  ```
  http_load -rate {{20}} -fetches {{1000}} {{path/to/urls.txt}}
  ```

- Emulate 1000 requests at 5 concurrent requests at a time, based on a given URL list file:

  ```
  http_load -parallel {{5}} -fetches {{1000}} {{path/to/
  urls.txt}}
  ```

httpie

A user friendly command-line HTTP tool.

- Send a GET request (default method with no request data):

  ```
  http {{https://example.com}}
  ```

- Send a POST request (default method with request data):

  ```
  http {{https://example.com}} {{hello=World}}
  ```

- Send a POST request with redirected input:

  ```
  http {{https://example.com}} < {{file.json}}
  ```

- Send a PUT request with a given json body:

  ```
  http PUT {{https://example.com/todos/7}} {{hello=world}}
  ```

- Send a DELETE request with a given request header:

  ```
  http DELETE {{https://example.com/todos/7}} {{API-Key:foo}}
  ```

- Show the whole HTTP exchange (both request and response):

  ```
  http -v {{https://example.com}}
  ```

- Download a file:

  ```
  http --download {{https://example.com}}
  ```

hwclock

Used for reading or changing the hardware clock. Usually requires root.

- Display the current time as reported by the hardware clock:

  ```
  hwclock
  ```

- Write the current software clock time to the hardware clock (sometimes used during system setup):

  ```
  hwclock --systohc
  ```

- Write the current hardware clock time to the software clock:

  ```
  hwclock --hctosys
  ```

i3

A dynamic tiling window manager.

More information: https://i3wm.org/docs/userguide.html

- Start i3 (Note that a pre-existing window manager must not be open when this command is run.):

```
i3
```

- Open a new terminal window:

```
Super + Return
```

- Create a new workspace:

```
Super + Shift + {{number}}
```

- Switch to workspace {{number}}:

```
Super + {{number}}
```

- Open new window horizontally:

```
Super + h
```

- Open new window vertically:

```
Super + v
```

- Open application (type out application name after executing command):

```
Super + D
```

i3 lock

Simple screen locker built for the i3 window manager.

More information: https://i3wm.org/i3lock

- Lock screen with a simple color background (rrggbb format):

```
i3lock -c {{0000ff}}
```

- Lock screen to a PNG background:

```
i3lock -i {{path/to/picture.png}}
```

- Disable the unlock indicator (removes feedback on keypress):

```
i3lock -u
```

- Display mouse pointer instead of hiding it ('default' for default pointer, 'win' for a MS Windows pointer):

```
i3lock -p {{default|win}}
```

- Lock screen to a PNG background displayed in multiple monitors, with enabled mouse pointer:

```
i3lock -i {{path/to/picture.png}} -p {{default|win}} -t
```

i7 z

An Intel CP U (only i3 , i5 and i7) realtime reporting tool.

- S tart i7 z (needs to be run in super user mode):

  ```
  su do i7 z
  ```

ifdown

Disable network interfaces.

More information: https://manned.org/ifdown.

- Disable interface eth0 :

```
ifdow n {{eth0 }}
```

- Disable all interfaces which are enabled:

```
ifdow n - a
```

iftop

S how bandwidth usage on an interface by host.

More information: https://manned.org/iftop.

- S how the bandwidth usage:

  ```
  su do iftop
  ```

- S how the bandwidth usage of a given interface:

  ```
  su do iftop - i {{interface}}
  ```

- S how the bandwidth usage with port information:

  ```
  su do iftop - P
  ```

- Do not show bar graphs of traffic:

  ```
  su do iftop - b
  ```

- Do not look up hostnames:

  ```
  su do iftop - n
  ```

- G et help about interactive commands:

  ```
  ?
  ```

ifup

Tool used to enable network interfaces.

More information: https://manpages.debian.org/latest/ifupdown/ifup.8.html.

- Enable interface eth0:

  ```
  ifup {{eth0}}
  ```

- Enable all the interfaces defined with "auto" in `/etc/network/interfaces`:

  ```
  ifup -a
  ```

imgp

Command-line image resizer and rotator for JPEG and PNG images.

- Convert single images and/or whole directories containing valid image formats:

  ```
  imgp -x {{1366x1000}} {{path/to/directory}} {{path/to/file}}
  ```

- Scale an image by 75% and overwrite the source image to a target resolution:

  ```
  imgp -x {{75}} -w {{path/to/file}}
  ```

- Rotate an image clockwise by 90 degrees:

  ```
  imgp -o {{90}} {{path/to/file}}
  ```

inotifywait

Waits for changes to one or more files.

- Run a command when a file changes:

```
while inotifywait {{path/to/file}}; do {{command}}; done
```

- Be quiet about watching for changes:

```
while inotifywait --quiet {{path/to/file}}; do {{command}}; done
```

- Watch a directory recursively for changes:

```
while inotifywait --recursive {{path/to/directory}}; do {{command}}; done
```

- Exclude files matching a regular expression:

```
while inotifywait --recursive {{path/to/directory}} --exclude '{{regular_expression}}'; do {{command}}; done
```

- Wait at most 30 seconds:

```
while inotifywait --timeout {{30}} {{path/to/file}}; do {{command}}; done
```

- Only watch for file modification events:

```
while inotifywait --event {{modify}} {{path/to/file}}; do {{command}}; done
```

inxi

Print a summary of system information and resources for debugging purposes.

- Print a short summary of CPU, memory, hard drive and kernel information:

  ```
  inxi
  ```

- Print a full description of CPU, memory, disk, network and process information:

  ```
  inxi -Fz
  ```

- Print information about the distribution's repository:

  ```
  inxi -r
  ```

iostat

Report statistics for devices and partitions.

- Display a report of CPU and disk statistics since system startup:

  ```
  iostat
  ```

- Display a report of CPU and disk statistics with units converted to megabytes:

  ```
  iostat -m
  ```

- Display CPU statistics:

  ```
  iostat -c
  ```

- Display disk statistics with disk names (including LVM):

  ```
  iostat -N
  ```

- Display extended disk statistics with disk names for device "sda":

  ```
  iostat -xN {{sda}}
  ```

- Display incremental reports of CPU and disk statistics every 2 seconds:

  ```
  iostat {{2}}
  ```

ip address

IP Address management subcommand.

- List network interfaces and their associated IP addresses:

  ```
  ip address
  ```

- Filter to show only active network interfaces:

  ```
  ip address show up
  ```

- Display information about a specific network interface:

  ```
  ip address show dev {{eth0}}
  ```

- Add an IP address to a network interface:

  ```
  ip address add {{ip_address}} dev {{eth0}}
  ```

- Remove an IP address from a network interface:

  ```
  ip address delete {{ip_address}} dev {{eth0}}
  ```

- Delete all IP addresses in a given scope from a network interface:

  ```
  ip address flush dev {{eth0}} scope {{global|host|link}}
  ```

ip link

Manage network interfaces.

More information: https://man7.org/linux/man-pages/man8/ip-link.8.html

- Show information about all network interfaces:

```
ip link
```

- Show information about a specific network interface:

```
ip link show {{ethN}}
```

- Bring a network interface up or down:

```
ip link set {{ethN}} {{up|down}}
```

- Give a meaningful name to a network interface:

```
ip link set {{ethN}} alias "{{LAN Interface}}"
```

- Change the MAC address of a network interface:

```
ip link set {{ethN}} address {{ff:ff:ff:ff:ff:ff}}
```

- Change the MTU size for a network interface to use jumbo frames:

```
ip link set {{ethN}} mtu {{9000}}
```

ip-neighbour

Neighbour/ARP tables management IP subcommand.

More information: https://manned.org/ip-neighbour.8.

- Display the neighbour/ARP table entries:

```
ip neighbour
```

- Remove entries in the neighbour table on device `eth0`:

```
sudo ip neighbour flush dev {{eth0}}
```

- Perform a neighbour lookup and return a neighbour entry:

```
ip neighbour get {{lookup_ip}} dev {{eth0}}
```

- Add or delete an ARP entry for the neighbour IP address to `eth0`:

```
sudo ip neighbour {{add|del}} {{ip_address}} lladdr
{{mac_address}} dev {{eth0}} nud reachable
```

- Change or replace an ARP entry for the neighbour IP address to `eth0`:

```
sudo ip neighbour {{change|replace}} {{ip_address}} lladdr
{{new_mac_address}} dev {{eth0}}
```

ip route

IP R outing table management subcommand.

More information: https://manned.org/ip-route.

- Display the routing table:

```
ip rou te {{show | list}}
```

- Add a default route using gateway forwarding:

```
su do ip rou te add defau lt v ia {{g atew ay_ip}}
```

- Add a default route using `eth0`:

```
su do ip rou te add defau lt dev {{eth0 }}
```

- Add a static route:

```
su do ip rou te add {{destination_ip}} v ia {{g atew ay_ip}} dev
{{eth0 }}
```

- Delete a static route:

```
su do ip rou te del {{destination_ip}} dev {{eth0 }}
```

- Change or replace a static route:

```
su do ip rou te {{chang e| replace}} {{destination_ip}} v ia
{{g atew ay_ip}} dev {{eth0 }}
```

- S how which route will be used by the kernel to reach an IP address:

```
ip rou te g et {{destination_ip}}
```

ip

Show / manipulate routing, devices, policy routing and tunnels.

More information: https://www.man7.org/linux/man-pages/man8/ip.8.html

- List interfaces with detailed info:

  ```
  ip address
  ```

- List interfaces with brief network layer info:

  ```
  ip -brief address
  ```

- List interfaces with brief link layer info:

  ```
  ip -brief link
  ```

- Display the routing table:

  ```
  ip route
  ```

- Show neighbors (ARP table):

  ```
  ip neighbour
  ```

- Make an interface up/down:

  ```
  ip link set {{interface}} up/down
  ```

- Add/Delete an ip address to an interface:

  ```
  ip addr add/del {{ip}}/{{mask}} dev {{interface}}
  ```

- Add a default route:

  ```
  ip route add default via {{ip}} dev {{interface}}
  ```

ipcalc

Perform simple operations and calculations on IP addresses and networks.

- Show information about an address or network with a given subnet mask:

  ```
  ipcalc {{1.2.3.4}} {{255.255.255.0}}
  ```

- Show information about an address or network in CIDR notation:

  ```
  ipcalc {{1.2.3.4}}/{{24}}
  ```

- Show the broadcast address of an address or network:

  ```
  ipcalc -b {{1.2.3.4}}/{{30}}
  ```

- Show the network address of provided IP address and netmask:

  ```
  ipcalc -n {{1.2.3.4}}/{{24}}
  ```

- Display geographic information about a given IP address:

  ```
  ipcalc -g {{1.2.3.4}}
  ```

ipcmk

Create IPC (Inter-process Communication) resources.

- Create a shared memory segment:

  ```
  ipcmk --shmem {{segment_size_in_bytes}}
  ```

- Create a semaphore:

  ```
  ipcmk --semaphore {{element_size}}
  ```

- Create a message queue:

  ```
  ipcmk --queue
  ```

- Create a shared memory segment with specific permissions (default is 0644):

  ```
  ipcmk --shmem {{segment_size_in_bytes}} {{octal_permissions}}
  ```

ipcrm

Delete IPC (Inter-process Communication) resources.

- Delete a shared memory segment by ID:

```
ipcrm --shmem-id {{shmem_id}}
```

- Delete a shared memory segment by key:

```
ipcrm --shmem-key {{shmem_key}}
```

- Delete an IPC queue by ID:

```
ipcrm --queue-id {{ipc_queue_id}}
```

- Delete an IPC queue by key:

```
ipcrm --queue-key {{ipc_queue_key}}
```

- Delete a semaphore by ID:

```
ipcrm --semaphore-id {{semaphore_id}}
```

- Delete a semaphore by key:

```
ipcrm --semaphore-key {{semaphore_key}}
```

- Delete all IPC resources:

```
ipcrm --all
```

iptables

Program that allows configuration of tables, chains and rules provided by the Linux kernel firewall.

More information: https://www.netfilter.org/projects/iptables/

- View chains, rules, and packet/byte counters for the filter table:

```
sudo iptables -vnL
```

- Set chain policy rule:

```
sudo iptables -P {{chain}} {{rule}}
```

- Append rule to chain policy for IP:

```
sudo iptables -A {{chain}} -s {{ip}} -j {{rule}}
```

- Append rule to chain policy for IP considering protocol and port:

```
sudo iptables -A {{chain}} -s {{ip}} -p {{protocol}} --dport {{port}} -j {{rule}}
```

- Delete chain rule:

```
sudo iptables -D {{chain}} {{rule_line_number}}
```

- Save iptables configuration of a given table to a file:

```
sudo iptables-save -t {{tablename}} > {{path/to/iptables_file}}
```

- Restore iptables configuration from a file:

```
sudo iptables-restore < {{path/to/iptables_file}}
```

isoinfo

Utility programs for dumping and verifying ISO disk images.

- List all the files included in an ISO image:

  ```
  isoinfo -f -i {{path/to/image.iso}}
  ```

- Extract a specific file from an ISO image and send it out stdout:

  ```
  isoinfo -i {{path/to/image.iso}} -x {{/PATH/TO/FILE/INSIDE/ISO.EXT}}
  ```

- Show header information for an ISO disk image:

  ```
  isoinfo -d -i {{path/to/image.iso}}
  ```

isosiz e

Display the siz e of an IS O file.

More information: https://manned.org/isosiz e

- Display the siz e of an IS O file:

```
isosize {{path/to/file. iso}}
```

- Display the block count and block siz e of an IS O file:

```
isosize - - sectors {{path/to/file. iso}}
```

- Display the siz e of an IS O file divided by a given number (only usable when --sectors is not given):

```
isosize - - div isor= {{nu m b er}} {{path/to/file. iso}}
```

iw

Show and manipulate wireless devices.

- Scan for available wireless networks:

  ```
  iw dev {{wlp}} scan
  ```

- Join an open wireless network:

  ```
  iw dev {{wlp}} connect {{SSID}}
  ```

- Close the current connection:

  ```
  iw dev {{wlp}} disconnect
  ```

- Show information about the current connection:

  ```
  iw dev {{wlp}} link
  ```

iwconfig

Configure and show the parameters of a wireless network interface.

More information: https://manned.org/iwconfig.

- Show the parameters and statistics of all the interfaces:

```
iwconfig
```

- Show the parameters and statistics of the specified interface:

```
iwconfig {{interface}}
```

- Set the ESSID (network name) of the specified interface (e.g., eth0 or wlp2s0):

```
iwconfig {{interface}} {{new_network_name}}
```

- Set the operating mode of the specified interface:

```
iwconfig {{interface}} mode {{ad_hoc|Managed|Master|Repeater|Secondary|Monitor|Auto}}
```

iwctl

A command-line tool for controlling the iwd network supplicant.

More information: https://iwd.wiki.kernel.org/gettingstarted.

- Start the interactive mode, in this mode you can enter the commands directly, with autocompletion:

  ```
  iwctl
  ```

- Call general help:

  ```
  iwctl --help
  ```

- Display your wifi stations:

  ```
  iwctl station list
  ```

- Start looking for networks with a station:

  ```
  iwctl station {{station}} scan
  ```

- Display the networks found by a station:

  ```
  iwctl station {{station}} get-networks
  ```

- Connect to a network with a station, if credentials are needed they will be asked:

  ```
  iwctl station {{station}} connect {{network_name}}
  ```

jobs

BASH builtin for viewing information about processes spawned by the current shell.

- View jobs spawned by the current shell:

  ```
  jobs
  ```

- List jobs and their process ids:

  ```
  jobs -l
  ```

- Display information about jobs with changed status:

  ```
  jobs -n
  ```

- Display process id of process group leader:

  ```
  jobs -p
  ```

- Display running processes:

  ```
  jobs -r
  ```

- Display stopped processes:

  ```
  jobs -s
  ```

journalctl

Query the systemd journal.

- Show all messages from this [b]oot:

  ```
  journalctl -b
  ```

- Show all messages from last [b]oot:

  ```
  journalctl -b -1
  ```

- Show all messages with priority level 3 (errors) from this [b]oot:

  ```
  journalctl -b --priority={{3}}
  ```

- [f]ollow new messages (like `tail -f` for traditional syslog):

  ```
  journalctl -f
  ```

- Show all messages by a specific [u]nit:

  ```
  journalctl -u {{unit}}
  ```

- Filter messages within a time range (either timestamp or placeholders like "yesterday"):

  ```
  journalctl --since {{now|today|yesterday|tomorrow}} --until
  {{YYYY-MM-DD HH:MM:SS}}
  ```

- Show all messages by a specific process:

  ```
  journalctl _PID={{pid}}
  ```

- Show all messages by a specific executable:

  ```
  journalctl {{path/to/executable}}
  ```

jpegtran

Perform lossless transformation of JPEG files.

More information: https://manned.org/jpegtran

- Mirror an image horizontally or vertically:

```
jpegtran -flip {{horizontal|vertical}} {{path/to/image.jpg}} > {{path/to/output.jpg}}
```

- Rotate an image 90, 180 or 270 degrees clockwise:

```
jpegtran -rotate {{90|180|270}} {{path/to/image.jpg}} > {{path/to/output.jpg}}
```

- Transpose the image across the upper left to lower right axis:

```
jpegtran -transpose {{path/to/image.jpg}} > {{path/to/output.jpg}}
```

- Transverse the image across the upper right to lower left axis:

```
jpegtran -transverse {{path/to/image.jpg}} > {{path/to/output.jpg}}
```

- Convert the image to grayscale:

```
jpegtran -grayscale {{path/to/image.jpg}} > {{path/to/output.jpg}}
```

- Crop the image to a rectangular region of width `W` and height `H` from the upper left corner, saving the output to a specific file:

```
jpegtran -crop {{W}}x{{H}} -outfile {{path/to/output.jpg}} {{path/to/image.jpg}}
```

- Crop the image to a rectangular region of width `W` and height `H`, starting at point `X` and `Y` from the upper left corner:

```
jpegtran -crop {{W}}x{{H}}+{{X}}+{{Y}} {{path/to/image.jpg}} > {{path/to/output.jpg}}
```

kde-inhibit

Inhibit various desktop functions while a command runs.

- Inhibit power management:

  ```
  kde-inhibit --power {{command}} {{command_arguments}}
  ```

- Inhibit screen saver:

  ```
  kde-inhibit --screenSaver {{command}} {{command_arguments}}
  ```

- Launch vlc, and inhibit colour correction (night mode) while it's running:

  ```
  kde-inhibit --colorCorrect {{vlc}}
  ```

kdocker

E asily dock applications to the system tray.

More information: https://github.com/user-none/KDocker.

- Display a cursor to send a window to the system tray when pressing the left mouse button (press any other mouse button to cancel):

  ```
  k dock er
  ```

- O pen an application and send it to the system tray:

  ```
  k dock er {{application}}
  ```

- S end focused window to the system tray:

  ```
  k dock er - f
  ```

- Display a cursor to send a window to the system tray with a custom icon when pressing the left mouse button:

  ```
  k dock er - i {{/path/to/icon}}
  ```

- O pen an application, send it to the system tray and if focus is lost, minimiz e it:

  ```
  k dock er - l {{application}}
  ```

- P rint version:

  ```
  k dock er - - v ersion
  ```

kexec

Directly reboot into a new kernel.

- Load a new kernel:

  ```
  k exec - l {{path/to/k ernel}} - - initrd= {{path/to/initrd}} - -
  com m and- line= {{arg u m ents}}
  ```

- Load a new kernel with current boot parameters:

  ```
  k exec - l {{path/to/k ernel}} - - initrd= {{path/to/initrd}} - -
  reu se- cm dline
  ```

- E xecute a currently loaded kernel:

  ```
  k exec - e
  ```

- U nload current kexec target kernel:

  ```
  k exec - u
  ```

kj v

T he word of G od available right on your desktop.

More information: https://github.com/bontibon/kj .v

- Display books:

  `k j v - l`

- O pen a specific book:

  `k j v {{G enesis}}`

- O pen a specific chapter of a book:

  `k j v {{G enesis}} {{2 }}`

- O pen a specific verse of a specific chapter of a book:

  `k j v {{J ohn}} {{3 }}: {{1 6 }}`

- O pen a specific range of verses of a book' s chapter:

  `k j v {{P rov erb s}} {{3 }}: {{1 - 6 }}`

- Display a specific range of verses of a book from different chapters:

  `k j v {{M atthew }} {{1 }}: {{7 }}- {{2 }}: {{6 }}`

- Display all verses that match a pattern:

  `k j v /{{P lag u es}}`

- Display all verses that match a pattern in a specific book:

  `k j v {{1 J n}}/{{antichrist}}`

konsole

Konsole: The KDE terminal emulator.

More information: https://konsole.kde.org.

- Open a new Konsole in a specific directory:

  ```
  konsole --workdir {{path/to/directory}}
  ```

- Run a specific command and do not close the window after it exits:

  ```
  konsole --noclose -e {{command and}}
  ```

- Open a new tab:

  ```
  konsole --new-tab
  ```

- Open a Konsole in the background and bring to the front when Ctrl+Shift+F12 (by default) is pressed:

  ```
  konsole --background-mode
  ```

- Open a Konsole with the emergency FALLBACK profile:

  ```
  konsole --fallback-profile
  ```

kpackagetool5

KPackage Manager: Install, list, remove Plasma packages.

More information: https://techbase.kde.org/Development/Tutorials/Plasma5/QML2/GettingStarted#Kpackagetool5

- List all known package types that can be installed:

```
kpackagetool5 --list-types
```

- Install the package from a directory:

```
kpackagetool5 --type {{package_type}} --install {{path/to/directory}}
```

- Update installed package from a directory:

```
kpackagetool5 --type {{package_type}} --upgrade {{path/to/directory}}
```

- List installed plasmoids (--global for all users):

```
kpackagetool5 --type Plasma/Applet --list --global
```

- Remove a plasmoid by name:

```
kpackagetool5 --type Plasma/Applet --remove "{{name}}"
```

kpartx

Create device maps from partition tables.

- Add partition mappings:

  ```
  k partx - a {{w hole_disk . im g }}
  ```

- Delete partition mappings:

  ```
  k partx - d {{w hole_disk . im g }}
  ```

- List partition mappings:

  ```
  k partx - l {{w hole_disk . im g }}
  ```

kreadconfig5

Read KConfig entries for KDE Plasma.

More information: https://userbase.kde.org/KDE_System_Administration/Configuration_Files

- Read a key from the global configuration:

```
kreadconfig5 --group {{group_name}} --key {{key_name}}
```

- Read a key from a specific configuration file:

```
kwriteconfig5 --file {{path/to/file}} --group {{group_name}} --key {{key_name}}
```

- Check if systemd is used to start the Plasma session:

```
kreadconfig5 --file {{startkdercc}} --group {{General}} --key {{systemdBoot}}
```

kscreen-console

Command-line tool to q uery KS creen' s status.

More information: https://manned.org/kscreen-console.

- S how all outputs and configuration files to attach to a bug report:

  ```
  k screen- console b u g
  ```

- S how paths to KS creen configuration files:

  ```
  k screen- console config
  ```

- S how KS creen output information and configuration:

  ```
  k screen- console ou tpu ts
  ```

- Monitor for changes:

  ```
  k screen- console m onitor
  ```

- S how the current KS creen configuration as J S O N :

  ```
  k screen- console j son
  ```

- Display help:

  ```
  k screen- console - - help
  ```

- Display help including Q t specific command-line options:

  ```
  k screen- console - - help- all
  ```

kscreen-doctor

Change and manipulate the screen setup from the command-line.

More information: https://invent.kde.org/plasma/libkscreen.

- Show display output information:

```
kscreen-doctor --outputs
```

- Set the rotation of a display output with an ID of 1 to the right:

```
kscreen-doctor {{output.1.rotation.right}}
```

- Set the scale of a display output with an ID of HDMI-2 to 2 (200%):

```
kscreen-doctor {{output.HDMI-2.scale.2}}
```

ksvgtopng5

Convert S V G files to P N G format.

More information: https://invent.kde.org/plasma/kde-cli-tools/-/blob/master/ksvgtopng/ksvgtopng.cpp.

- Convert an S V G file (should be an absolute path) to P N G :

```
k sv g topng 5  {{w idth}} {{heig ht}} {{path/to/file. sv g }} {{ou tpu t_filenam e. png }}
```

kwriteconfig5

Write KConfig entries for KDE Plasma.

More information: https://userbase.kde.org/KDE_System_Administration/Configuration_Files

- Display help:

```
kwriteconfig5 --help
```

- Set a global configuration key:

```
kwriteconfig5 --group {{group_name}} --key {{key}} {{value}
```

- Set a key in a specific configuration file:

```
kwriteconfig5 --file {{path/to/file}} --group {{group_name}}
--key {{key}} {{value}}
```

- Delete a key:

```
kwriteconfig5 --group {{group_name}} --key {{key}} --delete
```

- Use systemd to start the Plasma session when available:

```
kwriteconfig5 --file {{startkdercc}} --group {{General}} --k e
{{systemdBoot}} {{true}}
```

- Hide the title bar when a window is maximized (like Ubuntu):

```
kwriteconfig5 --file {{~/.config/kwinrc}} --group {{Window s
--key {{BorderlessMaximizedWindows}} {{true}}
```

- Configure KRunner to open with the Meta (Command/Windows) global hotkey:

```
kwriteconfig5 --file {{~/.config/kwinrc}} --group
{{ModifierOnlyShortcuts}} --key {{Meta}}
{{"org.kde.kglobalaccel,/component/
krunner_desktop,org.kde.kglobalaccel.Component,invokeShort
```

larasail

A CLI tool for managing Laravel on Digital Ocean servers.

More information: https://github.com/thedevdojo/larasail

- Set up the server with Laravel dependencies using the default PHP version:

  ```
  larasail setup
  ```

- Set up the server with Laravel dependencies using a specific PHP version:

  ```
  larasail setup {{php71}}
  ```

- Add a new Laravel site:

  ```
  larasail host {{domain}} {{path/to/site_directory}}
  ```

- Retrieve the Larasail user password:

  ```
  larasail pass
  ```

- Retrieve the Larasail MySQL password:

  ```
  larasail mysqlpass
  ```

lastb

Show a listing of last logged in users.

- Show a list of all last logged in users:

  ```
  sudo lastb
  ```

- Show a list of all last logged in users since a given time:

  ```
  sudo lastb --since {{YYYY-MM-DD}}
  ```

- Show a list of all last logged in users until a given time:

  ```
  sudo lastb --until {{YYYY-MM-DD}}
  ```

- Show a list of all logged in users at a specific time:

  ```
  sudo lastb --present {{hh:mm}}
  ```

- Show a list of all last logged in users and translate the IP into a hostname:

  ```
  sudo lastb --dns
  ```

lastcomm

Show last commands executed.

More information: https://manpages.debian.org/stable/acct/lastcomm.1.en.html

- Print information about all of the commands in the acct (record file):

  ```
  lastcomm
  ```

- Display commands executed by a given user:

  ```
  lastcomm --user {{user}}
  ```

- Display information about a given command executed on the system:

  ```
  lastcomm --command {{command}}
  ```

- Display information about commands executed on a given terminal:

  ```
  lastcomm --tty {{terminal_name}}
  ```

lastlog

Show the most recent login of all users or of a given user.

- Display the most recent login of all users:

```
lastlog
```

- Display lastlog record of the specified user:

```
lastlog -u {{username}}
```

- Display records before than 7 days:

```
lastlog -b {{7}}
```

- Display records more recent than 3 days:

```
lastlog -t {{3}}
```

ldconfig

Configure symlinks and cache for shared library dependencies.

- Update symlinks and rebuild the cache (usually run when a new library is installed):

  ```
  sudo ldconfig
  ```

- Update the symlinks for a given directory:

  ```
  sudo ldconfig -n {{path/to/directory}}
  ```

- Print the libraries in the cache and check whether a given library is present:

  ```
  ldconfig -p | grep {{library_name}}
  ```

ldd

Display shared library dependencies.

- Display shared library dependencies of a binary:

  ```
  ldd {{path/to/b inary}}
  ```

- Display unused direct dependencies:

  ```
  ldd - u  {{path/to/b inary}}
  ```

legit

Complementary command-line interface for G it.

More information: https://frostming.github.io/legit.

- S witch to a specified branch, stashing and restoring unstaged changes:

  ```
  g it  sw itch  {{targ et_b ranch}}
  ```

- S ynchroniz e current branch, automatically merging or rebasing, and stashing and unstashing:

  ```
  g it  sync
  ```

- P ublish a specified branch to the remote server:

  ```
  g it  pu b lish  {{b ranch_nam e}}
  ```

- R emove a branch from the remote server:

  ```
  g it  u npu b lish  {{b ranch_nam e}}
  ```

- List all branches and their publication status:

  ```
  g it  b ranches  {{g lob _pattern}}
  ```

- R emove the last commit from the history:

  ```
  g it  u ndo  {{- - hard}}
  ```

lftp

Sophisticated file transfer program.

More information: https://lftp.yar.ru/lftp-man.html.

- Connect to an FTP server:

  ```
  lftp {{ftp.example.com}}
  ```

- Download multiple files (glob expression):

  ```
  mget {{path/to/*.png}}
  ```

- Upload multiple files (glob expression):

  ```
  mput {{path/to/*.zip}}
  ```

- Delete multiple files on the remote server:

  ```
  mrm {{path/to/*.txt}}
  ```

- Rename a file on the remote server:

  ```
  mv {{original_filename}} {{new_filename}}
  ```

- Download or update an entire directory:

  ```
  mirror {{path/to/remote_dir}} {{path/to/local_output_dir}}
  ```

- Upload or update an entire directory:

  ```
  mirror -R {{path/to/local_dir}} {{path/to/remote_output_dir}}
  ```

libreoffice

CLI for the powerful and free office suite LibreO ffice.

More information: https://www.libreoffice.org/.

- O pen a space-separated list of files in read-only mode:

  ```
  lib reoffice - - v iew  {{path/to/file1 }} {{path/to/file2 }}
  ```

- Display the content of specific files:

  ```
  lib reoffice - - cat {{path/to/file1 }} {{path/to/file2 }}
  ```

- P rint files to a specific printer:

  ```
  lib reoffice - - pt {{printer_nam e}} {{path/to/file1 }} {{path/
  to/file2 }}
  ```

- Convert all `. doc`files in current directory to pdf:

  ```
  lib reoffice - - conv ert- to {{pdf}} {{* . doc}}
  ```

light

CLI to control the backlight of your screen.

- G et the current backlight value in percent:

  ```
  lig ht
  ```

- S et the backlight value to 5 0 percent:

  ```
  lig ht - S  {{5 0 }}
  ```

- R educe 2 0 percent from the current backlight value:

  ```
  lig ht - U  {{2 0 }}
  ```

- Add 2 0 percent to the current backlight value:

  ```
  lig ht - A  {{2 0 }}
  ```

line

R ead a single line of input.

- R ead input:

  ```
  line
  ```

locate

Find filenames quickly.

- Look for pattern in the database. Note: the database is recomputed periodically (usually weekly or daily):

  ```
  locate {{pattern}}
  ```

- Look for a file by its exact filename (a pattern containing no globbing characters is interpreted as `*pattern*`):

  ```
  locate */{{filename}}
  ```

- Recompute the database. You need to do it if you want to find recently added files:

  ```
  sudo updatedb
  ```

logger

Add messages to syslog (/var/log/syslog).

- Log a message to syslog:

```
logger {{message}}
```

- Take input from stdin and log to syslog:

```
echo {{log_entry}} | logger
```

- Send the output to a remote syslog server running at a given port. Default port is 514:

```
echo {{log_entry}} | logger --server {{hostname}} --port {{port}}
```

- Use a specific tag for every line logged. Default is the name of logged in user:

```
echo {{log_entry}} | logger --tag {{tag}}
```

- Log messages with a given priority. Default is `user.notice`. See `man logger` for all priority options:

```
echo {{log_entry}} | logger --priority {{user.warning}}
```

login

Initiates a session for a user.

- Log in as a user:

  ```
  log in {{u ser}}
  ```

- Log in as user without authentication if user is preauthenticated:

  ```
  log in - f {{u ser}}
  ```

- Log in as user and preserve environment:

  ```
  log in - p {{u ser}}
  ```

- Log in as a user on a remote host:

  ```
  log in - h {{host}} {{u ser}}
  ```

logsave

Save the output of a command in a logfile.

More information: https://manned.org/logsave.

- Execute command with specified argument(s) and save its output to log file:

  ```
  logsave {{path/to/log file}} {{command and}}
  ```

- Take input from standard input and save it in a log file:

  ```
  logsave {{log file}} -
  ```

- Append the output to a log file, instead of replacing its current contents:

  ```
  logsave -a {{log file}} {{command and}}
  ```

- Show verbose output:

  ```
  logsave -v {{log file}} {{command and}}
  ```

logwatch

Summarizes many different logs for common services (e.g., apache, pam_unix, sshd, etc.) in a single report.

- Analyze logs for a range of dates at certain level of detail:

  ```
  logwatch --range {{yesterday|today|all|help}} --detail {{low|medium|others}}'
  ```

- Restrict report to only include information for a selected service:

  ```
  logwatch --range {{all}} --service {{apache|pam_unix|etc}}
  ```

losetup

Set up and control loop devices.

- List loop devices with detailed info:

  ```
  losetup -a
  ```

- Attach a file to a given loop device:

  ```
  sudo losetup /dev/{{loop}} /{{path/to/file}}
  ```

- Attach a file to a new free loop device and scan the device for partitions:

  ```
  sudo losetup --show --partscan -f /{{path/to/file}}
  ```

- Attach a file to a read-only loop device:

  ```
  sudo losetup --read-only /dev/{{loop}} /{{path/to/file}}
  ```

- Detach all loop devices:

  ```
  sudo losetup -D
  ```

- Detach a given loop device:

  ```
  sudo losetup -d /dev/{{loop}}
  ```

lrunzip

A large file decompression program.

See also: `lrzip`, `plzip`, `lrzip-tar`, `lrzuntar`.

- Decompress a file:

  ```
  lrunzip {{filename.lrz}}
  ```

- Decompress a file using a specific number of processor threads:

  ```
  lrunzip -p {{8}} {{filename.lrz}}
  ```

- Decompress a file and silently overwrite files if they exist:

  ```
  lrunzip -f {{filename.lrz}}
  ```

- Keep broken or damaged files instead of deleting them when decompressing:

  ```
  lrunzip -K {{filename.lrz}}
  ```

- Specify output file name and/or path:

  ```
  lrunzip -o {{outfilename}} {{filename.lrz}}
  ```

lrzip

A large file compression program.

See also `lrunzip`, `lrzip-tar`, `lrzuntar`.

- Compress a file with LZMA - slow compression, fast decompression:

  ```
  lrzip {{filename}}
  ```

- Compress a file with BZIP2 - good middle ground for compression/speed:

  ```
  lrzip -b {{filename}}
  ```

- Compress with ZPAQ - extreme compression, but very slow:

  ```
  lrzip -z {{filename}}
  ```

- Compress with LZO - light compression, extremely fast decompression:

  ```
  lrzip -l {{filename}}
  ```

- Compress a file and password protect/encrypt it:

  ```
  lrzip -e {{filename}}
  ```

- Override the number of processor threads to use:

  ```
  lrzip -p {{8}} {{filename}}
  ```

lrztar

A wrapper for `lrzip` to simplify compression of directories.

See also: `tar`, `lrzuntar`, `lrzunzip`.

- Archive a directory with `tar`, then compress:

  ```
  lrztar {{path/to/directory}}
  ```

- Same as above, with ZPAQ - extreme compression, but very slow:

  ```
  lrztar -z {{path/to/directory}}
  ```

- Specify the output file:

  ```
  lrztar -o {{path/to/file}} {{path/to/directory}}
  ```

- Override the number of processor threads to use:

  ```
  lrztar -p {{8}} {{path/to/directory}}
  ```

- Force overwriting of existing files:

  ```
  lrztar -f {{path/to/directory}}
  ```

lrz untar

A wrapper for `lrunzip` to simplify decompression of directories.

See also: `lrztar`, `lrzip`.

- Decompress from a file to the current directory:

  ```
  lrzuntar {{path/to/archive.tar.lrz}}
  ```

- Decompress from a file to the current directory using a specific number of processor threads:

  ```
  lrzuntar -p {{8}} {{path/to/archive.tar.lrz}}
  ```

- Decompress from a file to the current directory and silently overwrite items that already exist:

  ```
  lrzuntar -f {{archive.tar.lrz}}
  ```

- Specify the output path:

  ```
  lrzuntar -O {{path/to/directory}} {{archive.tar.lrz}}
  ```

- Delete the compressed file after decompression:

  ```
  lrzuntar -D {{path/to/archive.tar.lrz}}
  ```

lsattr

List file attributes on a Linux filesystem.

- Display the attributes of the files in the current directory:

  ```
  lsattr
  ```

- List the attributes of files in a particular path:

  ```
  lsattr {{path}}
  ```

- List file attributes recursively in the current and subseq uent directories:

  ```
  lsattr - R
  ```

- S how attributes of all the files in the current directory, including hidden ones:

  ```
  lsattr - a
  ```

- Display attributes of directories in the current directory:

  ```
  lsattr - d
  ```

lsb_release

Provides certain LSB (Linux Standard Base) and distribution-specific information.

- Print all available information:

  ```
  lsb_release -a
  ```

- Print a description (usually the full name) of the operating system:

  ```
  lsb_release -d
  ```

- Print only the operating system name (ID), suppressing the field name:

  ```
  lsb_release -i -s
  ```

- Print the release number and codename of the distribution, suppressing the field names:

  ```
  lsb_release -rcs
  ```

lsblk

Lists information about devices.

- List all storage devices in a tree-like format:

  ```
  lsblk
  ```

- Also list empty devices:

  ```
  lsblk -a
  ```

- Print the SIZE column in bytes rather than in a human-readable format:

  ```
  lsblk -b
  ```

- Output info about filesystems:

  ```
  lsblk -f
  ```

- Use ASCII characters for tree formatting:

  ```
  lsblk -i
  ```

- Output info about block-device topology:

  ```
  lsblk -t
  ```

- Exclude the devices specified by the comma-separated list of major device numbers:

  ```
  lsblk -e {{1,7}}
  ```

- Display a customized summary using a comma-separated list of columns:

  ```
  lsblk --output {{NAME}},{{SERIAL}},{{MODEL}},{{TRAN
  {{TYPE}},{{SIZE}},{{FSTYPE}},{{MOUNTPOINT}}
  ```

lscpu

Displays information about the CPU architecture.

- Display information about all CPUs:

  ```
  lscpu
  ```

- Display information in a table:

  ```
  lscpu --extended
  ```

- Display only information about offline CPUs in a table:

  ```
  lscpu --extended --offline
  ```

lshw

List detailed information about hardware configurations as root user.

- Launch the G U I:

  ```
  su do lshw  - X
  ```

- List all hardwares in tabular format:

  ```
  su do lshw  - short
  ```

- List all disks and storage controllers in tabular format:

  ```
  su do lshw  - class disk  - class storag e - short
  ```

- S ave all network interfaces to an H T ML file:

  ```
  su do lshw  - class netw ork  - htm l >  {{interfaces. htm l}}
  ```

lslocks

List local system locks.

- List all local system locks:

```
lslock s
```

- List locks with defined column headers:

```
lslock s - - ou tpu t {{P I D }}, {{C O M M A N D }}, {{P A T H }}
```

- List locks producing a raw output (no columns), and without column headers:

```
lslock s - - raw  - - noheading s
```

- List locks by P ID input:

```
lslock s - - pid {{P I D }}
```

- List locks with j son output to stdout:

```
lslock s - - j son
```

lslogins

Show information about users on a Linux system.

More information: https://man7.org/linux/man-pages/man1/lslogins.1.html

- Display users in the system:

```
lslogins
```

- Display users belonging to a specific group:

```
lslogins --group={{group}}
```

- Display user accounts:

```
lslogins --user-accs
```

- Display last logins:

```
lslogins --last
```

- Display system accounts:

```
lslogins --system-accs
```

- Display supplementary groups:

```
lslogins --supp-groups
```

lsmod

Shows the status of linux kernel modules.

See also **`modprobe`**, which loads kernel modules.

- List all currently loaded kernel modules:

  ```
  lsmod
  ```

lspci

List all PCI devices.

- Show a brief list of devices:

  ```
  lspci
  ```

- Display additional info:

  ```
  lspci -v
  ```

- Display drivers and modules handling each device:

  ```
  lspci -k
  ```

- Show a specific device:

  ```
  lspci -s {{0 0 : 1 8 . 3 }}
  ```

- Dump info in a readable form:

  ```
  lspci -v m
  ```

lsscsi

List S CS I devices (or hosts) and their attributes.

- List all S CS I devices:

  ```
  lsscsi
  ```

- List all S CS I devices with detailed attributes:

  ```
  lsscsi -L
  ```

- List all S CS I devices with human readable disk capacity:

  ```
  lsscsi -s
  ```

lsusb

Display information about U S B buses and devices connected to them.

- List all the U S B devices available:

  ```
  lsu sb
  ```

- List the U S B hierarchy as a tree:

  ```
  lsu sb - t
  ```

- List verbose information about U S B devices:

  ```
  lsu sb - - v erb ose
  ```

- List detailed information about a U S B device:

  ```
  lsu sb - D {{dev ice}}
  ```

- List devices with a specified vendor and product id only:

  ```
  lsu sb - d {{v endor}}: {{produ ct}}
  ```

ltrace

Display dynamic library calls of a process.

More information: https://manned.org/ltrace.

- Print (trace) library calls of a program binary:

```
ltrace ./{{program}}
```

- Count library calls. Print a handy summary at the bottom:

```
ltrace -c {{path/to/program}}
```

- Trace calls to malloc and free, omit those done by libc:

```
ltrace -e malloc+free-@libc.so* {{path/to/program}}
```

- Write to file instead of terminal:

```
ltrace -o {{file}} {{path/to/program}}
```

lvcreate

Creates a logical volume in an existing volume group. A volume group is a collection of logical and physical volumes.

See also: `lvm`.

More information: https://man7.org/linux/man-pages/man8/lvcreate.8.html

- Create a logical volume of 10 gigabytes in the volume group vg1:

```
lvcreate -L {{10G}} {{vg1}}
```

- Create a 1500 megabyte linear logical volume named mylv in the volume group vg1:

```
lvcreate -L {{1500}} -n {{mylv}} {{vg1}}
```

- Create a logical volume called mylv that uses 60% of the total space in volume group vg1:

```
lvcreate -l {{60%VG}} -n {{mylv}} {{vg1}}
```

- Create a logical volume called mylv that uses all of the unallocated space in the volume group vg1:

```
lvcreate -l {{100%FREE}} -n {{mylv}} {{vg1}}
```

lvdisplay

Display information about Logical Volume Manager (LVM) logical volumes.

See also: `lvm`.

More information: https://man7.org/linux/man-pages/man8/lvdisplay.8.html

- Display information about all logical volumes:

  ```
  sudo lvdisplay
  ```

- Display information about all logical volumes in volume group vg1:

  ```
  sudo lvdisplay {{vg1}}
  ```

- Display information about logical volume lv1 in volume group vg1:

  ```
  sudo lvdisplay {{vg1/lv1}}
  ```

lvextend

Increase the size of a logical volume.

See also: `lvm`.

More information: https://man7.org/linux/man-pages/man8/lvextend.8.html

- Increase a volume's size to 120 GB:

  ```
  lvextend --size {{120G}} {{logical_volume}}
  ```

- Increase a volume's size by 40 GB as well as the underlying filesystem:

  ```
  lvextend --size +{{40G}} -r {{logical_volume}}
  ```

- Increase a volume's size to 100% of the free physical volume space:

  ```
  lvextend --size {{100}}%FREE {{logical_volume}}
  ```

lvm

Manage physical volumes, volume groups, and logical volumes using the Logical Volume Manager (LVM) interactive shell.

More information: https://man7.org/linux/man-pages/man8/lvm.8.html

- Start the Logical Volume Manager interactive shell:

  ```
  sudo lvm
  ```

- List the Logical Volume Manager commands:

  ```
  sudo lvm help
  ```

- Initialize a drive or partition to be used as a physical volume:

  ```
  sudo lvm pv create {{/dev/sdXY}}
  ```

- Display information about physical volumes:

  ```
  sudo lvm pv display
  ```

- Create a volume group called vg1 from the physical volume on `/dev/sdXY`:

  ```
  sudo lvm vg create {{vg1}} {{/dev/sdXY}}
  ```

- Display information about volume groups:

  ```
  sudo lvm vg display
  ```

- Create a logical volume with size 10G from volume group vg1:

  ```
  sudo lvm lv create -L {{10G}} {{vg1}}
  ```

- Display information about logical volumes:

  ```
  sudo lvm lv display
  ```

lvreduce

Reduce the size of a logical volume.

See also: `lvm`.

More information: https://man7.org/linux/man-pages/man8/lvreduce.8.html

- Reduce a volume's size to 120 GB:

```
lvreduce --size {{120G}} {{logical_volume}}
```

- Reduce a volume's size by 40 GB as well as the underlying filesystem:

```
lvreduce --size -{{40G}} -r {{logical_volume}}
```

lvremove

Remove one or more logical volumes.

See also: `lvm`.

More information: https://man7.org/linux/man-pages/man8/lvremove.8.html

- Remove a logical volume in a volume group:

  ```
  sudo lvremove {{volume_group}}/{{logical_volume}}
  ```

- Remove all logical volumes in a volume group:

  ```
  sudo lvremove {{volume_group}}
  ```

lvresize

Change the size of a logical volume.

See also: `lvm`.

More information: https://man7.org/linux/man-pages/man8/lvresize.8.html

- Change the size of a logical volume to 120 GB:

```
lvresize --size {{120G}} {{volume_group}}/{{logical_volume}}
```

- Extend the size of a logical volume as well as the underlying filesystem by 120 GB:

```
lvresize --size +{{120G}} --resizefs {{volume_group}}/
{{logical_volume}}
```

- Extend the size of a logical volume to 100% of the free physical volume space:

```
lvresize --size {{100}}%FREE {{volume_group}}/
{{logical_volume}}
```

- Reduce the size of a logical volume as well as the underlying filesystem by 120 GB:

```
lvresize --size -{{120G}} --resizefs {{volume_group}}/
{{logical_volume}}
```

lvs

Display information about logical volumes.

See also: `lvm`.

More information: https://man7.org/linux/man-pages/man8/lvs.8.html

- Display information about logical volumes:

  ```
  lvs
  ```

- Display all logical volumes:

  ```
  lvs -a
  ```

- Change default display to show more details:

  ```
  lvs -v
  ```

- Display only specific fields:

  ```
  lvs -o {{field_name_1}},{{field_name_2}}
  ```

- Append field to default display:

  ```
  lvs -o +{{field_name}}
  ```

- Suppress heading line:

  ```
  lvs --noheadings
  ```

- Use a separator to separate fields:

  ```
  lvs --separator {{=}}
  ```

lxc

Manage Linux containers using the lxd R E S T AP I.

Any container names or patterns can be prefixed with the name of a remote server.

- List local containers matching a string. O mit the string to list all local containers:

  ```
  lxc list {{m atch_string }}
  ```

- List images matching a string. O mit the string to list all images:

  ```
  lxc im ag e list [ {{rem ote}}: ] {{m atch_string }}
  ```

- Create a new container from an image:

  ```
  lxc init [ {{rem ote}}: ] {{im ag e}} {{container}}
  ```

- S tart a container:

  ```
  lxc start [ {{rem ote}}: ] {{container}}
  ```

- S top a container:

  ```
  lxc stop [ {{rem ote}}: ] {{container}}
  ```

- S how detailed info about a container:

  ```
  lxc info [ {{rem ote}}: ] {{container}}
  ```

- T ake a snapshot of a container:

  ```
  lxc snapshot [ {{rem ote}}: ] {{container}} {{snapshot}}
  ```

lxi

Control LXI compatible instruments such as oscilloscopes.

More information: https://github.com/lxi-tools/lxi-tools.

- Discover LXI devices on available networks:

```
lxi discover
```

- Capture a screenshot, detecting a plugin automatically:

```
lxi screenshot --address {{ip_address}}
```

- Capture a screenshot using a specified plugin:

```
lxi screenshot --address {{ip_address}} --plugin {{rigol-1000z}}
```

- Send an SCPI command to an instrument:

```
lxi scpi --address {{ip_address}} "{{*IDN?}}"
```

- Run a benchmark for request and response performance:

```
lxi benchmark --address {{ip_address}}
```

lxterminal

T erminal emulator for LX DE .

More information: https://wiki.lxde.org/en/LX T erminal

- O pen an LX T erminal window:

  ```
  lxterm inal
  ```

- O pen an LX T erminal window, run a command, and then exit:

  ```
  lxterm inal - e " {{com m and}}"
  ```

- O pen an LX T erminal window with multiple tabs:

  ```
  lxterm inal - - tab s= {{tab _nam e1 , tab _nam e2 , . . . }}
  ```

- O pen an LX T erminal window with a specific title:

  ```
  lxterm inal - - title= {{title_nam e}}
  ```

- O pen an LX T erminal window with a specific working directory:

  ```
  lxterm inal - - w ork ing - directory= {{path/to/directory}}
  ```

lynis

System and security auditing tool.

More information: https://cisofy.com/documentation/lynis/.

- Check that Lynis is up-to-date:

  ```
  sudo lynis update info
  ```

- Run a security audit of the system:

  ```
  sudo lynis audit system
  ```

- Run a security audit of a Dockerfile:

  ```
  sudo lynis audit dockerfile {{path/to/dockerfile}}
  ```

mac2unix

Change macOS-style line endings to Unix-style.

Replaces LF with CR.

- Change the line endings of a file:

  ```
  mac2unix {{filename}}
  ```

- Create a copy with Unix-style line endings:

  ```
  mac2unix -n {{filename}} {{new_filename}}
  ```

macchanger

Command-line utility for manipulating network interface MAC addresses.

- View the current and permanent MAC addresses of a interface:

  ```
  macchanger --show {{interface}}
  ```

- Set interface to a random MAC:

  ```
  macchanger --random {{interface}}
  ```

- Set interface to a specific MAC:

  ```
  macchanger --mac {{XX:XX:XX:XX:XX:XX}} {{interface}}
  ```

- Reset interface to its permanent hardware MAC:

  ```
  macchanger --permanent {{interface}}
  ```

maim

Screenshot utility.

More information: https://github.com/naelstrof/maim.

- Capture a screenshot and save it to the given path:

  ```
  maim {{path/to/screenshot.png}}
  ```

- Capture a screenshot of the selected region:

  ```
  maim --select {{path/to/screenshot.png}}
  ```

- Capture a screenshot of the selected region and save it in the clipboard (requires `xclip`):

  ```
  maim --select | xclip -selection clipboard -target image/png
  ```

- Capture a screenshot of the current active window (requires `xdotool`):

  ```
  maim --window $(xdotool getactivewindow) {{path/to/screenshot.png}}
  ```

makepkg

Creates a package installable with the **`pacman`** package manager.

Runs the commands from a **`PKGBUILD`** file to build the package.

More information: https://wiki.archlinux.org/index.php/Makepkg.

- Make a package (run in the same directory as a `PKGBUILD`):

  `makepkg`

- Make a package and install its dependencies:

  `makepkg --syncdeps`

- Same as above, but install the package with `pacman` when done:

  `makepkg --syncdeps --install`

- Make a package, but skip source checksums:

  `makepkg --skipchecksums`

mandb

Manage the pre-formatted manual page database.

More information: https://man7.org/linux/man-pages/man8/mandb.8.html

- Purge and process manual pages:

```
mandb
```

- Update a single entry:

```
mandb --filename {{path/to/file}}
```

- Create entries from scratch instead of updating:

```
mandb --create
```

- Only process user databases:

```
mandb --user-db
```

- Do not purge obsolete entries:

```
mandb --no-purge
```

- Check the validity of manual pages:

```
mandb --test
```

manpath

Determine the search path for manual pages.

- Display the search path used to find man pages:

  ```
  m anpath
  ```

- S how the entire global manpath:

  ```
  m anpath - - g lob al
  ```

mcookie

Generates random 128 bit hexadecimal numbers.

- Generate a random number:

  ```
  mcookie
  ```

- Generate a random number, using the contents of a file as a seed for the randomness:

  ```
  mcookie --file {{path/to/file}}
  ```

- Generate a random number, using a specific number of bytes from a file as a seed for the randomness:

  ```
  mcookie --file {{path/to/file}} --max-size {{number_of_bytes}}
  ```

- Print the details of the randomness used, such as the origin and seed for each source:

  ```
  mcookie --verbose
  ```

mdadm

RAID management utility.

More information: https://manned.org/mdadm.

- Create array:

```
mdadm --create {{/dev/md/MyRAID}} --level {{raid_level}} --
raid-devices {{number_of_disks}} {{/dev/sdXN}}
```

- Stop array:

```
mdadm --stop {{/dev/md0}}
```

- Mark disk as failed:

```
mdadm --fail {{/dev/md0}} {{/dev/sdXN}}
```

- Remove disk:

```
mdadm --remove {{/dev/md0}} {{/dev/sdXN}}
```

- Add disk to array:

```
mdadm --assemble {{/dev/md0}} {{/dev/sdXN}}
```

- Show RAID info:

```
mdadm --detail {{/dev/md0}}
```

mdbook

Create online books by writing makrdown files.

More information: [https://rust-lang.github.io/mdB ook/](https://rust-lang.github.io/mdBook/)

- Create a mdbook proj ect in the current directory:

  `m db ook init`

- Create a mdbook proj ect in a specific directory:

  `m db ook init {{path/to/directory}}`

- Clean the directory with the generated book:

  `m db ook clean`

- S erve a book at `http: //localhost: 3 0,0 0`, auto build when file changes:

  `m db ook serv e`

- Watch a set of Markdown files and automatically build when a file is changed:

  `m db ook w atch`

Medusa

A modular and parallel login brute-forcer for a variety of protocols.

- Execute brute force against an FTP server using a file containing usernames and a file containing passwords:

  ```
  medusa -M ftp -h host -U {{path/to/username_file}} -P {{path/to/password_file}}
  ```

- Execute a login attempt against a HTTP server using the username, password and user-agent specified:

  ```
  medusa -M HTTP -h host -u {{username}} -p {{password}} -m USER-AGENT:"{{Agent}}"
  ```

- Execute a brute force against a MySQL server using a file containing usernames and a hash:

  ```
  medusa -M mysql -h host -U {{path/to/username_file}} -p {{hash}} -m PASS:HASH
  ```

- Execute a brute force against a list of SMB servers using a username and a pwdump file:

  ```
  medusa -M smbnt -H {{path/to/hosts_file}} -C {{path/to/pwdump_file}} -u {{username}} -m PASS:HASH
  ```

microcom

A minimalistic terminal program, used to access remote devices via a serial, CAN or telnet connection from the console.

- Open a serial port using the specified baud rate:

  ```
  microcom --port {{path/to/serial_port}} --speed {{baud_rate}}
  ```

- Establish a telnet connection to the specified host:

  ```
  microcom --telnet {{hostname}}:{{port}}
  ```

mimetype

Automatically determine the MIME type of a file.

- Print the MIME type of a given file:

  ```
  mimetype {{path/to/file}}
  ```

- Display only the MIME type, and not the filename:

  ```
  mimetype --brief {{path/to/file}}
  ```

- Display a description of the MIME type:

  ```
  mimetype --describe {{path/to/file}}
  ```

- Determine the MIME type of stdin (does not check a filename):

  ```
  {{some_command}} | mimetype --stdin
  ```

- Display debug information about how the MIME type was determined:

  ```
  mimetype --debug {{path/to/file}}
  ```

- Display all the possible MIME types of a given file in confidence order:

  ```
  mimetype --all {{path/to/file}}
  ```

- Explicitly specify the 2-letter language code of the output:

  ```
  mimetype --language {{path/to/file}}
  ```

mke2fs

Creates a Linux filesystem inside a partition.

- Create an ext2 filesystem in partition 1 of device b(sdb 1):

  ```
  mkfs.ext2 {{/dev/sdb1}}
  ```

- Create an ext3 filesystem in partition 1 of device b(sdb 1):

  ```
  mkfs.ext3 {{/dev/sdb1}}
  ```

- Create an ext4 filesystem in partition 1 of device b(sdb 1):

  ```
  mkfs.ext4 {{/dev/sdb1}}
  ```

mkfs.btrfs

Create a btrfs filesystem.

Defaults to `raid1`, which specifies 2 copies of a given data block spread across 2 different devices.

More information: https://btrfs.wiki.kernel.org/index.php/Manpage/mkfs.btrfs.

- Create a btrfs filesystem on a single device:

```
sudo mkfs.btrfs --metadata single --data single {{/dev/sda}}
```

- Create a btrfs filesystem on multiple devices with raid1:

```
sudo mkfs.btrfs --metadata raid1 --data raid1 {{/dev/sda}} {{/dev/sdb}} {{/dev/sdN}}
```

- Set a label for the filesystem:

```
sudo mkfs.btrfs --label "{{label}}" {{/dev/sda}} [{{/dev/sdN}}]
```

mkfs.cramfs

Creates a R O M filesystem inside a partition.

- Create a R O M filesystem inside partition 1 on device /dev/sdb 1:

  ```
  m k fs. cram fs {{/dev /sdb 1 }}
  ```

- Create a R O M filesystem with a volume-name:

  ```
  m k fs. cram fs - n {{v olu m e_nam e}} {{/dev /sdb 1 }}
  ```

mkfs.exfat

Creates an exfat filesystem inside a partition.

- Create an exfat filesystem inside partition 1 on device b (sdb 1):

  ```
  m k fs. exfat {{/dev /sdb 1 }}
  ```

- Create filesystem with a volume-name:

  ```
  m k fs. exfat - n {{v olu m e_nam e}} {{/dev /sdb 1 }}
  ```

- Create filesystem with a volume-id:

  ```
  m k fs. exfat - i {{v olu m e_id}} {{/dev /sdb 1 }}
  ```

mkfs.ext4

Creates an ext4 filesystem inside a partition.

- Create an ext4 filesystem inside partition 1 on device sdb:

```
sudo mkfs.ext4 {{/dev/sdb1}}
```

- Create an ext4 filesystem with a volume-label:

```
sudo mkfs.ext4 -L {{volume_label}} {{/dev/sdb1}}
```

mkfs.fat

Creates an MS-DOS filesystem inside a partition.

- Create a fat filesystem inside partition 1 on device b sdb 1:

  ```
  mkfs.fat {{/dev/sdb1}}
  ```

- Create filesystem with a volume-name:

  ```
  mkfs.fat -n {{volume_name}} {{/dev/sdb1}}
  ```

- Create filesystem with a volume-id:

  ```
  mkfs.fat -i {{volume_id}} {{/dev/sdb1}}
  ```

- Use 5 instead of 2 file allocation tables:

  ```
  mkfs.fat -f 5 {{/dev/sdb1}}
  ```

mkfs

Build a Linux filesystem on a hard disk partition.

This command is deprecated in favor of filesystem specific mkfs.utils.

- Build a Linux ext2 filesystem on a partition:

  ```
  mkfs {{path/to/partition}}
  ```

- Build a filesystem of a specified type:

  ```
  mkfs -t {{ext4}} {{path/to/partition}}
  ```

- Build a filesystem of a specified type and check for bad blocks:

  ```
  mkfs -c -t {{ntfs}} {{path/to/partition}}
  ```

mkfs.minix

Creates a Minix filesystem inside a partition.

- Create a Minix filesystem inside partition 1 on device b {sdb 1}:

```
mkfs.minix {{/dev/sdb1}}
```

mkfs.ntfs

Creates a NTFS filesystem inside a partition.

- Create a NTFS filesystem inside partition 1 on device sdb(1):

  ```
  mkfs.ntfs {{/dev/sdb1}}
  ```

- Create filesystem with a volume-label:

  ```
  mkfs.ntfs -L {{volume_label}} {{/dev/sdb1}}
  ```

- Create filesystem with specific UUID:

  ```
  mkfs.ntfs -U {{UUID}} {{/dev/sdb1}}
  ```

mkfs.vfat

Creates an MS-DOS filesystem inside a partition.

- Create a vfat filesystem inside partition 1 on device b (sdb 1):

  ```
  mkfs.vfat {{/dev/sdb1}}
  ```

- Create filesystem with a volume-name:

  ```
  mkfs.vfat -n {{volume_name}} {{/dev/sdb1}}
  ```

- Create filesystem with a volume-id:

  ```
  mkfs.vfat -i {{volume_id}} {{/dev/sdb1}}
  ```

- Use 5 instead of 2 file allocation tables:

  ```
  mkfs.vfat -f 5 {{/dev/sdb1}}
  ```

mkinitcpio

Generates initial ramdisk environments for booting the Linux kernel based on the specified preset(s).

More information: https://man.archlinux.org/man/mkinitcpio.8.

- Perform a dry run (print what would be done without actually doing it):

```
mkinitcpio
```

- Generate a ramdisk environment based on the `linux` preset:

```
mkinitcpio --preset {{linux}}
```

- Generate a ramdisk environment based on the `linux-lts` preset:

```
mkinitcpio --preset {{linux-lts}}
```

- Generate ramdisk environments based on all existing presets (used to regenerate all the initramfs images after a change in `/etc/mkinitcpio.conf`):

```
mkinitcpio --allpresets
```

- Generate an initramfs image using an alternative configuration file:

```
mkinitcpio --config {{path/to/mkinitcpio.conf}} --generate {{path/to/initramfs.img}}
```

- Generate an initramfs image for a kernel other than the one currently running (the installed kernel releases can be found in `/usr/lib/modules/`):

```
mkinitcpio --kernel {{kernelversion}} --generate {{path/to/initramfs.img}}
```

- List all available hooks:

```
mkinitcpio --listhooks
```

- Display help for a specific hook:

```
mkinitcpio --hookhelp {{hookname}}
```

mkisofs

Create ISO files from directories.

Also aliased as **`genisoimage`**

- Create an ISO from a directory:

  ```
  mkisofs -o {{filename.iso}} {{path/to/source_directory}}
  ```

- Set the disc label when creating an ISO:

  ```
  mkisofs -o {{filename.iso}} -V "{{label_name}}" {{path/to/source_directory}}
  ```

mklost+found

Create a lost+found directory.

More information: https://manned.org/mklost+found

- Create a `lost+found` directory in the current directory:

  `mklost+found`

mknod

Create block or character device special files.

More information: https://www.gnu.org/software/coreutils/mknod.

- Create a block device:

```
su do m k nod {{path/to/dev ice_file}} b {{m aj or_dev ice_nu m b er}} {{m inor_dev ice_nu m b er}}
```

- Create a character device:

```
su do m k nod {{path/to/dev ice_file}} c {{m aj or_dev ice_nu m b er}} {{m inor_dev ice_nu m b er}}
```

- Create a F IF O (q ueue) device:

```
su do m k nod {{path/to/dev ice_file}} p
```

- Create a device file with default S E Linux security context:

```
su do m k nod - Z {{path/to/dev ice_file}} {{type}} {{m aj or_dev ice_nu m b er}} {{m inor_dev ice_nu m b er}}
```

mkswap

Sets up a Linux swap area on a device or in a file.

- Setup a given partition as swap area:

  `sudo mkswap {{/dev/sdb7}}`

- Use a given file as swap area:

  `sudo mkswap {{path/to/file}}`

- Check a partition for bad blocks before creating the swap area:

  `sudo mkswap -c {{/dev/sdb7}}`

- Specify a label for the file (to allow `swapon` to use the label):

  `sudo mkswap -L {{swap1}} {{path/to/file}}`

mmcli

Control and monitor the ModemManager.

More information: https://www.freedesktop.org/software/ModemManager/man/latest/mmcli.1_.html

- List SMS messages available on the modem:

```
sudo mmcli --modem={{modem}} --messaging-list-sms
```

- Delete a message from the modem, specifying its path:

```
sudo mmcli --modem={{modem}} --messaging-delete-sms={{path/message_file}}
```

mocp

Music on Console (MO C) audio player.

More information: https://manned.org/mocp.

- Launch the MO C terminal U I:

  ```
  m ocp
  ```

- Launch the MO C terminal U I in a specific directory:

  ```
  m ocp {{path/to/directory}}
  ```

- S tart the MO C server in the background, without launching the MO C terminal U I:

  ```
  m ocp - - serv er
  ```

- Add a specific song to the play q ueue while MO C is in the background:

  ```
  m ocp - - enq u eu e {{path/to/au dio_file}}
  ```

- Add songs recursively to the play q ueue while MO C is in the background:

  ```
  m ocp - - append {{path/to/directory}}
  ```

- Clear the play q ueue while MO C is in the background:

  ```
  m ocp - - clear
  ```

- P lay or stop the currently q ueued song while MO C is in the background:

  ```
  m ocp - - {{play| stop}}
  ```

- S top the MO C server while it' s in the background:

  ```
  m ocp - - exit
  ```

modinfo

E xtract information about a Linux kernel module.

- List all attributes of a kernel module:

  ```
  m odinfo {{k ernel_m odu le}}
  ```

- List the specified attribute only:

  ```
  m odinfo - F {{au thor| description| license| parm | filenam e}}
  {{k ernel_m odu le}}
  ```

modprobe

Add or remove modules from the Linux kernel.

- Pretend to load a module into the kernel, but don't actually do it:

  ```
  sudo modprobe --dry-run {{module_name}}
  ```

- Load a module into the kernel:

  ```
  sudo modprobe {{module_name}}
  ```

- Remove a module from the kernel:

  ```
  sudo modprobe --remove {{module_name}}
  ```

- Remove a module and those that depend on it from the kernel:

  ```
  sudo modprobe --remove-dependencies {{module_name}}
  ```

- Show a kernel module's dependencies:

  ```
  sudo modprobe --show-depends {{module_name}}
  ```

module

Modify a users' environment using the module command.

More information: https://lmod.readthedocs.io/en/latest/0_1_0__user.html

- Display available modules:

  ```
  m odu le av ail
  ```

- S earch for a module by name:

  ```
  m odu le spider {{m odu le_nam e}}
  ```

- Load a module:

  ```
  m odu le load {{m odu le_nam e}}
  ```

- Display loaded modules:

  ```
  m odu le list
  ```

- U nload a specific loaded module:

  ```
  m odu le {{m odu le_nam e}}
  ```

- U nload all loaded modules:

  ```
  m odu le pu rg e
  ```

mono

Runtime for the .NET Framework.

More information: https://www.mono-project.com/docs/

- Run a .NET assembly in debug mode:

```
mono --debug {{path/to/program.exe}}
```

- Run a .NET assembly:

```
mono {{path/to/program.exe}}
```

mons

A tool to quickly manage two displays.

More information: https://github.com/Vento/mons

- Enable only the primary monitor:

  ```
  mons -o
  ```

- Enable only the secondary monitor:

  ```
  mons -s
  ```

- Duplicate the primary monitor onto the secondary monitor, using the resolution of the primary monitor:

  ```
  mons -d
  ```

- Mirror the primary monitor onto the secondary monitor, using the resolution of the secondary monitor:

  ```
  mons -m
  ```

mountpoint

Test if a directory is a filesystem mountpoint.

- Check if a directory is a mountpoint:

  ```
  mountpoint {{path/to/directory}}
  ```

- Check if a directory is a mountpoint without showing any output:

  ```
  mountpoint -q {{path/to/directory}}
  ```

- Show major/minor numbers of a mountpoint's filesystem:

  ```
  mountpoint --fs-devno {{path/to/directory}}
  ```

mpstat

Report CPU statistics.

- Display CPU statistics every 2 seconds:

  ```
  mpstat {{2}}
  ```

- Display 5 reports, one by one, at 2 second intervals:

  ```
  mpstat {{2}} {{5}}
  ```

- Display 5 reports, one by one, from a given processor, at 2 second intervals:

  ```
  mpstat -P {{0}} {{2}} {{5}}
  ```

mssh

G T K+ based S S H client for interacting with multiple S S H servers at once.

- O pen a new window and connect to multiple S S H servers:

  ```
  m ssh {{u ser@ host1 }} {{u ser@ host2 }} {{. . . }}
  ```

- O pen a new window and connect to a group of servers predefined in `~ /. m ssh_clu sters`

  ```
  m ssh - - alias {{alias_nam e}}
  ```

mullvad

CLI client for Mullvad VPN.

More information: https://mullvad.net/.

- Link your mullvad account with the specified account number:

  ```
  mullvad account set {{account_number}}
  ```

- Enable LAN access while VPN is on:

  ```
  mullvad lan set allow
  ```

- Establish the VPN tunnel:

  ```
  mullvad connect
  ```

- Check status of VPN tunnel:

  ```
  mullvad status
  ```

mycli

A CLI for MyS Q L, MariaDB , and P ercona with auto-completion and syntax highlighting.

- Connect to a database with the currently logged in user:

  ```
  m ycli {{datab ase_nam e}}
  ```

- Connect to a database with the specified user:

  ```
  m ycli - u  {{u ser}} {{datab ase_nam e}}
  ```

- Connect to a database on the specified host with the specified user:

  ```
  m ycli - u  {{u ser}} - h {{host}} {{datab ase_nam e}}
  ```

n

Tool to manage multiple node versions.

- Install a given version of node. If the version is already installed, it will be activated:

  ```
  n {{version}}
  ```

- Display installed versions and interactively activate one of them:

  ```
  n
  ```

- Remove a version:

  ```
  n rm {{version}}
  ```

- Execute a file with a given version:

  ```
  n use {{version}} {{file.js}}
  ```

- Output binary path for a version:

  ```
  n bin {{version}}
  ```

named

Execute the DNS (Dynamic Name Service) server daemon that converts host names to IP addresses and vice versa.

More information: https://manned.org/named.

- Read the default configuration file `/etc/named.conf`, read any initial data and listen for queries:

  ```
  named
  ```

- Read a custom configuration file:

  ```
  named -c {{path/to/named.conf}}
  ```

- Use IPv4 or IPv6 only, even if the host machine is capable of utilising other protocols:

  ```
  named {{-4|-6}}
  ```

- Listen for queries on a specific port instead of the default port 53:

  ```
  named -p {{port}}
  ```

- Run the server in the foreground and do not daemonize:

  ```
  named -f
  ```

namei

Follows a pathname (which can be a symbolic link) until a terminal point is found (a file/directory/char device etc).

This program is useful for finding "too many levels of symbolic links" problems.

- Resolve the pathnames specified as the argument parameters:

  ```
  namei {{path/to/a}} {{path/to/b}} {{path/to/c}}
  ```

- Display the results in a long-listing format:

  ```
  namei --long {{path/to/a}} {{path/to/b}} {{path/to/c}}
  ```

- Show the mode bits of each file type in the style of `ls`:

  ```
  namei --modes {{path/to/a}} {{path/to/b}} {{path/to/c}}
  ```

- Show owner and group name of each file:

  ```
  namei --owners {{path/to/a}} {{path/to/b}} {{path/to/c}}
  ```

- Don't follow symlinks while resolving:

  ```
  namei --nosymlinks {{path/to/a}} {{path/to/b}} {{path/to/c}}
  ```

nautilus

Default file explorer for G N O ME desktop environment.

Also known as G N O ME F iles.

More information: https://manned.org/nautilus.

- Launch N autilus:

  ```
  nau tilu s
  ```

- Launch N autilus as root user:

  ```
  su do  nau tilu s
  ```

- Launch N autilus and display a specific directory:

  ```
  nau tilu s  {{path/to/directory}}
  ```

- Launch N autilus with a specific file or directory selected:

  ```
  nau tilu s  - - select {{path/to/file_or_directory}}
  ```

- Launch N autilus in a separated window:

  ```
  nau tilu s  - - new - w indow
  ```

- Close all N autilus instances:

  ```
  nau tilu s  - - q u it
  ```

- Display help:

  ```
  nau tilu s  - - help
  ```

ncat

Use the normal `cat` functionality over networks.

- Listen for input on the specified port and write it to the specified file:

  ```
  ncat -l {{port}} > {{path/to/file}}
  ```

- Accept multiple connections and keep ncat open after they have been closed:

  ```
  ncat -lk {{port}}
  ```

- Write output of specified file to the specified host on the specified port:

  ```
  ncat {{address}} {{port}} < {{path/to/file}}
  ```

ncdu

Disk usage analyzer with an ncurses interface.

- Analyze the current working directory:

  ```
  ncdu
  ```

- Analyze a given directory:

  ```
  ncdu {{path/to/directory}}
  ```

- Save results to a file:

  ```
  ncdu -o {{path/to/file}}
  ```

- Exclude files that match a pattern, argument can be given multiple times to add more patterns:

  ```
  ncdu --exclude '{{*.txt}}'
  ```

ndctl

Utility for managing Non-Volatile DIMMs.

- Create an 'fsdax' mode namespace:

  ```
  ndctl create-namespace --mode={{fsdax}}
  ```

- Change the mode of a namespace to 'raw':

  ```
  ndctl create-namespace --reconfig=re={{namespaceX.Y}} --mode={{raw}}
  ```

- Check a sector mode namespace for consistency, and repair if needed:

  ```
  ndctl check-namespace --repair {{namespaceX.Y}}
  ```

- List all namespaces, regions, and buses (including disabled ones):

  ```
  ndctl list --namespaces --regions --buses --idle
  ```

- List a specific namespace and include lots of additional information:

  ```
  ndctl list -vvv --namespace={{namespaceX.Y}}
  ```

- Run a monitor to watch for SMART health events for NVDIMMs on the 'ACPI.NFIT' bus:

  ```
  ndctl monitor --bus={{ACPI.NFIT}}
  ```

- Remove a namespace (when applicable) or reset it to an initial state:

  ```
  ndctl destroy-namespace --force {{namespaceX.Y}}
  ```

nemo

File manager and graphical shell for Cinnamon.

More information: https://github.com/linuxmint/nemo.

- Open a new window showing the user's home directory:

  ```
  nemo
  ```

- Open a new window showing the current directory:

  ```
  nemo .
  ```

- Close all open nemo windows:

  ```
  nemo --quit
  ```

nethogs

Monitor bandwidth usage per process.

More information: https://github.com/raboof/nethogs.

- Start nethogs as root (default device is eth0):

  ```
  sudo nethogs
  ```

- Monitor bandwidth on specific device:

  ```
  sudo nethogs {{device}}
  ```

- Monitor bandwidth on multiple devices:

  ```
  sudo nethogs {{device1}} {{device2}}
  ```

- Specify refresh rate:

  ```
  sudo nethogs -t {{seconds}}
  ```

netselect-apt

Create a `sources.list` file for a Debian mirror with the lowest latency.

More information: https://manpages.debian.org/buster/netselect-apt/netselect-apt.1.html

- Create `sources.list` using the lowest latency server:

  ```
  sudo netselect-apt
  ```

- Specify Debian branch, stable is used by default:

  ```
  sudo netselect-apt {{testing}}
  ```

- Include non-free section:

  ```
  sudo netselect-apt --non-free
  ```

- Specify a country for the mirror list lookup:

  ```
  sudo netselect-apt -c {{India}}
  ```

netselect

Speed test for choosing a fast network server.

More information: https://github.com/apenwarr/netselect.

- Choose the server with the lowest latency:

```
sudo netselect {{host_1}} {{host_2}}
```

- Display nameserver resolution and statistics:

```
sudo netselect -vv {{host_1}} {{host_2}}
```

- Define maximum TTL (time to live):

```
sudo netselect -m {{10}} {{host_1}} {{host_2}}
```

- Print fastest N servers among the hosts:

```
sudo netselect -s {{N}} {{host_1}} {{host_2}} {{host_3}}
```

- List available options:

```
netselect
```

netstat

Displays network-related information such as open connections, open socket ports, etc.

More information: https://man7.org/linux/man-pages/man8/netstat.8.html

- List all ports:

```
netstat --all
```

- List all listening ports:

```
netstat --listening
```

- List listening TCP ports:

```
netstat --tcp
```

- Display PID and program names:

```
netstat --program
```

- List information continuously:

```
netstat --continuous
```

- List routes and do not resolve IP addresses to hostnames:

```
netstat --route --numeric
```

- List listening TCP and UDP ports (+ user and process if you're root):

```
netstat --listening --program --numeric --tcp --udp --exte
```

newgrp

Switch primary group membership.

- Change user's primary group membership:

  ```
  newgrp {{group_name}}
  ```

- Reset primary group membership to user's default group in `/etc/passwd`

  ```
  newgrp
  ```

nft

Allows configuration of tables, chains and rules provided by the Linux kernel firewall.

N ftables replaces iptables.

- V iew current configuration:

  ```
  su do nft list ru leset
  ```

- Add a new table with family " inet" and table " filter" :

  ```
  su do nft add tab le {{inet}} {{filter}}
  ```

- Add a new chain to accept all inbound traffic:

  ```
  su do nft add chain {{inet}} {{filter}} {{inpu t}} \ { type
  {{filter}} hook {{inpu t}} priority {{0 }} \ ; policy {{accept}}
  \ }
  ```

- Add a new rule to accept several T CP ports:

  ```
  su do nft add ru le {{inet}} {{filter}} {{inpu t}} {{tcp}}
  {{dport \ { telnet, ssh, http, https \ } accept}}
  ```

- S how rule handles:

  ```
  su do nft - - handle - - nu m eric list chain {{fam ily}} {{tab le}}
  {{chain}}
  ```

- Delete a rule:

  ```
  su do nft delete ru le {{inet}} {{filter}} {{inpu t}} handle
  {{3 }}
  ```

- S ave current configuration:

  ```
  su do nft list ru leset > {{/etc/nftab les. conf}}
  ```

nixos-rebuild

Reconfigure a NixOS machine.

More information: https://nixos.org/nixos/manual/#sec-changing-config

- Build and switch to the new configuration, making it the boot default:

  ```
  sudo nixos-rebuild switch
  ```

- Build and switch to the new configuration, making it the boot default and naming the boot entry:

  ```
  sudo nixos-rebuild switch -p {{name}}
  ```

- Build and switch to the new configuration, making it the boot default and installing updates:

  ```
  sudo nixos-rebuild switch --upgrade
  ```

- Rollback changes to the configuration, switching to the previous generation:

  ```
  sudo nixos-rebuild switch --rollback
  ```

- Build the new configuration and make it the boot default without switching to it:

  ```
  sudo nixos-rebuild boot
  ```

- Build and activate the new configuration, but don't make a boot entry (for testing purposes):

  ```
  sudo nixos-rebuild test
  ```

- Build the configuration and open it in a virtual machine:

  ```
  sudo nixos-rebuild build-vm
  ```

nm

List symbol names in object files.

- List global (extern) functions in a file (prefixed with T):

```
nm -g {{file.o}}
```

- List only undefined symbols in a file:

```
nm -u {{file.o}}
```

- List all symbols, even debugging symbols:

```
nm -a {{file.o}}
```

- Demangle C++ symbols (make them readable):

```
nm --demangle {{file.o}}
```

nmcli connection

Connection management with NetworkManager.

More information: https://man.archlinux.org/man/nmcli.1.

- List all NetworkManager connections (shows name, uuid, type and device):

```
nmcli connection
```

- Activate a connection by specifying an uuid:

```
nmcli connection up uuid {{uuid}}
```

- Deactivate a connection:

```
nmcli connection down uuid {{uuid}}
```

- Create an auto-configured dual stack connection:

```
nmcli connection add ifname {{interface_name}} type
{{ethernet}} ipv4.method {{auto}} ipv6.method {{auto}}
```

- Create a static IPv6-only connection:

```
nmcli connection add ifname {{interface_name}} type
{{ethernet}} ip6 {{2001:db8::2/64}} gw6 {{2001:db8::1
ipv6.dns {{2001:db8::1}} ipv4.method {{ignore}}
```

- Create a static IPv4-only connection:

```
nmcli connection add ifname {{interface_name}} type
{{ethernet}} ip4 {{10.0.0.7/8}} gw4 {{10.0.0.1}} ipv4.
{{10.0.0.1}} ipv6.method {{ignore}}
```

- Create a VPN connection using OpenVPN from a OVPN file:

```
nmcli connection import type {{openvpn}} file {{path/to/
vpn_config.ovpn}}
```

nmcli device

Hardware device management with NetworkManager.

More information: https://man.archlinux.org/man/nmcli.1.

- Print the statuses of all network interfaces:

  ```
  nmcli device status
  ```

- Print the available Wi-Fi access points:

  ```
  nmcli device wifi
  ```

- Connect to the Wi-Fi network with a specified name and password:

  ```
  nmcli device wifi connect {{ssid}} password {{password}}
  ```

- Print password and QR code for the current Wi-Fi network:

  ```
  nmcli device wifi show-password
  ```

nmcli monitor

Monitor changes to the N etworkManager connection status.

More information: https://developer.gnome.org/N etworkManager/stable/nmcli.html

- S tart monitoring N etworkManager changes:

  ```
  nm cli m onitor
  ```

nmcli

A command-line tool for controlling N etworkManager.

More information: https://manned.org/nmcli.

- R un an`nm cli`subcommand:

  ```
  nm cli {{ag ent| connection| dev ice| g eneral| help| m onitor|
  netw ork ing | radio}} {{com m and_options}}
  ```

- Display the current version of N etworkManager:

  ```
  nm cli - - v ersion
  ```

- Display help:

  ```
  nm cli - - help
  ```

- Display help for a subcommand:

  ```
  nm cli {{su b com m and}} - - help
  ```

nmon

A system administrator, tuner, and benchmark tool.

- Start nmon:

  ```
  nmon
  ```

- Save records to file ("-s 300 -c 288" by default):

  ```
  nmon -f
  ```

- Save records to file with a total of 240 measurements, by taking 30 seconds between each measurement:

  ```
  nmon -f -s {{30}} -c {{240}}
  ```

nmtui

Text user interface for controlling NetworkManager.

Use arrow keys to navigate, enter to select an option.

- Open the user interface:

  ```
  nmtui
  ```

- Show a list of available connections, with the option to activate or deactivate them:

  ```
  nmtui connect
  ```

- Connect to a given network:

  ```
  nmtui connect {{name|uuid|device|SSID}}
  ```

- Edit/Add/Delete a given network:

  ```
  nmtui edit {{name|id}}
  ```

- Set the system hostname:

  ```
  nmtui hostname
  ```

nologin

Alternative shell that prevents a user from logging in.

- Set a user's login shell to `nologin` to prevent the user from logging in:

  ```
  chsh -s {{user}} nologin
  ```

- Customize message for users with the login shell of `nologin`:

  ```
  echo "{{declined_login_message}}" > /etc/nologin.txt
  ```

notify-send

Uses the current desktop environment's notification system to create a notification.

- Show a notification with the title "Test" and the content "This is a test":

```
notify-send "{{Test}}" "{{This is a test}}"
```

- Show a notification with a custom icon:

```
notify-send -i {{icon.png}} "{{Test}}" "{{This is a test}}"
```

- Show a notification for 5 seconds:

```
notify-send -t 5000 "{{Test}}" "{{This is a test}}"
```

- Show a notification with an app's icon:

```
notify-send "{{Test}}" --icon={{google-chrome}}
```

nsenter

Run a new command in a running process' namespace.

Particularly useful for docker images or chroot jails.

More information: https://github.com/jpetazzo/nsenter/

- Run command in existing processes network namespace:

```
nsenter -t {{pid}} -n {{command}} {{command_arguments}}
```

- Run a new command in an existing processes ps-table namespace:

```
nsenter -t {{pid}} -p {{command}} {{command_arguments}}
```

- Run command in existing processes IPC namespace:

```
nsenter -t {{pid}} -i {{command}} {{command_arguments}}
```

nsnake

Snake game in the terminal.

More information: https://github.com/alexdantas/nsnake/.

- Start a snake game:

  ```
  nsnake
  ```

- Navigate the snake:

  ```
  {{Up|Down|Left|Right}} arrow key
  ```

- Pause/unpause the game:

  ```
  p
  ```

- Quit the game:

  ```
  q
  ```

- Show help during the game:

  ```
  h
  ```

ntfsfix

Fix common problems on an NTFS partition.

- Fix a given NTFS partition:

```
sudo ntfsfix {{/dev/sdXN}}
```

ntpq

Query the Network Time Protocol (NTP) daemon.

More information: https://www.eecis.udel.edu/~mills/ntp/html/ntpq.html

- Start `ntpq` in interactive mode:

  ```
  ntpq --interactive
  ```

- Print a list of NTP peers:

  ```
  ntpq --peers
  ```

- Print a list of NTP peers without resolving hostnames from IP addresses:

  ```
  ntpq --numeric --peers
  ```

- Use `ntpq` in debugging mode:

  ```
  ntpq --debug-level
  ```

- Print NTP system variables values:

  ```
  ntpq --command={{rv}}
  ```

numactl

Control NUMA policy for processes or shared memory.

More information: https://man7.org/linux/man-pages/man8/numactl.8.html

- Run a command on node 0 with memory allocated on node 0 and 1:

```
numactl --cpunodebind={{0}} --membind={{0,1}} --{{command arguments}}
```

- Run a command on CPUs (cores) 0-4 and 8-12 of the current cpuset:

```
numactl --physcpubind={{+0-4,8-12}} --{{command}} {{command_arguments}}
```

- Run a command with its memory interleaved on all CPUs:

```
numactl --interleave={{all}} --{{command}} {{command_arguments}}
```

numlockx

Control the number lock key status in X 1 1 sessions.

More information: http://www.mike-devlin.com/linux/R E ADME -numlockx.htm

- S how the current number lock status:

  ```
  nu m lock x statu s
  ```

- T urn the number lock on:

  ```
  nu m lock x on
  ```

- T urn the number lock off:

  ```
  nu m lock x off
  ```

- T oggle the current state:

  ```
  nu m lock x tog g le
  ```

openfortivpn

A VPN client, for Fortinet's proprietary PPP+SSL VPN solution.

More information: https://github.com/adrienverge/openfortivpn.

- Connect to a VPN with a username and password:

  ```
  openfortivpn --username={{username}} --password={{password}}
  ```

- Connect to a VPN using a specific configuration file (defaults to `/etc/openfortivpn/config`):

  ```
  sudo openfortivpn --config={{path/to/config}}
  ```

- Connect to a VPN by specifying the host and port:

  ```
  openfortivpn {{host}}:{{port}}
  ```

- Trust a given gateway by passing in its certificate's sha256 sum:

  ```
  openfortivpn --trusted-cert={{sha256_sum}}
  ```

openrc

The OpenRC service manager.

See also **rc-status**, **rc-update**, and **rc-service**.

More information: https://wiki.gentoo.org/wiki/OpenRC

- Change to a specific runlevel:

  `sudo openrc {{runlevel_name}}`

- Change to a specific runlevel, but don't stop any existing services:

  `sudo openrc --no-stop {{runlevel_name}}`

openvpn3

OpenVPN 3 Linux client.

More information: https://community.openvpn.net/openvpn/wiki/OpenVPN3Linux

- Start a new VPN session:

  ```
  openvpn3 session-start --config {{path/to/config.conf}}
  ```

- List established sessions:

  ```
  openvpn3 sessions-list
  ```

- Disconnect the currently established session started with given configuration:

  ```
  openvpn3 session-manage --config {{path/to/config.conf}} --disconnect
  ```

- Import VPN configuration:

  ```
  openvpn3 config-import --config {{path/to/config.conf}}
  ```

- List imported configurations:

  ```
  openvpn3 configs-list
  ```

opkg

A lightweight package manager used to install O penWrt packages.

- Install a package:

  ```
  opk g  install {{pack ag e}}
  ```

- R emove a package:

  ```
  opk g  rem ov e {{pack ag e}}
  ```

- U pdate the list of available packages:

  ```
  opk g  u pdate
  ```

- U pgrade all the installed packages:

  ```
  opk g  u pg rade
  ```

- U pgrade one or more specific package(s):

  ```
  opk g  u pg rade {{pack ag e( s) }}
  ```

- Display information for a specific package:

  ```
  opk g  info {{pack ag e}}
  ```

- List all the available packages:

  ```
  opk g  list
  ```

pacaur

A utility for Arch Linux to build and install packages from the Arch User Repository.

- Synchronize and update all packages (includes AUR):

  ```
  pacaur -Syu
  ```

- Synchronize and update only AUR packages:

  ```
  pacaur -Syua
  ```

- Install a new package (includes AUR):

  ```
  pacaur -S {{package_name}}
  ```

- Remove a package and its dependencies (includes AUR packages):

  ```
  pacaur -Rs {{package_name}}
  ```

- Search the package database for a keyword (includes AUR):

  ```
  pacaur -Ss {{keyword}}
  ```

- List all currently installed packages (includes AUR packages):

  ```
  pacaur -Qs
  ```

paccache

A pacman cache cleaning utility.

- Remove all but the 3 most recent package versions from the pacman cache:

  ```
  paccache -r
  ```

- Set the number of package versions to keep:

  ```
  paccache -rk {{num_versions}}
  ```

- Perform a dry-run and show the number of candidate packages for deletion:

  ```
  paccache -d
  ```

- Move candidate packages to a directory instead of deleting them:

  ```
  paccache -m {{path/to/directory}}
  ```

pacman --database

O perate on the Arch Linux package database.

Modify certain attributes of the installed packages.

More information: https://man.archlinux.org/man/pacman.8.

- Display help:

```
pacman --database --help
```

- Mark a package as implicitly installed:

```
sudo pacman --database --asdeps {{package_name}}
```

- Mark a package as explicitly installed:

```
sudo pacman --database --asexplicit {{package_name}}
```

- Check that all the package dependencies are installed:

```
pacman --database --check
```

- Check the repositories to ensure all specified dependencies are available:

```
pacman --database --check --check
```

- Display only error messages:

```
pacman --database --check --quiet
```

pacman --deptest

Check each dependency specified and return a list of dependencies that are not currently satisfied on the system.

More information: https://man.archlinux.org/man/pacman.8.

- Print the package names of the dependencies that aren't installed:

```
pacman --deptest {{package_name1}} {{package_name2}}
```

- Check if the installed package satisfies the given minimum version:

```
pacman --deptest "{{bash>=5}}"
```

- Check if a later version of a package is installed:

```
pacman --deptest "{{bash>5}}"
```

- Display help:

```
pacman --deptest --help
```

pacman --files

Arch Linux package manager utility.

See also: `pkgfile`.

More information: https://man.archlinux.org/man/pacman.8.

- Display help:

  `pacman --files --help`

- Update the package database:

  `sudo pacman --files --refresh`

- Find the package that owns a specific file:

  `pacman --files {{filename}}`

- Find the package that owns a specific file, using a regular expression:

  `pacman --files --regex '{{regular_expression}}'`

- List only the package names:

  `pacman --files --quiet {{filename}}`

- List the files owned by a specific package:

  `pacman --files --list {{package_name}}`

- List only the absolute path to the files:

  `pacman --query --list --quiet {{package_name}}`

pacman-mirrors

Generate a pacman mirrorlist for Manjaro Linux.

Every run of pacman-mirrors requires you to synchronize your database and update your system using `sudo pacman -Syyu`

More information: https://wiki.manjaro.org/index.php?title=Pacman-mirrors

- Generate a mirrorlist using the default settings:

  ```
  sudo pacman-mirrors --fasttrack
  ```

- Get the status of the current mirrors:

  ```
  pacman-mirrors --status
  ```

- Display the current branch:

  ```
  pacman-mirrors --get-branch
  ```

- Switch to a different branch:

  ```
  sudo pacman-mirrors --api --set-branch {{stable|unstable|testing}}
  ```

- Generate a mirrorlist, only using mirrors in your country:

  ```
  sudo pacman-mirrors --geoip
  ```

pacman --q uery

Arch Linux package manager utility.

More information: https://man.archlinux.org/man/pacman.8.

- List installed packages and versions:

```
pacman --query
```

- List only packages and versions that were explicitly installed:

```
pacman --query --explicit
```

- Find which package owns a file:

```
pacman --query --owns {{filename}}
```

- Display information about an installed package:

```
pacman --query --info {{package_name}}
```

- List files owned by a package:

```
pacman --query --list {{package_name}}
```

- List orphan packages (installed as dependencies but not required by any package):

```
pacman --query --unrequired --deps --quiet
```

- List installed packages not found in the repositories:

```
pacman --query --foreign
```

- List outdated packages:

```
pacman --query --upgrades
```

pacman --remove

Arch Linux package manager utility.

More information: https://man.archlinux.org/man/pacman.8.

- Display help for this subcommand:

```
pacman --remove --help
```

- Remove a package and its dependencies:

```
sudo pacman --remove --recursive {{package_name}}
```

- Remove a package and both its dependencies and configuration files:

```
sudo pacman --remove --recursive --nosave {{package_name}}
```

- Remove a package without prompting:

```
sudo pacman --remove --noconfirm {{package_name}}
```

- Remove orphan packages (installed as dependencies but not required by any package):

```
sudo pacman --remove --recursive --nosave $(pacman --query --unrequired --deps --quiet)
```

- Remove a package and all packages that depend on it:

```
sudo pacman --remove --cascade {{package_name}}
```

- List packages that would be affected (does not remove any packages):

```
pacman --remove --print {{package_name}}
```

pacman --sync

Arch Linux package manager utility.

More information: https://man.archlinux.org/man/pacman.8.

- Install a new package:

```
sudo pacman --sync {{package name}}
```

- Synchronize and update all packages (add `--downloadonly` to download the packages and not update them):

```
sudo pacman --sync --refresh --sysupgrade
```

- Update all packages and install a new one without prompting:

```
sudo pacman --sync --refresh --sysupgrade --noconfirm {{package name}}
```

- Search the package database for a regular expression or keyword:

```
pacman --sync --search "{{search_pattern}}"
```

- Display information about a package:

```
pacman --sync --info {{package name}}
```

- Overwrite conflicting files during a package update:

```
sudo pacman --sync --refresh --sysupgrade --overwrite {{path/to/file}}
```

- Synchronize and update all packages, but ignore a specific package (can be used more than once):

```
sudo pacman --sync --refresh --sysupgrade --ignore {{package name}}
```

- Remove not installed packages and unused repositories from the cache (use two `--clean` flags to clean all packages):

```
sudo pacman --sync --clean
```

pacman --upgrade

Arch Linux package manager utility.

More information: https://man.archlinux.org/man/pacman.8.

- Display help:

```
pacman --upgrade --help
```

- Install one or more packages from files:

```
sudo pacman --upgrade {{path/to/package1.pkg.tar.zst}} {{path/to/package2.pkg.tar.zst}}
```

- Install a package without prompting:

```
sudo pacman --upgrade --noconfirm {{path/to/package.pkg.tar.zst}}
```

- Overwrite conflicting files during a package installation:

```
sudo pacman --upgrade --overwrite {{path/to/file}} {{path/to/package.pkg.tar.zst}}
```

- Install a package, skipping the dependency version checks:

```
sudo pacman --upgrade --nodeps {{path/to/package.pkg.tar.zst}}
```

- List packages that would be affected (does not install any packages):

```
pacman --query --print {{path/to/package.pkg.tar.zst}}
```

pacman

Arch Linux package manager utility.

More information: https://man.archlinux.org/man/pacman.8.

- S ynchroniz e and update all packages:

  ```
  pacm an - - sync - - refresh - - sysu pg rade
  ```

- Install a new package:

  ```
  pacm an - - sync {{pack ag e_nam e}}
  ```

- R emove a package and its dependencies:

  ```
  pacm an - - rem ov e - - recu rsiv e {{pack ag e_nam e}}
  ```

- S earch the package database for a regular expression or keyword:

  ```
  pacm an - - sync - - search " {{search_pattern}}"
  ```

- List installed packages and versions:

  ```
  pacm an - - q u ery
  ```

- List only the explicitly installed packages and versions:

  ```
  pacm an - - q u ery - - explicit
  ```

- List orphan packages (installed as dependencies but not actually req uired by any package):

  ```
  pacm an - - q u ery - - u nreq u ired - - deps - - q u iet
  ```

- E mpty the entire pacman cache:

  ```
  pacm an - - sync - - clean - - clean
  ```

pacman4_console

A text-based console game inspired by the original Pacman.

More information: https://github.com/YoctoForBeaglebone/pacman4_console

- Start a game at Level 1:

  ```
  pacman4console
  ```

- Start a game on a certain level (there are nine official levels):

  ```
  pacman4console --level={{level_number}}
  ```

- Start the pacman4console level editor, saving to a specified text file:

  ```
  pacman4consoleedit {{path/to/level_file}}
  ```

- Play a custom level:

  ```
  pacman4console --level={{path/to/level_file}}
  ```

pacstrap

Arch Linux install script to install packages to the specified new root directory.

More information: https://man.archlinux.org/man/pacstrap.8.

- Install the `base` package, Linux kernel and firmware for common hardware:

  ```
  pacstrap {{path/to/new/root}} {{base}} {{linux}} {{linux-firmware}}
  ```

- Install the `base` package, Linux LTS kernel and `base-devel` build tools:

  ```
  pacstrap {{path/to/new/root}} {{base}} {{base-devel}} {{linux-lts}}
  ```

- Install packages without copy the host's mirrorlist to the target:

  ```
  pacstrap -M {{path/to/new/root}} {{packages}}
  ```

- Use an alternate configuration file for Pacman:

  ```
  pacstrap -C {{path/to/pacman.conf}} {{path/to/new/root}} {{packages}}
  ```

- Install packages using the package cache on the host instead of on the target:

  ```
  pacstrap -c {{path/to/new/root}} {{packages}}
  ```

- Install packages without copy the host's pacman keyring to the target:

  ```
  pacstrap -G {{path/to/new/root}} {{packages}}
  ```

- Install packages in interactive mode (prompts for confirmation):

  ```
  pacstrap -i {{path/to/new/root}} {{packages}}
  ```

- Install packages using package files:

  ```
  pacstrap -U {{path/to/new/root}} {{path/to/package1}} {{path/to/package2}}
  ```

pamac

A command-line utility for the G U I package manager pamac.

If you can' t see the AU R packages, enable it in `/et c/ pa ma c. con f` in the G U I.

More information: https://wiki.manj aro.org/index.php/P amac

- Install a new package:

  ```
  pam ac install {{pack ag e_nam e}}
  ```

- R emove a package and its no longer req uired dependencies (orphans):

  ```
  pam ac rem ov e - - orphans {{pack ag e_nam e}}
  ```

- S earch the package database for a package:

  ```
  pam ac search {{pack ag e_nam e}}
  ```

- List installed packages:

  ```
  pam ac list - - installed
  ```

- Check for package updates:

  ```
  pam ac check u pdates
  ```

- U pgrade all packages:

  ```
  pam ac u pg rade
  ```

parted

A partition manipulation program.

See also: `partprobe`.

More information: https://www.gnu.org/software/parted/parted.html.

- List partitions on all block devices:

  ```
  sudo parted --list
  ```

- Start interactive mode with the specified disk selected:

  ```
  sudo parted {{/dev/sdX}}
  ```

- Create a new partition table of the specified label-type:

  ```
  sudo parted --script {{/dev/sdX}} mklabel {{aix|amiga|bsd|dvh|gpt|loop|msdos|pc98|sun}}
  ```

- Show partition information in interactive mode:

  ```
  print
  ```

- Select a disk in interactive mode:

  ```
  select {{/dev/sdX}}
  ```

- Create a 16GB partition with the specified filesystem in interactive mode:

  ```
  mkpart {{primary|logical|extended}} {{btrfs|ext2|ext3|ext4|fat16|fat32|hfs|hfs+|linux-swap|ntfs|reiserfs|udf|xfs}} {{0%}} {{16G}}
  ```

- Resize a partition in interactive mode:

  ```
  resizepart {{/dev/sdXN}} {{end_position_of_partition}}
  ```

- Remove a partition in interactive mode:

  ```
  rm {{/dev/sdXN}}
  ```

partprobe

Notify the operating system kernel of partition table changes.

More information: https://manned.org/partprobe.

- Notify the operating system kernel of partition table changes:

  ```
  sudo partprobe
  ```

- Notify the kernel of partition table changes and show a summary of devices and their partitions:

  ```
  sudo partprobe --summary
  ```

- Show a summary of devices and their partitions but don't notify the kernel:

  ```
  sudo partprobe --summary --dry-run
  ```

partx

Parse a partition table and tell the kernel about it.

More information: https://man7.org/linux/man-pages/man8/partx.8.html

- List the partitions on a block device or disk image:

```
sudo partx --list {{path/to/device_or_disk_image}}
```

- Add all the partitions found in a given block device to the kernel:

```
sudo partx --add --verbose {{path/to/device_or_disk_image}}
```

- Delete all the partitions found from the kernel (does not alter partitions on disk):

```
sudo partx --delete {{path/to/device_or_disk_image}}
```

paru

An AUR helper and pacman wrapper.

More information: https://github.com/Morganamilo/paru.

- Interactively search for and install a package:

  ```
  paru {{package_name_or_search_term}}
  ```

- Synchronize and update all packages:

  ```
  paru
  ```

- Upgrade AUR packages:

  ```
  paru -Sua
  ```

- Get information about a package:

  ```
  paru -Si {{package_name}}
  ```

- Download PKGBUILD and other package source files from the AUR or ABS:

  ```
  paru --getpkgbuild {{package_name}}
  ```

- Display the PKGBUILD file of a package:

  ```
  paru --getpkgbuild --print {{package_name}}
  ```

pasuspender

Temporarily suspends **pulseaudio** while another command is running to allow access to alsa.

- Suspend pulseaudio while running `jackd`:

  ```
  pasuspender -- {{jackd -d alsa --device hw:0}}
  ```

pdfgrep

Search text in PDF files.

- Find lines that match pattern in a PDF:

  ```
  pdfgrep {{pattern}} {{file.pdf}}
  ```

- Include file name and page number for each matched line:

  ```
  pdfgrep --with-filename --page-number {{pattern}} {{file.pdf}}
  ```

- Do a case insensitive search for lines that begin with "foo" and return the first 3 matches:

  ```
  pdfgrep --max-count {{3}} --ignore-case {{'^foo'}} {{file.pdf}}
  ```

- Find pattern in files with a `.pdf` extension in the current directory recursively:

  ```
  pdfgrep --recursive {{pattern}}
  ```

- Find pattern on files that match a specific glob in the current directory recursively:

  ```
  pdfgrep --recursive --include {{'*book.pdf'}} {{pattern}}
  ```

perf

Framework for linux performance counter measurements.

- Display basic performance counter stats for a command:

  `perf stat {{gcc hello.c}}`

- Display system-wide real time performance counter profile:

  `sudo perf top`

- Run a command and record its profile into `perf.data`

  `sudo perf record {{command}}`

- Read `perf.data` (created by `perf record`) and display the profile:

  `sudo perf report`

rename

Rename multiple files.

NOTE: this page refers to the command from the `perl-rename` Arch Linux package.

- Rename files using a Perl Common Regular Expression (substitute 'foo' with 'bar' wherever found):

  ```
  rename {{'s/foo/bar/'}} {{*}}
  ```

- Dry-run - display which renames would occur without performing them:

  ```
  rename -n {{'s/foo/bar/'}} {{*}}
  ```

- Force renaming even if the operation would remove existing destination files:

  ```
  rename -f {{'s/foo/bar/'}} {{*}}
  ```

- Convert filenames to lower case (use `-f` in case-insensitive filesystems to prevent "already exists" errors):

  ```
  rename 'y/A-Z/a-z/' {{*}}
  ```

- Replace whitespace with underscores:

  ```
  rename 's/\s+/_/g' {{*}}
  ```

phar

Create, update or extract PHP archives (PHAR).

- Add space-separated files or directories to a Phar file:

```
phar add -f {{path/to/phar_file}} {{files_or_directories}}
```

- Display the contents of a Phar file:

```
phar list -f {{path/to/phar_file}}
```

- Delete the specified file or directory from a Phar file:

```
phar delete -f {{path/to/phar_file}} -e {{file_or_directory}}
```

- Display full usage information and available hashing/compression algorithms:

```
phar help
```

- Compress or uncompress files and directories in a Phar file:

```
phar compress -f {{path/to/phar_file}} -c {{algorithm}}
```

- Get information about a Phar file:

```
phar info -f {{path/to/phar_file}}
```

- Sign a Phar file with a specific hash algorithm:

```
phar sign -f {{path/to/phar_file}} -h {{algorithm}}
```

- Sign a Phar file with an OpenSSL private key:

```
phar sign -f {{path/to/phar_file}} -h openssl -y {{path/to/
private_key}}
```

photorec

Deleted file recovery tool.

It is recommended to write recovered files to a disk separate to the one being recovered from.

More information: https://www.cgsecurity.org/wiki/PhotoRec.

- Run PhotoRec on a specific device:

  ```
  sudo photorec {{/dev/sdb}}
  ```

- Run PhotoRec on a disk image (`image.dd`):

  ```
  sudo photorec {{path/to/image.dd}}
  ```

phpdismod

Disable PHP extensions on Debian-based OSes.

- Disable the json extension for every SAPI of every PHP version:

  ```
  sudo phpdismod {{json}}
  ```

- Disable the json extension for PHP 7.3 with the cli SAPI:

  ```
  sudo phpdismod -v {{7.3}} -s {{cli}} {{json}}
  ```

phpenmod

Enable PHP extensions on Debian-based OSes.

- Enable the json extension for every SAPI of every PHP version:

  ```
  sudo phpenmod {{json}}
  ```

- Enable the json extension for PHP 7.3 with the cli SAPI:

  ```
  sudo phpenmod -v {{7.3}} -s {{cli}} {{json}}
  ```

phpquery

PHP extension manager for Debian-based OSes.

- List available PHP versions:

  ```
  sudo phpquery -V
  ```

- List available SAPIs for PHP 7.3:

  ```
  sudo phpquery -v {{7.3}} -S
  ```

- List enabled extensions for PHP 7.3 with the cli SAPI:

  ```
  sudo phpquery -v {{7.3}} -s {{cli}} -M
  ```

- Check if the json extension is enabled for PHP 7.3 with the apache2 SAPI:

  ```
  sudo phpquery -v {{7.3}} -s {{apache2}} -m {{json}}
  ```

physlock

Lock all consoles and virtual terminals.

More information: http://github.com/muennich/physlock.

- Lock every console (req uire current user or root to unlock):

  ```
  physlock
  ```

- Mute kernel messages on console while locked:

  ```
  physlock -m
  ```

- Disable S ysR q mechanism while locked:

  ```
  physlock -s
  ```

- Display a message before the password prompt:

  ```
  physlock -p "{{L ock ed! }}"
  ```

- F ork and detach physlock (useful for suspend or hibernate scripts):

  ```
  physlock -d
  ```

pi

Compute decimal Archimedes' constant Pi on the command-line.

More information: http://manpages.ubuntu.com/manpages/trusty/man1/pi.1.html

- Display 100 decimal digits of Archimedes' constant Pi:

  ```
  pi
  ```

- Display a specified number of decimal digits of Archimedes' constant Pi:

  ```
  pi {{number}}
  ```

- Display help:

  ```
  pi --help
  ```

- Display version:

  ```
  pi --version
  ```

- Display recommended readings:

  ```
  pi --bibliography
  ```

pidof

G ets the ID of a process using its name.

- List all process IDs with given name:

  ```
  pidof {{b ash}}
  ```

- List a single process ID with given name:

  ```
  pidof - s {{b ash}}
  ```

- List process IDs including scripts with given name:

  ```
  pidof - x {{script. py}}
  ```

- Kill all processes with given name:

  ```
  k ill " $ ( pidof {{nam e}}) "
  ```

pidstat

Show system resource usage, including CPU, memory, IO etc.

- Show CPU statistics at a 2 second interval for 10 times:

  ```
  pidstat {{2}} {{10}}
  ```

- Show page faults and memory utilization:

  ```
  pidstat -r
  ```

- Show input/output usage per process id:

  ```
  pidstat -d
  ```

- Show information on a specific PID:

  ```
  pidstat -p {{PID}}
  ```

- Show memory statistics for all processes whose command name include "fox" or "bird":

  ```
  pidstat -C "{{fox|bird}}" -r -p ALL
  ```

pihole

Terminal interface for the Pi-hole ad-blocking DNS server.

More information: https://pi-hole.net.

- Check the Pi-hole daemon's status:

  ```
  pihole status
  ```

- Update Pi-hole:

  ```
  pihole updatePihole
  ```

- Monitor detailed system status:

  ```
  pihole chronometer
  ```

- Start or stop the daemon:

  ```
  pihole {{enable|disable}}
  ```

- Restart the daemon (not the server itself):

  ```
  pihole restartdns
  ```

- Whitelist or blacklist a domain:

  ```
  pihole {{whitelist|blacklist}} {{example.com}}
  ```

- Search the lists for a domain:

  ```
  pihole query {{example.com}}
  ```

- Open a real-time log of connections:

  ```
  pihole tail
  ```

pivpn

Easy security-hardened OpenVPN setup and manager.

Originally designed for the Raspberry Pi, but works on other Linux devices too.

More information: http://www.pivpn.io/.

- Add a new client device:

  ```
  sudo pivpn add
  ```

- List all client devices:

  ```
  sudo pivpn list
  ```

- List currently connected devices and their statistics:

  ```
  sudo pivpn clients
  ```

- Revoke a previously authenticated device:

  ```
  sudo pivpn revoke
  ```

- Uninstall PiVPN:

  ```
  sudo pivpn uninstall
  ```

pkg-config

Provide the details of installed libraries for compiling applications.

More information: https://www.freedesktop.org/wiki/Software/pkg-config/

- Get the list of libraries and their dependencies:

```
pkg-config --libs {{library1 library2 ...}}
```

- Get the list of libraries, their dependencies, and proper cflags for gcc:

```
pkg-config --cflags --libs {{library1 library2 ...}}
```

- Compile your code with libgtk-3, libwebkit2gtk-4.0 and all their dependencies:

```
c++ example.cpp $(pkg-config --cflags --libs gtk+-3.0 webkit2gtk-4.0) -o example
```

pkgadd

Add a package to a CRUX system.

- Install a local software package:

  ```
  pkg add {{package_name}}
  ```

- Update an already installed package from a local package:

  ```
  pkg add -u {{package_name}}
  ```

pkgfile

Tool for searching files from packages in the official repositories on arch-based systems.

See also `pacman -Fl`, describing the usage of `pacman --files`.

More information: https://man.archlinux.org/man/extra/pkgfile/pkgfile.1.

- Synchronize the pkgfile database:

  ```
  sudo pkgfile --update
  ```

- Search for a package that owns a specific file:

  ```
  pkgfile {{filename}}
  ```

- List all files provided by a package:

  ```
  pkgfile --list {{package_name}}
  ```

- List only files provided by a package located within the `bin` or `sbin` directory:

  ```
  pkgfile --list --binaries {{package_name}}
  ```

- Search for a package that owns a specific file using case insensitive matching:

  ```
  pkgfile --ignorecase {{filename}}
  ```

- Search for a package that owns a specific file in the `bin` or `sbin` directory:

  ```
  pkgfile --binaries {{filename}}
  ```

- Search for a package that owns a specific file, displaying the package version:

  ```
  pkgfile --verbose {{filename}}
  ```

- Search for a package that owns a specific file in a specific repository:

  ```
  pkgfile --repo {{repository_name}} {{filename}}
  ```

pkginfo

Query the package database on a CRUX system.

- List installed packages and their versions:

  ```
  pkg info -i
  ```

- List files owned by a package:

  ```
  pkg info -l {{package name}}
  ```

- List the owner(s) of files matching a pattern:

  ```
  pkg info -o {{pattern}}
  ```

- Print the footprint of a file:

  ```
  pkg info -f {{file}}
  ```

pkgmk

Make a binary package for use with pkgadd on CRUX.

- Make and download a package:

  ```
  pkgmk -d
  ```

- Install the package after making it:

  ```
  pkgmk -d -i
  ```

- Upgrade the package after making it:

  ```
  pkgmk -d -u
  ```

- Ignore the footprint when making a package:

  ```
  pkgmk -d -if
  ```

- Ignore the MD5 sum when making a package:

  ```
  pkgmk -d -im
  ```

- Update the package's footprint:

  ```
  pkgmk -uf
  ```

pkgrm

Remove a package from a CRUX system.

- Remove an installed package:

```
pkg rm {{package_name}}
```

playerctl

Utility to control different media players.

More information: https://github.com/altdesktop/playerctl.

- Toggle play:

  ```
  playerctl play-pause
  ```

- Next media:

  ```
  playerctl next
  ```

- Previous media:

  ```
  playerctl previous
  ```

- List all players:

  ```
  playerctl --list-all
  ```

- Send a command to a specific player:

  ```
  playerctl --player={{player_name}} {{command}}
  ```

- Send a command to all players:

  ```
  playerctl --all-players {{command}}
  ```

- Show now playing:

  ```
  playerctl metadata --format "Now playing: {{artist}} - {{album}} - {{title}}"
  ```

pmount

Mount arbitrary hotpluggable devices as a normal user.

More information: https://manned.org/pmount.

- Mount a device below `/media/`(using device as mount point):

  ```
  pmount {{/dev/to/block/device}}
  ```

- Mount a device with a specific filesystem type to `/media/label`:

  ```
  pmount --type {{filesystem}} {{/dev/to/block/device}} {{label}}
  ```

- Mount a CD-ROM (filesystem type ISO9660) in read-only mode:

  ```
  pmount --type {{iso9660}} --read-only {{/dev/cdrom}}
  ```

- Mount an NTFS-formatted disk, forcing read-write access:

  ```
  pmount --type {{ntfs}} --read-write {{/dev/sdX}}
  ```

- Display all mounted removable devices:

  ```
  pmount
  ```

ports

Update/list the ports tree on a CRUX system.

- Update the ports tree:

  ```
  ports -u
  ```

- List the ports in the current tree:

  ```
  ports -l
  ```

- Check the differences between installed packages and the ports tree:

  ```
  ports -d
  ```

postfix

Postfix mail transfer agent (MTA) control program.

See also **`dovecot`**, a mail delivery agent (MDA) that integrates with Postfix.

More information: http://www.postfix.org.

- Check the configuration:

  ```
  sudo postfix check
  ```

- Check the status of the Postfix daemon:

  ```
  sudo postfix status
  ```

- Start Postfix:

  ```
  sudo postfix start
  ```

- Gracefully stop Postfix:

  ```
  sudo postfix stop
  ```

- Flush the mail queue:

  ```
  sudo postfix flush
  ```

- Reload the configuration files:

  ```
  sudo postfix reload
  ```

poweroff

Power off the system.

More information: https://www.man7.org/linux/man-pages/man8/poweroff.8.html

- Power off the system:

  ```
  poweroff
  ```

- Halt the system (same as `halt`):

  ```
  poweroff --halt
  ```

- Reboot the system (same as `reboot`):

  ```
  poweroff --reboot
  ```

- Shut down immediately without contacting the system manager:

  ```
  poweroff --force --force
  ```

- Write the wtmp shutdown entry without shutting down the system:

  ```
  poweroff --wtmp-only
  ```

powertop

Optimize battery power usage.

- Calibrate power usage measurements:

  ```
  sudo powertop --calibrate
  ```

- Generate HTML power usage report in the current directory:

  ```
  sudo powertop --html= {{power_report.html}}
  ```

- Tune to optimal settings:

  ```
  sudo powertop --auto-tune
  ```

rename

Rename multiple files.

NOTE: this page refers to the command from the **prename** Fedora package.

- Rename files using a Perl Common Regular Expression (substitute 'foo' with 'bar' wherever found):

  ```
  rename {{'s/foo/bar/'}} {{*}}
  ```

- Dry-run - display which renames would occur without performing them:

  ```
  rename -n {{'s/foo/bar/'}} {{*}}
  ```

- Force renaming even if the operation would remove existing destination files:

  ```
  rename -f {{'s/foo/bar/'}} {{*}}
  ```

- Convert filenames to lower case (use `-f` in case-insensitive filesystems to prevent "already exists" errors):

  ```
  rename 'y/A-Z/a-z/' {{*}}
  ```

- Replace whitespace with underscores:

  ```
  rename 's/\s+/_/g' {{*}}
  ```

print

An alias to a **`run-mailcap`** action print.

Originally **`run-mailcap`** is used to process mime-type/file.

- Print action can be used to print any file on default run-mailcap tool:

  `print {{filename}}`

- With `run-mailcap`

  `run-mailcap --action=print {{filename}}`

procs

Display information about the active processes.

More information: https://github.com/dalance/procs.

- List all processes showing the PID, user, CPU usage, memory usage, and the command which started them:

  ```
  procs
  ```

- Show information about processes, if the commands which started them contain `zsh`:

  ```
  procs {{zsh}}
  ```

- Show information about all processes sorted by CPU time in [a]scending or [d]escending order:

  ```
  procs {{--sortd|--sorta}} cpu
  ```

- Show information about processes with either a PID, command, or user containing (`zsh` or `firefox`):

  ```
  procs --or {{PID|command|user}} {{41}} {{firefox}}
  ```

- Show information about processes with both PID `41` and a command or user containing `zsh`:

  ```
  procs --and {{41}} {{zsh}}
  ```

progress

Display/Monitor the progress of running coreutils.

More information: https://github.com/X_fennec/progress

- Show the progress of running coreutils:

  ```
  progress
  ```

- Show the progress of running coreutils in quiet mode:

  ```
  progress -q
  ```

- Launch and monitor a single long-running command:

  ```
  {{command}} & progress -mp $!
  ```

protontricks

A simple wrapper that does Winetricks things for Proton enabled games, requires Winetricks.

More information: https://github.com/Matoking/protontricks.

- Show the protontricks help message:

  ```
  protontricks
  ```

- Run the protontricks GUI:

  ```
  protontricks --gui
  ```

- Run Winetricks for a specific game:

  ```
  protontricks {{appid}} {{winetricks_args}}
  ```

- Run a command within a games installation directory:

  ```
  protontricks -c {{command}} {{appid}}
  ```

protonvpn-cli connect

Official client to connect to ProtonVPN from the command-line.

More information: https://protonvpn.com/support/linux-vpn-setup/.

- Connect to ProtonVPN interactively:

  ```
  protonvpn-cli connect
  ```

- Connect to ProtonVPN using the fastest server available:

  ```
  protonvpn-cli connect --fastest
  ```

- Connect to ProtonVPN using a specific server with a specific protocol:

  ```
  protonvpn-cli connect {{server_name}} --protocol {{udp|tcp}}
  ```

- Connect to ProtonVPN using a random server with a specific protocol:

  ```
  protonvpn-cli connect --random --protocol {{udp|tcp}}
  ```

- Connect to ProtonVPN using the fastest Tor-supporting server:

  ```
  protonvpn-cli connect --tor
  ```

- Display help:

  ```
  protonvpn connect --help
  ```

protonvpn-cli

Official client for ProtonVPN service from the command-line.

More information: https://github.com/ProtonVPN/linux-cli

- Log in to the ProtonVPN account:

  ```
  protonvpn-cli login {{username}}
  ```

- Start a kill switch upon connecting to ProtonVPN:

  ```
  protonvpn-cli killswitch --on
  ```

- Connect to ProtonVPN interactively:

  ```
  protonvpn-cli connect
  ```

- Display connection status:

  ```
  protonvpn-cli status
  ```

- Block malware using ProtonVPN NetShield:

  ```
  protonvpn-cli netshield --malware
  ```

- Disconnect from ProtonVPN:

  ```
  protonvpn-cli disconnect
  ```

- Display the current ProtonVPN configuration:

  ```
  protonvpn-cli config --list
  ```

- Display help for a subcommand:

  ```
  protonvpn-cli {{subcommand}} --help
  ```

prt-get

The CRUX package manager.

- Install a package:

  ```
  prt-get install {{package_name}}
  ```

- Install a package with dependency handling:

  ```
  prt-get depinst {{package_name}}
  ```

- Update a package manually:

  ```
  prt-get upgrade {{package_name}}
  ```

- Remove a package:

  ```
  prt-get remove {{package_name}}
  ```

- Upgrade the system from the local ports tree:

  ```
  prt-get sysup
  ```

- Search the ports tree:

  ```
  prt-get search {{package_name}}
  ```

- Search for a file in a package:

  ```
  prt-get fsearch {{file}}
  ```

pstree

A convenient tool to show running processes as a tree.

- Display a tree of processes:

  ```
  pstree
  ```

- Display a tree of processes with P IDs:

  ```
  pstree - p
  ```

- Display all process trees rooted at processes owned by specified user:

  ```
  pstree {{u ser}}
  ```

ptx

G enerate a permuted index of words from one or more text files.

More information: https://www.gnu.org/software/coreutils/ptx.

- G enerate a permuted index where the first field of each line is an index reference:

  ```
  ptx - - references {{path/to/file}}
  ```

- G enerate a permuted index with automatically generated index references:

  ```
  ptx - - au to- reference {{path/to/file}}
  ```

- G enerate a permuted index with a fixed width:

  ```
  ptx - - w idth= {{w idth_in_colu m ns}} {{path/to/file}}
  ```

- G enerate a permuted index with a list of filtered words:

  ```
  ptx - - only- file= {{path/to/filter}} {{path/to/file}}
  ```

- G enerate a permuted index with S Y S V -style behaviors:

  ```
  ptx - - traditional {{path/to/file}}
  ```

pulseaudio

The pulseaudio sound system daemon and manager.

More information: https://www.freedesktop.org/wiki/Software/PulseAudio/

- Check if pulseaudio is running (a non-zero exit code means it is not running):

```
pulseaudio --check
```

- Start the pulseaudio daemon in the background:

```
pulseaudio --start
```

- Kill the running pulseaudio daemon:

```
pulseaudio --kill
```

- List available modules:

```
pulseaudio --dump-modules
```

- Load a module into the currently running daemon with the specified arguments:

```
pulseaudio --load="{{module_name}} {{arguments}}"
```

pvcreate

Initialize a disk or partition for use as a physical volume.

See also: `lvm`.

More information: https://man7.org/linux/man-pages/man8/pvcreate.8.html.

- Initialize the `/dev/sda1` volume for use by LVM:

  ```
  pv create {{/dev/sda1}}
  ```

- Force the creation without any confirmation prompts:

  ```
  pv create --force {{/dev/sda1}}
  ```

pvdisplay

Display information about Logical Volume Manager (LVM) physical volumes.

See also: `lvm`.

More information: https://man7.org/linux/man-pages/man8/pvdisplay.8.html.

- Display information about all physical volumes:

  ```
  sudo pvdisplay
  ```

- Display information about the physical volume on drive `/dev/sdX:Y`:

  ```
  sudo pvdisplay {{/dev/sdX:Y}}
  ```

pvs

Display information about physical volumes.

See also: `lvm`.

More information: https://man7.org/linux/man-pages/man8/pvs.8.html

- Display information about physical volumes:

  ```
  pvs
  ```

- Display non-physical volumes:

  ```
  pvs -a
  ```

- Change default display to show more details:

  ```
  pvs -v
  ```

- Display only specific fields:

  ```
  pvs -o {{field_name_1}},{{field_name_2}}
  ```

- Append field to default display:

  ```
  pvs -o +{{field_name}}
  ```

- Suppress heading line:

  ```
  pvs --noheadings
  ```

- Use separator to separate fields:

  ```
  pvs --separator {{special_character}}
  ```

pwdx

Print working directory of a process.

- Print current working directory of a process:

  ```
  pwdx {{process_id}}
  ```

pwgen

G enerate pronounceable passwords.

- G enerate random password with s[y] mbols:

  ```
  pw g en - y {{leng th}}
  ```

- G enerate secure, hard-to-memoriz e passwords:

  ```
  pw g en - s {{leng th}}
  ```

- G enerate password with at least one capital letter in them:

  ```
  pw g en - c {{leng th}}
  ```

q j oypad

Translate input from gamepads or joysticks into keyboard strokes or mouse actions.

More information: http://q j oypad.sourceforge.net/

- Start QJoyPad:

  ```
  q j oypad
  ```

- Start QJoyPad and look for devices in a specific directory:

  ```
  q j oypad --device={{path/to/directory}}
  ```

- Start QJoyPad but don't show a system tray icon:

  ```
  q j oypad --notray
  ```

- Start QJoyPad and force the window manager to use a system tray icon:

  ```
  q j oypad --force-tray
  ```

- Force a running instance of QJoyPad to update its list of devices and layouts:

  ```
  q j oypad --update
  ```

- Load the given layout in an already running instance of QJoyPad, or start QJoyPad using the given layout:

  ```
  q j oypad "{{layout}}"
  ```

q sub

Submits a script to the queue management system TORQUE.

- Submit a script with default settings (depends on TORQUE settings):

  ```
  qsub {{script.sh}}
  ```

- Submit a script with a specified wallclock runtime limit of 1 hour, 2 minutes and 3 seconds:

  ```
  qsub -l walltime={{1}}:{{2}}:{{3}} {{script.sh}}
  ```

- Submit a script that is executed on 2 nodes using 4 cores per node:

  ```
  qsub -l nodes={{2}}:ppn={{4}} {{script.sh}}
  ```

- Submit a script to a specific queue. Note that different queues can have different maximum and minimum runtime limits:

  ```
  qsub -q {{queue_name}} {{script.sh}}
  ```

q_tchooser

A wrapper used to select between Qt development binary versions.

More information: https://manned.org/q_tchooser

- List available Qt versions from the configuration files:

  ```
  q tchooser --list-versions
  ```

- Print environment information:

  ```
  q tchooser --print-env
  ```

- Run the specified tool using the specified Qt version:

  ```
  q tchooser --run-tool={{tool}} --qt={{version_name}}
  ```

- Add a Qt version entry to be able to choose from:

  ```
  q tchooser --install {{version_name}} {{path/to/qmake}}
  ```

- Display all available options:

  ```
  q tchooser --help
  ```

quotacheck

Scan a filesystem for disk usage; create, check and repair quota files.

It is best to run quotacheck with quotas turned off to prevent damage or loss to quota files.

- Check quotas on all mounted non-NFS filesystems:

  ```
  sudo quotacheck --all
  ```

- Force check even if quotas are enabled (this can cause damage or loss to quota files):

  ```
  sudo quotacheck --force {{mountpoint}}
  ```

- Check quotas on a given filesystem in debug mode:

  ```
  sudo quotacheck --debug {{mountpoint}}
  ```

- Check quotas on a given filesystem, displaying the progress:

  ```
  sudo quotacheck --verbose {{mountpoint}}
  ```

- Check user quotas:

  ```
  sudo quotacheck --user {{user}} {{mountpoint}}
  ```

- Check group quotas:

  ```
  sudo quotacheck --group {{group}} {{mountpoint}}
  ```

radeontop

Show utilisation of AMD GPUs.

May require root privileges depending on your system.

More information: https://github.com/clbr/radeontop.

- Show the utilisation of the default AMD GPU:

  ```
  radeontop
  ```

- Enable colored output:

  ```
  radeontop --color
  ```

- Select a specific GPU (the bus number is the first number in the output of `lspci`):

  ```
  radeontop --bus {{bus_number}}
  ```

- Specify the display refresh rate (higher means more GPU overhead):

  ```
  radeontop --ticks {{samples_per_second}}
  ```

rankmirrors

Rank a list of Pacman mirrors by connection and opening speed.

Writes the new mirrorlist to stdout.

More information: https://wiki.archlinux.org/index.php/mirrors.

- Rank a mirror list:

  ```
  rankmirrors {{/etc/pacman.d/mirrorlist}}
  ```

- Output only a given number of the top ranking servers:

  ```
  rankmirrors -n {{number}} {{/etc/pacman.d/mirrorlist}}
  ```

- Be verbose when generating the mirrorlist:

  ```
  rankmirrors -v {{/etc/pacman.d/mirrorlist}}
  ```

- Test only a specific URL:

  ```
  rankmirrors --url {{url}}
  ```

- Output only the response times instead of a full mirrorlist:

  ```
  rankmirrors --times {{/etc/pacman.d/mirrorlist}}
  ```

rc-service

Locate and run OpenRC services with arguments.

See also `open rc`

- Show a service's status:

  ```
  rc-service {{service_name}} status
  ```

- Start a service:

  ```
  sudo rc-service {{service_name}} start
  ```

- Stop a service:

  ```
  sudo rc-service {{service_name}} stop
  ```

- Restart a service:

  ```
  sudo rc-service {{service_name}} restart
  ```

- Simulate running a service's custom command:

  ```
  sudo rc-service --dry-run {{service_name}} {{command_name}}
  ```

- Actually run a service's custom command:

  ```
  sudo rc-service {{service_name}} {{command_name}}
  ```

- Resolve the location of a service definition on disk:

  ```
  sudo rc-service --resolve {{service_name}}
  ```

rc-status

Show status info about runlevels.

See also **`open rc`**

- Show a summary of services and their status:

  `rc-status`

- Include services in all runlevels in the summary:

  `rc-status --all`

- List services that have crashed:

  `rc-status --crashed`

- List manually started services:

  `rc-status --manual`

- List supervised services:

  `rc-status --supervised`

- Get the current runlevel:

  `rc-status --runlevel`

- List all runlevels:

  `rc-status --list`

rc-update

Add and remove OpenRC services to and from runlevels.

See also: `openrc`

- List all services and the runlevels they are added to:

  ```
  rc-update show
  ```

- Add a service to a runlevel:

  ```
  sudo rc-update add {{service_name}} {{runlevel}}
  ```

- Delete a service from a runlevel:

  ```
  sudo rc-update delete {{service_name}} {{runlevel}}
  ```

- Delete a service from all runlevels:

  ```
  sudo rc-update --all delete {{service_name}}
  ```

rdesktop

Remote Desktop Protocol client.

It can be used to connect the remote computer using the RDP protocol.

- Connect to a remote computer (default port is 3389):

```
rdesktop -u {{username}} -p {{password}} {{host:port}}
```

- Simple Examples:

```
rdesktop -u Administrator -p password123 192.168.1.111:3
```

- Connect to a remote computer with full screen (press `Ctrl+Alt+Enter` to exist):

```
rdesktop -u {{username}} -p {{password}} -f {{host:port}}
```

- Use the customed resolution (use the letter 'x' between the number):

```
rdesktop -u {{username}} -p {{password}} -g 1366x768 {{host:port}}
```

- Connect to a remote computer using domain user:

```
rdesktop -u {{username}} -p {{password}} -d {{domainname}} {{host:port}}
```

- Use the 16 bit color (speed up):

```
rdesktop -u {{username}} -p {{password}} -a 16 {{host:port}}
```

readelf

Displays information about E LF files.

More information: http://man7 .org/linux/man-pages/man1 /readelf.1 .html

- Display all information about the E LF file:

  ```
  readelf - all {{path/to/b inary}}
  ```

- Display all the headers present in the E LF file:

  ```
  readelf - - headers {{path/to/b inary}}
  ```

- Display the entries in symbol table section of the E LF file, if it has one:

  ```
  readelf - - sym b ols {{path/to/b inary}}
  ```

- Display the information contained in the E LF header at the start of the file:

  ```
  readelf - - file- header {{path/to/b inary}}
  ```

reboot

Reboot the system.

More information: https://www.man7.org/linux/man-pages/man8/reboot.8.html

- Reboot the system:

  ```
  reboot
  ```

- Power off the system (same as `poweroff`):

  ```
  reboot --poweroff
  ```

- Halt the system (same as `halt`):

  ```
  reboot --halt
  ```

- Reboot immediately without contacting the system manager:

  ```
  reboot --force --force
  ```

- Write the wtmp shutdown entry without rebooting the system:

  ```
  reboot --wtmp-only
  ```

reflector

Arch script to fetch and sort mirrorlists.

- Get all mirrors, sort for download speed and save them:

  ```
  sudo reflector --sort {{rate}} --save {{/etc/pacman.d/mirrorlist}}
  ```

- Only get German HTTPS mirrors:

  ```
  reflector --country {{Germany}} --protocol {{https}}
  ```

- Only get the 10 recently sync'd mirrors:

  ```
  reflector --latest {{10}}
  ```

rename

Rename multiple files.

NOTE: this page refers to the command from the **util-linux** package.

For the Perl version, see **file-rename** or **perl-rename**.

Warning: This command has no safeguards and will overwrite files without prompting.

- Rename files using simple substitutions (substitute 'foo' with 'bar' wherever found):

  ```
  rename {{foo}} {{bar}} {{*}}
  ```

- Dry-run - display which renames would occur without performing them:

  ```
  rename -vn {{foo}} {{bar}} {{*}}
  ```

- Do not overwrite existing files:

  ```
  rename -o {{foo}} {{bar}} {{*}}
  ```

- Change file extensions:

  ```
  rename {{.ext}} {{.bak}} {{*.ext}}
  ```

- Prepend "foo" to all filenames in the current directory:

  ```
  rename {{''}} {{'foo'}} {{*}}
  ```

- Rename a group of increasingly numbered files zero-padding the numbers up to 3 digits:

  ```
  rename {{foo}} {{foo00}} {{foo?}} && rename {{foo}} {{foo0}} {{foo??}}
  ```

reportbug

Bug report tool of Debian distribution.

More information: https://manpages.debian.org/buster/reportbug/reportbug.1.en.html

- Generate a bug report about a specific package, then send it by e-mail:

```
reportbug {{package}}
```

- Report a bug that is not about a specific package (general problem, infrastructure, etc.):

```
reportbug other
```

- Write the bug report to a file instead of sending it by e-mail:

```
reportbug -o {{filename}} {{package}}
```

repquota

Display a summary of existing file quotas for a filesystem.

- Report stats for all quotas in use:

  ```
  sudo repquota -all
  ```

- Report quota stats for all users, even those who aren't using any of their quota:

  ```
  sudo repquota -v {{filesystem}}
  ```

- Report on quotas for users only:

  ```
  repquota --user {{filesystem}}
  ```

- Report on quotas for groups only:

  ```
  sudo repquota --group {{filesystem}}
  ```

- Report on used quota and limits in a human-readable format:

  ```
  sudo repquota --human-readable {{filesystem}}
  ```

- Report on all quotas for users and groups in a human-readable format:

  ```
  sudo repquota -augs
  ```

reset

R einitialises the current terminal. Clears the entire terminal screen.

- R einitialise the current terminal:

  ```
  reset
  ```

- Display the terminal type instead:

  ```
  reset - q
  ```

resize2fs

Resize an ext2, ext3 or ext4 filesystem.

Does not resize the underlying partition, and the filesystem must be unmounted.

- Automatically resize a filesystem:

  ```
  resize2fs {{/dev/sdX N}}
  ```

- Resize the filesystem to a size of 40G, displaying a progress bar:

  ```
  resize2fs -p {{/dev/sdX N}} {{40G}}
  ```

- Shrink the filesystem to its minimum possible size:

  ```
  resize2fs -M {{/dev/sdX N}}
  ```

resolveip

Resolve hostnames to their IP addresses and vice versa.

More information: https://mariadb.com/kb/en/resolveip/.

- Resolve a hostname to an IP address:

  ```
  resolveip {{example.org}}
  ```

- Resolve an IP address to a hostname:

  ```
  resolveip {{1.1.1.1}}
  ```

- Silent mode. Produces less output:

  ```
  resolveip --silent {{example.org}}
  ```

rfkill

Enable and disable wireless devices.

- List devices:

  ```
  rfkill
  ```

- Filter by columns:

  ```
  rfkill -o {{ID,TYPE,DEVICE}}
  ```

- Block devices by type (e.g. bluetooth, wlan):

  ```
  rfkill block {{bluetooth}}
  ```

- Unblock devices by type (e.g. bluetooth, wlan):

  ```
  rfkill unblock {{wlan}}
  ```

- Output in JSON format:

  ```
  rfkill -J
  ```

rig

Utility to piece together a random first name, last name, street number and address, along with a geographically consistent (ie, they all match the same area) city, state, ZIP code, and area code.

More information: https://manpages.ubuntu.com/manpages/focal/man6/rig.6.html

- Display a random name (male or female) and address:

  ```
  rig
  ```

- Display a [m]ale (or [f]emale) random name and address:

  ```
  rig -{{m|f}}
  ```

- Use data files from a specific directory (default is `/usr/share/rig`):

  ```
  rig -d {{path/to/directory}}
  ```

- Display a specific number of identities:

  ```
  rig -c {{number}}
  ```

- Display a specific number of female identities:

  ```
  rig -f -c {{number}}
  ```

rofi

An application launcher and window switcher.

More information: https://github.com/davatorium/rofi.

- Show the list of apps:

  ```
  rofi -show drun
  ```

- Show the list of all commands:

  ```
  rofi -show run
  ```

- Switch between windows:

  ```
  rofi -show window
  ```

- Pipe a list of items to stdin and print the selected item to stdout:

  ```
  printf "{{Choice1\nChoice2\nChoice3}}" | rofi -dmenu
  ```

rolldice

Roll virtual dice.

More information: https://manned.org/rolldice.

- Roll a single 20 sided dice:

  ```
  rolldice d{{20}}
  ```

- Roll two six sided dice and drop the lowest roll:

  ```
  rolldice {{2}}d{{6}}s{{1}}
  ```

- Roll two 20 sided dice and add a modifier value:

  ```
  rolldice {{2}}d{{20}}{{+5}}
  ```

- Roll a 20 sided dice two times:

  ```
  rolldice {{2}}xd{{20}}
  ```

rpcclient

MS-RPC client tool (part of the samba suite).

More information: https://www.samba.org/samba/docs/current/man-html/rpcclient.1.html

- Connect to a remote host:

```
rpcclient --user {{domain}}\{{username}}%{{password}} {{ip}}
```

- Connect to a remote host on a domain without a password:

```
rpcclient --user {{username}} --workgroup {{domain}} --no-pass {{ip}}
```

- Connect to a remote host, passing the password hash:

```
rpcclient --user {{domain}}\{{username}} --pw-nt-hash {{ip}}
```

- Execute shell commands on a remote host:

```
rpcclient --user {{domain}}\{{username}}%{{password}} --command and {{semicolon_separated_commands}} {{ip}}
```

- Display domain users:

```
rpcclient $> enumdomusers
```

- Display privileges:

```
rpcclient $> enumprivs
```

- Display information about a specific user:

```
rpcclient $> queryuser {{username|rid}}
```

- Create a new user in the domain:

```
rpcclient $> createdomuser {{username}}
```

rpcinfo

Makes an RPC call to an RPC server and reports what it finds.

- Show full table of all RPC services registered on localhost:

  ```
  rpcinfo
  ```

- Show concise table of all RPC services registered on localhost:

  ```
  rpcinfo -s {{localhost}}
  ```

- Display table of statistics of rpcbind operations on localhost:

  ```
  rpcinfo -m
  ```

- Display list of entries of given service name (mountd) and version number (2) on a remote nfs share:

  ```
  rpcinfo -l {{remote_nfs_server_ip}} {{mountd}} {{2}}
  ```

- Delete the registration for version 1 of the mountd service for all transports:

  ```
  rpcinfo -d {{mountd}} {{1}}
  ```

rpm

RPM Package Manager.

- Show version of httpd package:

  ```
  rpm -q {{httpd}}
  ```

- List versions of all matching packages:

  ```
  rpm -qa '{{mariadb*}}'
  ```

- Forcibly install a package regardless of currently installed versions:

  ```
  rpm -U {{package_name.rpm}} --force
  ```

- Identify owner of a file and show version of the package:

  ```
  rpm -qf {{/etc/postfix/main.cf}}
  ```

- List package-owned files:

  ```
  rpm -ql {{kernel}}
  ```

- Show scriptlets from an RPM file:

  ```
  rpm -qp --scripts {{package_name.rpm}}
  ```

- Show changed, missing and/or incorrectly installed files of matching packages:

  ```
  rpm -Va '{{php-*}}'
  ```

rpmbuild

RPM Package Build tool.

More information: https://docs.fedoraproject.org/en-US/quick-docs/creating-rpm-packages/.

- Build binary and source packages:

```
rpm build -ba {{path/to/spec_file}}
```

- Build a binary package without source package:

```
rpm build -bb {{path/to/spec_file}}
```

- Specify additional variables when building a package:

```
rpm build -bb {{path/to/spec_file}} --define "{{variable1}} {{value1}}" --define "{{variable2}} {{value2}}"
```

rspamc

Command-line client for rspamd servers.

- Train the bayesian filter to recognise an email as spam:

  `rspamc learn_spam {{path/to/email_file}}`

- Train the bayesian filter to recognise an email as ham:

  `rspamc learn_ham {{path/to/email_file}}`

- Generate a manual report on an email:

  `rspamc symbols {{path/to/email_file}}`

- Show server statistics:

  `rspamc stat`

rtcwake

Enter a system sleep state until specified wakeup time relative to your bios clock.

- Show whether an alarm is set or not:

```
sudo rtcwake -m show -v
```

- Suspend to ram and wakeup after 10 seconds:

```
sudo rtcwake -m mem -s {{10}}
```

- Suspend to disk (higher power saving) and wakeup 15 minutes later:

```
sudo rtcwake -m disk --date +{{15}}min
```

- Freeze the system (more efficient than suspend-to-ram but linux > 3.9 required) and wakeup at a given date and time:

```
sudo rtcwake -m freeze --date {{YYYYMMDDhhmm}}
```

- Disable a previously set alarm:

```
sudo rtcwake -m disable
```

- Perform a dry run to wakeup the computer at a given time. (Press Ctrl + C to abort):

```
sudo rtcwake -m on --date {{hh:ss}}
```

rtorrent

Download torrents over the command-line.

More information: https://github.com/rakshasa/rtorrent.

- Add a torrent file or magnet to be downloaded:

  `rtorrent {{torrent_or_m ag net}}`

- S tart the download:

  `< C trl> S`

- V iew details about downloading torrent:

  `- >`

- Close rtorrent safely:

  `< C trl> Q`

run-mailcap

Run MailCap Programs.

Run mailcap view, see, edit, compose, print - execute programs via entries in the mailcap file (or any of its aliases) will use the given action to process each mime-type/file.

- Individual actions/programs on run-mailcap can be invoked with action flag:

  ```
  run-mailcap --action=ACTION [--option[=value]]
  ```

- In simple language:

  ```
  run-mailcap --action=ACTION {{filename}}
  ```

- Turn on extra information:

  ```
  run-mailcap --action=ACTION --debug {{filename}}
  ```

- Ignore any "copiousoutput" directive and forward output to standard output:

  ```
  run-mailcap --action=ACTION --nopager {{filename}}
  ```

- Display the found command without actually executing it:

  ```
  run-mailcap --action=ACTION --norun {{filename}}
  ```

runcon

Run a program in a different SELinux security context.

With neither context nor command, print the current security context.

More information: https://www.gnu.org/software/coreutils/runcon.

- Determine the current domain:

```
runcon
```

- Specify the domain to run a command in:

```
runcon -t {{domain}}_t {{command}}
```

- Specify the context role to run a command with:

```
runcon -r {{role}}_r {{command}}
```

- Specify the full context to run a command with:

```
runcon {{user}}_u:{{role}}_r:{{domain}}_t {{command}}
```

runuser

Run commands as a specific user and group without asking for password (needs root privileges).

- Run command as a different user:

  ```
  runuser {{user}} -c '{{command}}'
  ```

- Run command as a different user and group:

  ```
  runuser {{user}} -g {{group}} -c '{{command}}'
  ```

- Start a login shell as a specific user:

  ```
  runuser {{user}} -l
  ```

- Specify a shell for running instead of the default shell (also works for login):

  ```
  runuser {{user}} -s {{/bin/sh}}
  ```

- Preserve the entire environment of root (only if `--login` is not specified):

  ```
  runuser {{user}} --preserve-environment -c '{{command}}'
  ```

rusnapshot

BTRFS snapshotting utility written in Rust.

More information: https://github.com/Edu4rdSHL/rusnapshot

- Create a snapshot using a config file:

```
sudo rusnapshot --config {{path/to/config.toml}} --cr
```

- List created snapshots:

```
sudo rusnapshot -c {{path/to/config.toml}} --list
```

- Delete a snapshot by ID or the name of the snapshot:

```
sudo rusnapshot -c {{path/to/config.toml}} --del --id
{{snapshot_id}}
```

- Delete all `hourly` snapshots:

```
sudo rusnapshot -c {{path/to/config.toml}} --list --keep
{{0}} --clean --kind {{hourly}}
```

- Create a read-write snapshot:

```
sudo rusnapshot -c {{path/to/config.toml}} --cr --rw
```

- Restore a snapshot:

```
sudo rusnapshot -c {{path/to/config.toml}} --id
{{snapshot_id}} --restore
```

sa

Summarizes accounting information. Part of the acct package.

Shows commands called by users, including basic info on CPU time spent processing and I/O rates.

- Display executable invocations per user (username not displayed):

  ```
  sudo sa
  ```

- Display executable invocations per user, showing responsible usernames:

  ```
  sudo sa --print-users
  ```

- List resources used recently per user:

  ```
  sudo sa --user-summary
  ```

sacct

Display accounting data from the S lurm service.

More information: https://slurm.schedmd.com/sacct.html.

- Display j ob id, j ob name, partition, account, number of allocated cpus, j ob state, and j ob exit codes for recent j obs:

```
sacct
```

- Display j ob id, j ob state, j ob exit code for recent j obs:

```
sacct - - b rief
```

- Display the allocations of a j ob:

```
sacct - - j ob s {{j ob _id}} - - allocations
```

- Display elapsed time, j ob name, number of req uested CP U s, and memory req uested of a j ob:

```
sacct - - j ob s {{j ob _id}} - - form at= {{elapsed}}, {{j ob nam e}},
{{req cpu s}}, {{req m em }}
```

sacctmgr

View, setup, and manage Slurm accounts.

More information: https://slurm.schedmd.com/sacctmgr.html.

- Show current configuration:

```
sacctmgr show configuration
```

- Add a cluster to the slurm database:

```
sacctmgr add cluster {{cluster_name}}
```

- Add an account to the slurm database:

```
sacctmgr add account {{account_name}}
cluster={{cluster_of_account}}
```

- Show details of user/association/cluster/account:

```
sacctmgr show {{user/association/cluster/account}}
```

sam

AWS S erverless Application Model (S AM) CLI.

More information: https://github.com/awslabs/aws-sam-cli.

- Initializ e a serverless application:

  ```
  sam init
  ```

- Initializ e a serverless application with a specific runtime:

  ```
  sam init - - ru ntim e {{python3 . 7 }}
  ```

- P ackage a S AM application:

  ```
  sam pack ag e
  ```

- B uild your Lambda function code:

  ```
  sam b u ild
  ```

- R un your serverless application locally:

  ```
  sam local start- api
  ```

- Deploy an AWS S AM application:

  ```
  sam deploy
  ```

sar

Monitor performance of various Linux subsystems.

- Report I/O and transfer rate issued to physical devices, one per second (press CTRL+C to quit):

  ```
  sar -b {{1}}
  ```

- Report a total of 10 network device statistics, one per 2 seconds:

  ```
  sar -n DEV {{2}} {{10}}
  ```

- Report CPU utilization, one per 2 seconds:

  ```
  sar -u ALL {{2}}
  ```

- Report a total of 20 memory utilization statistics, one per second:

  ```
  sar -r ALL {{1}} {{20}}
  ```

- Report the run queue length and load averages, one per second:

  ```
  sar -q {{1}}
  ```

- Report paging statistics, one per 5 seconds:

  ```
  sar -B {{5}}
  ```

sbatch

Submit a batch job to the SLURM scheduler.

- Submit a batch job:

  ```
  sbatch {{path/to/job.sh}}
  ```

- Submit a batch job with a custom name:

  ```
  sbatch --job-name={{myjob}} {{path/to/job.sh}}
  ```

- Submit a batch job with a time limit of 30 minutes:

  ```
  sbatch --time={{00:30:00}} {{path/to/job.sh}}
  ```

- Submit a job and request multiple nodes:

  ```
  sbatch --nodes={{3}} {{path/to/job.sh}}
  ```

scancel

Cancel a Slurm job.

More information: https://slurm.schedmd.com/scancel.html.

- Cancel a job using its ID:

  ```
  scancel {{job_id}}
  ```

- Cancel all jobs from a user:

  ```
  scancel {{user_name}}
  ```

scanimage

Scans images with the Scanner Access Now Easy API.

More information: http://sane-project.org/man/scanimage.1.html

- List available scanners to ensure the target device is connected and recognized:

  ```
  scanimage -L
  ```

- Scan an image and save it to a file:

  ```
  scanimage --format={{pnm|tiff|png|jpeg}} > {{path/to/new_image}}
  ```

schroot

Run command or start an interactive shell with a different root directory. More customizable than `chroot`.

More information: https://wiki.debian.org/Schroot

- Run a command in a specific chroot:

```
schroot --chroot {{chroot}} {{command}}
```

- Run a command with options in a specific chroot:

```
schroot --chroot {{chroot}} {{command}} --
{{command_options}}
```

- Run a command in all available chroots:

```
schroot --all {{command}}
```

- Start an interactive shell with in a specific chroot as a specific user:

```
schroot --chroot {{chroot}} --user {{user}}
```

- List available chroots:

```
schroot --list
```

scontrol

View information about and modify jobs.

More information: https://slurm.schedmd.com/scontrol.html.

- Show information for job:

  ```
  scontrol show job {{job_id}}
  ```

- Suspend a comma-separated list of running jobs:

  ```
  scontrol suspend {{job_id}}
  ```

- Resume a comma-separated list of suspended jobs:

  ```
  scontrol resume {{job_id}}
  ```

- Hold a comma-separated list of queued jobs (Use `release` command to permit the jobs to be scheduled):

  ```
  scontrol hold {{job_id}}
  ```

- Release a comma-separated list of suspended job:

  ```
  scontrol release {{job_id}}
  ```

screenkey

A screencast tool to display keys pressed.

More information: https://www.thregr.org/~wavexx/software/screenkey/

- Display keys which are currently being pressed on the screen:

```
screenkey
```

- Display keys and mouse buttons which are currently being pressed on the screen:

```
screenkey --mouse
```

- Launch the settings menu of screenkey:

```
screenkey --show-settings
```

- Launch screenkey at a specific position:

```
screenkey --position {{top|center|bottom|fixed}}
```

- Change the format of the key modifiers displayed on screen:

```
screenkey --mods-mode {{normal|emacs|mac|win|tux}}
```

- Change the appearance of screenkey:

```
screenkey --bg-color "{{#a1b2c3}}" --font {{Hack}} --font-color {{yellow}} --opacity {{0.8}}
```

- Drag and select a window on screen to display screenkey:

```
screenkey --position fixed --geometry {{$(slop -n -f '%g')}}
```

script

Record all terminal output to file.

- Record a new session to a file named `typescript` in the current directory:

  ```
  script
  ```

- Record a new session to a custom filepath:

  ```
  script {{path/to/session.out}}
  ```

- Record a new session, appending to an existing file:

  ```
  script -a {{path/to/session.out}}
  ```

- Record timing information (data is outputted to the standard error):

  ```
  script -t 2> {{path/to/timing file}}
  ```

scriptreplay

Replay a typescript created by the `script` command to the standard output.

- Replay a typescript at the speed it was recorded:

  ```
  scriptreplay {{path/to/timing_file}} {{path/to/typescript}}
  ```

- Replay a typescript at double the original speed:

  ```
  scriptreplay {{path/to/timing file}} {{path/to/typescript}} 2
  ```

- Replay a typescript at half the original speed:

  ```
  scriptreplay {{path/to/timing file}} {{path/to/typescript}}
  0.5
  ```

scrot

Screen capture utility.

More information: https://github.com/resurrecting-open-source-projects/scrot

- Capture a screenshot and save it to the current directory with the current date as the filename:

  ```
  scrot
  ```

- Capture a screenshot and save it as `capture.png`:

  ```
  scrot {{capture.png}}
  ```

- Capture a screenshot interactively:

  ```
  scrot --select
  ```

- Capture a screenshot from the currently focused window:

  ```
  scrot --focused
  ```

- Display a countdown of 10 seconds before taking a screenshot:

  ```
  scrot --countdown --delay {{10}}
  ```

see

Alias to **`run-mailcap view`**.

An alias to a **`run-mailcap print`** action.

- See action can be used to view any file (usually image) on default mailcap explorer:

  ```
  see {{filename}}
  ```

- Using with `run-mailcap`:

  ```
  run-mailcap --action=view {{filename}}
  ```

semanage

S E Linux P olicy Management tool.

More information: https://manned.org/semanage.

- O utput local customiz ations:

```
sem anag e - S {{store}} - o {{path/to/ou tpu t_file}}
```

- T ake a set of commands from a specified file and load them in a single transaction:

```
sem anag e - S {{store}} - i {{path/to/inpu t_file}}
```

- Manage booleans. B ooleans allow the administrator to modify the confinement of processes based on the current configuration:

```
sem anag e b oolean - S {{store}} {{- - delete| - - m odify| - - list| - -
noheading | - - deleteall}} {{- on| - off}} - F {{b oolean|
b oolean_file}}
```

- Manage policy modules:

```
sem anag e m odu le - S {{store}} {{- - add| - - delete| - - list| - -
m odify}} {{- - enab le| - - disab le}} {{m odu le_nam e}}
```

- Disable/E nable dontaudit rules in policy:

```
sem anag e dontau dit - S {{store}} {{on| off}}
```

sensible-browser

O pen the default browser.

- O pen a new window of the default browser:

  ```
  sensib le- b row ser
  ```

- O pen a U R L in the default browser:

  ```
  sensib le- b row ser {{u rl}}
  ```

sensible-editor

Open the default editor.

- Open a file in the default editor:

  ```
  sensible-editor {{file}}
  ```

- Open a file in the default editor, with the cursor at the end of the file:

  ```
  sensible-editor +  {{file}}
  ```

- Open a file in the default editor, with the cursor at the beginning of line 10:

  ```
  sensible-editor +10  {{file}}
  ```

- Open 3 files in vertically split editor windows at the same time:

  ```
  sensible-editor -O3  {{file_1}} {{file_2}} {{file_3}}
  ```

sensors

Report sensors information.

- Show the current readings of all sensor chips:

  ```
  sensors
  ```

- Show temperatures in degrees Fahrenheit:

  ```
  sensors --fahrenheit
  ```

service

Manage services by running init scripts.

The full script path should be omitted (`/etc/init.d/` is assumed).

- List the name and status of all services:

  ```
  service --status-all
  ```

- Start/Stop/Restart/Reload service (start/stop should always be available):

  ```
  service {{service_name}} {{start|stop|restart|reload}}
  ```

- Do a full restart (runs script twice with start and stop):

  ```
  service {{service_name}} --full-restart
  ```

- Show the current status of a service:

  ```
  service {{service_name}} status
  ```

setfacl

Set file access control lists (ACL).

- Modify ACL of a file for user with read and write access:

```
setfacl -m u:{{username}}:rw {{file}}
```

- Modify default ACL of a file for all users:

```
setfacl -d -m u::rw {{file}}
```

- Remove ACL of a file for an user:

```
setfacl -x u:{{username}} {{file}}
```

- Remove all ACL entries of a file:

```
setfacl -b {{file}}
```

setxkbmap

Set the keyboard using the X Keyboard Extension.

- Set the keyboard in French AZERTY:

  ```
  setxkbmap {{fr}}
  ```

- Set multiple keyboard layouts, their variants and switching option:

  ```
  setxkbmap -layout {{us,de}} -variant {{,qwerty}} -option {{'grp:alt_caps_toggle'}}
  ```

- Get help:

  ```
  setxkbmap -help
  ```

- List all layouts:

  ```
  localectl list-x11-keymap-layouts
  ```

- List variants for the layout:

  ```
  localectl list-x11-keymap-variants {{de}}
  ```

- List available switching options:

  ```
  localectl list-x11-keymap-options | grep grp:
  ```

sfill

Securely overwrite the free space and inodes of the partition where the specified directory resides.

More information: https://manned.org/sfill.

- Overwrite free space and inodes of a disk with 38 writes (slow but secure):

```
sfill {{/path/to/mounted_disk_directory}}
```

- Overwrite free space and inodes of a disk with 6 writes (fast but less secure) and show status:

```
sfill -l -v {{/path/to/mounted_disk_directory}}
```

- Overwrite free space and inodes of a disk with 1 write (very fast but insecure) and show status:

```
sfill -ll -v {{/path/to/mounted_disk_directory}}
```

- Overwrite only free space of a disk:

```
sfill -I {{/path/to/mounted_disk_directory}}
```

- Overwrite only free inodes of a disk:

```
sfill -i {{/path/to/mounted_disk_directory}}
```

shutdown

Shutdown and reboot the system.

- Power off (halt) immediately:

  ```
  shutdown -h now
  ```

- Reboot immediately:

  ```
  shutdown -r now
  ```

- Reboot in 5 minutes:

  ```
  shutdown -r +{{5}} &
  ```

- Shutdown at 1:00 pm (Uses 24h clock):

  ```
  shutdown -h 13:00
  ```

- Cancel a pending shutdown/reboot operation:

  ```
  shutdown -c
  ```

sic

Simple IRC client.

Part of the suckless tools.

More information: https://tools.suckless.org/sic/.

- Connect to the default host (irc.ofct.net) with the nickname set in the `$USER` environment variable:

  ```
  sic
  ```

- Connect to a given host, using a given nickname:

  ```
  sic -h {{host}} -n {{nickname}}
  ```

- Connect to a given host, using a given nickname and password:

  ```
  sic -h {{host}} -n {{nickname}} -k {{password}}
  ```

- Join a channel:

  ```
  :j #{{channel}}<Enter>
  ```

- Send a message to a channel or user:

  ```
  :m #{{channel|user}}<Enter>
  ```

- Set default channel or user:

  ```
  :s #{{channel|user}}<Enter>
  ```

silentcast

Silent screencast creator. Saves in `mkv` and animated gif formats.

More information: https://github.com/colinkeenan/silentcast.

- Launch silentcast:

  ```
  silentcast
  ```

- Launch silentcast on a specific display:

  ```
  silentcast --display={{display}}
  ```

sinfo

View information about Slurm nodes and partitions.

See also `squeue` and `sbatch`, which are also part of the Slurm workload manager.

More information: https://slurm.schedmd.com/sinfo.html.

- Show a quick summary overview of the cluster:

  `sinfo --summarize`

- View the detailed status of all partitions across the entire cluster:

  `sinfo`

- View the detailed status of a specific partition:

  `sinfo --partition {{partition_name}}`

- View information about idle nodes:

  `sinfo --states {{idle}}`

- Summarise dead nodes:

  `sinfo --dead`

- List dead nodes and the reasons why:

  `sinfo --list-reasons`

slapt-get

An apt like system for Slackware package management.

Package sources need to be configured in the slapt-getrc file.

- Update the list of available packages and versions:

  ```
  slapt-get --update
  ```

- Install a package, or update it to the latest available version:

  ```
  slapt-get --install {{package name}}
  ```

- Remove a package:

  ```
  slapt-get --remove {{package name}}
  ```

- Upgrade all installed packages to their latest available versions:

  ```
  slapt-get --upgrade
  ```

- Locate packages of interest by the package name, disk set, or version:

  ```
  slapt-get --search {{package name}}
  ```

- Show information about a package:

  ```
  slapt-get --show {{package name}}
  ```

slapt-src

A utility to automate building of slackbuilds.

S lackB uild sources need to be configured in the slapt-srcrc file.

More information: https://github.com/j aos/slapt-src

- U pdate the list of available slackbuilds and versions:

  ```
  slapt- src - - u pdate
  ```

- List all available slackbuilds:

  ```
  slapt- src - - list
  ```

- F etch, build and install the specified slackbuild(s):

  ```
  slapt- src - - install {{slack b u ild_nam e}}
  ```

- Locate slackbuilds of interest by their name or description:

  ```
  slapt- src - - search {{search_term }}
  ```

- Display information about a slackbuild:

  ```
  slapt- src - - show  {{slack b u ild_nam e}}
  ```

slop

G et a selection of the screen.

More information: https://github.com/naelstrof/slop.

- Wait for the user to make a selection and output its geometry to standard output:

  ```
  slop
  ```

- Double click, rather than click and drag, to draw a selection:

  ```
  slop -D
  ```

- H ighlight the selection rather than outlining it:

  ```
  slop -l
  ```

- S pecify the output format:

  ```
  slop -f {{format_string}}
  ```

- S pecify the selection rectangle's color:

  ```
  slop -c {{red}},{{green}},{{blue}},{{alpha}}
  ```

sm

Displays a short message fullscreen.

More information: https://github.com/nomeata/screen-message.

- Display a message in full-screen:

```
sm " {{Hello World!}}"
```

- Display a message with inverted colors:

```
sm -i " {{Hello World!}}"
```

- Display a message with a custom foreground color:

```
sm -f {{blue}} " {{Hello World!}}"
```

- Display a message with a custom background color:

```
sm -b {{#008888}} " {{Hello World!}}"
```

- Display a message rotated 3 times (in steps of 90 degrees, counterclockwise):

```
sm -r {{3}} " {{Hello World!}}"
```

- Display a message using the output from another command:

```
{{echo " Hello World! "}} | sm -
```

smbclient

FTP-like client to access SMB/CIFS resources on servers.

- Connect to a share (user will be prompted for password; `exit` to quit the session):

  ```
  smbclient {{//server/share}}
  ```

- Connect with a different username:

  ```
  smbclient {{//server/share}} --user {{username}}
  ```

- Connect with a different workgroup:

  ```
  smbclient {{//server/share}} --workgroup {{domain}} --user
  {{username}}
  ```

- Connect with a username and password:

  ```
  smbclient {{//server/share}} --user {{username%password}}
  ```

- Download a file from the server:

  ```
  smbclient {{//server/share}} --directory {{path/to/
  directory}} --command "get {{file.txt}}"
  ```

- Upload a file to the server:

  ```
  smbclient {{//server/share}} --directory {{path/to/
  directory}} --command "put {{file.txt}}"
  ```

smbget

`wget`-like utility for downloading files from SMB servers.

More information: https://www.samba.org/samba/docs/current/man-html/smbget.1.html

- Download a file from a server:

```
smbget {{smb://server/share/file}}
```

- Download a share or directory recursively:

```
smbget --recursive {{smb://server/share}}
```

- Connect with a username and password:

```
smbget {{smb://server/share/file}} --user {{username%password}}
```

- Require encrypted transfers:

```
smbget {{smb://server/share/file}} --encrypt
```

smbmap

SMB enumeration tool.

More information: https://github.com/ShawnDEvans/smbmap

- Display SMB shares and permissions on a host, prompting for user's password or NTLM hash:

```
smbmap -u {{username}} --prompt -H {{ip}}
```

- Display SMB shares and permissions on a host, specifying the domain and passing the password NTLM hash:

```
smbmap -u {{username}} --prompt -d {{domain}} -H {{ip}}
```

- Display SMB shares and list a single level of directories and files:

```
smbmap -u {{username}} --prompt -H {{ip}} -r
```

- Display SMB shares and recursively list a defined number of levels of directories and files:

```
smbmap -u {{username}} --prompt -H {{ip}} -R --depth {{3}}
```

- Display SMB shares and recursively list directories and files, downloading the files matching a regular expression:

```
smbmap -u {{username}} --prompt -H {{ip}} -R -A {{pattern}}
```

- Display SMB shares and recursively list directories and files, searching for file content matching a regular expression:

```
smbmap -u {{username}} --prompt -H {{ip}} -R -F {{pattern}}
```

- Execute a shell command on a remote system:

```
smbmap -u {{username}} --prompt -H {{ip}} -x {{command}}
```

- Upload a file to a remote system:

```
smbmap -u {{username}} --prompt -H {{ip}} --upload {{source}} {{destination}}
```

smbpasswd

Change a user's SMB password.

Samba users must also have a local Unix account.

- Change the current user's SMB password:

  ```
  smbpasswd
  ```

- Add a specified user to Samba and set password(user should already exist in system):

  ```
  smbpasswd -a {{username}}
  ```

- Modify an existing Samba user's password:

  ```
  smbpasswd {{username}}
  ```

- Delete a Samba user:

  ```
  smbpasswd -x {{username}}
  ```

snake4

Snake game in the terminal.

More information: https://manpages.debian.org/snake4/snake4.6.en.html

- Start a snake game:

  ```
  snake4
  ```

- Choose level:

  ```
  {{1 | 2 | 3 | 4 | 5}}
  ```

- Navigate the snake:

  ```
  {{Up| Down| Left| Right}} arrow key
  ```

- Pause game:

  ```
  Spacebar
  ```

- Quit game:

  ```
  q
  ```

- Show the high scores:

  ```
  snake4 --highscores
  ```

snake4 scores

Show the high scores from the snake4 game.

More information: https://manpages.debian.org/snake4/snake4.6.en.html

- Show the highscores:

```
snake4 scores
```

snap

Tool for managing the "snap" self-contained software packages.

Similar to what **apt** is for ".deb".

- Search for a package:

  ```
  snap find {{package name}}
  ```

- Install a package:

  ```
  snap install {{package name}}
  ```

- Update a package:

  ```
  snap refresh {{package name}}
  ```

- Update a package to another channel (track, risk, or branch):

  ```
  snap refresh {{package name}} --channel={{channel}}
  ```

- Update all packages:

  ```
  snap refresh
  ```

- Display basic information about installed snap software:

  ```
  snap list
  ```

- Uninstall a package:

  ```
  snap remove {{package name}}
  ```

- Check for recent snap changes in the system:

  ```
  snap changes
  ```

snapper

Filesystem snapshot management tool.

More information: http://snapper.io/manpages/snapper.html.

- List snapshot configs:

  ```
  snapper list-configs
  ```

- Create snapper config:

  ```
  snapper -c {{config}} create-config {{path/to/directory}}
  ```

- Create a snapshot with a description:

  ```
  snapper -c {{config}} create -d "{{snapshot_description}}"
  ```

- List snapshots for a config:

  ```
  snapper -c {{config}} list
  ```

- Delete a snapshot:

  ```
  snapper -c {{config}} delete {{snapshot_number}}
  ```

- Delete a range of snapshots:

  ```
  snapper -c {{config}} delete {{snapshot_X}}-{{snapshot_Y}}
  ```

snmpwalk

SNMP query tool.

More information: https://manned.org/snmpwalk.

- Query the system information of a remote host using SNMPv1 and a community string:

```
snmpwalk -v1 -c {{community}} {{ip}}
```

- Query specific system information on a remote host by OID using SNMPv2 on a specified port:

```
snmpwalk -v2c -c {{community}} {{ip}}:{{port}} {{oid}}
```

- Query specific system information on a remote host by OID using SNMPv3 and authentication without encryption:

```
snmpwalk -v3 -l {{authNoPriv}} -u {{username}} -a {{MD5|S
-A {{passphrase}} {{ip}} {{oid}}
```

- Query specific system information on a remote host by OID using SNMPv3, authentication, and encryption:

```
snmpwalk -v3 -l {{authPriv}} -u {{username}} -a {{MD5|SH
A {{auth_passphrase}} -x {{DES|AES}} -X {{enc_passphrase}}
{{ip}} {{oid}}
```

- Query specific system information on a remote host by OID using SNMPv3 without authentication or encryption:

```
snmpwalk -v3 -l {{noAuthNoPriv}} -u {{username}} {{ip}}
{{oid}}
```

spectre-meltdown-checker

S pectre and Meltdown mitigation detection tool.

More information: https://manned.org/spectre-meltdown-checker.1.

- Check the currently running kernel for S pectre or Meltdown:

```
su do spectre- m eltdow n- check er
```

- Check the currently running kernel and show an explanation of the actions to take in order to mitigate a vulnerability:

```
su do spectre- m eltdow n- check er - - explain
```

- Check for specific variants (defaults to all):

```
su do spectre- m eltdow n- check er - - v ariant {{1 | 2 | 3 | 3 a| 4 | l1 tf
m sb ds| m fb ds| m lpds| m dsu m | taa| m cespc| srb ds}}
```

- Display output using a specific output format:

```
su do spectre- m eltdow n- check er - - b atch {{text| j son| nrpe|
prom etheu s| short}}
```

- Don' t use the /sys interface even if present:

```
su do spectre- m eltdow n- check er - - no- sysfs
```

- Check a non-running kernel:

```
su do spectre- m eltdow n- check er - - k ernel {{path/to/
k ernel_file}}
```

speedometer

Python script that shows a network traffic graph in the terminal.

More information: http://excess.org/speedometer.

- Show graph for a specific interface:

```
speedometer -r {{eth0}} -t {{eth0}}
```

spi

A meta package manager that handles both packages and slackbuilds.

More information: https://github.com/gapan/spi.

- U pdate the list of available packages and slackbuilds:

  ```
  spi - - u pdate
  ```

- Install a package or slackbuild:

  ```
  spi - - install {{pack ag e/slack b u ild_nam e}}
  ```

- U pgrade all installed packages to the latest versions available:

  ```
  spi - - u pg rade
  ```

- Locate packages or slackbuilds of interest by package name or description:

  ```
  spi {{search_term s}}
  ```

- Display information about a package or slackbuild:

  ```
  spi - - show  {{pack ag e/slack b u ild_nam e}}
  ```

- P urge the local package and slackbuild caches:

  ```
  spi - - clean
  ```

sq ueue

View the jobs queued in the SLURM scheduler.

- View the queue:

  ```
  squeue
  ```

- View jobs queued by a specific user:

  ```
  squeue -u {{username}}
  ```

- View the queue and refresh every 5 seconds:

  ```
  squeue -i {{5}}
  ```

- View the queue with expected start times:

  ```
  squeue --start
  ```

sreport

Generate reports on jobs, users, and clusters from accounting data.

More information: https://slurm.schedmd.com/sreport.html.

- Show pipe delimited cluster utilization data:

  ```
  sreport --parsable cluster utilization
  ```

- Show number of jobs run:

  ```
  sreport job sizes printjob count
  ```

- Show users with highest cpu time use:

  ```
  sreport user topuser
  ```

srun

Create an interactive slurm job or connect to an existing job.

More information: https://slurm.schedmd.com/srun.html.

- Submit a basic interactive job:

  ```
  srun --pty /bin/bash
  ```

- Submit an interactive job with different attributes:

  ```
  srun --ntasks-per-node={{number_cores}} --mem-per-cpu={{memory_MB}} --pty /bin/bash
  ```

- Connect to a worker node with a job running:

  ```
  srun --jobid={{job_id}} --pty /bin/bash
  ```

ss

Utility to investigate sockets.

More information: https://manned.org/ss.8.

- Show all TCP/UDP/RAW/UNIX sockets:

```
ss -a {{-t|-u|-w|-x}}
```

- Filter TCP sockets by states, only/exclude:

```
ss {{state/exclude}} {{bucket/big/connected/synchronized/...}}
```

- Show all TCP sockets connected to the local HTTPS port (443):

```
ss -t src :{{443}}
```

- Show all TCP sockets listening on the local 8080 port:

```
ss -lt src :{{8080}}
```

- Show all TCP sockets along with processes connected to a remote ssh port:

```
ss -pt dst :{{ssh}}
```

- Show all UDP sockets connected on specific source and destination ports:

```
ss -u 'sport == :{{source_port}} and dport == :{{destination_port}}'
```

- Show all TCP IPv4 sockets locally connected on the subnet 192.168.0.0/16:

```
ss -4t src {{192.168/16}}
```

ssh-add

Manage loaded ssh keys in the ssh-agent.

E nsure that ssh-agent is up and running for the keys to be loaded in it.

- Add the default ssh keys in `~/.ssh` to the ssh-agent:

  ```
  ssh-add
  ```

- Add a specific key to the ssh-agent:

  ```
  ssh-add {{path/to/private_key}}
  ```

- List fingerprints of currently loaded keys:

  ```
  ssh-add -l
  ```

- Delete a key from the ssh-agent:

  ```
  ssh-add -d {{path/to/private_key}}
  ```

- Delete all currently loaded keys from the ssh-agent:

  ```
  ssh-add -D
  ```

sshuttle

Transparent proxy server that tunnels traffic over an SSH connection.

Doesn't require root or any special setup on the remote SSH server, though root access on the local machine is prompted for.

- Forward all IPv4 TCP traffic via a remote SSH server:

  ```
  sshuttle --remote={{user name}}@{{sshserver}} {{0.0.0.0/0}}
  ```

- Also forward all DNS traffic to the server's default DNS resolver:

  ```
  sshuttle --dns --remote={{user name}}@{{sshserver}}
  {{0.0.0.0/0}}
  ```

- Forward all traffic except that which is bound for a specific subnet:

  ```
  sshuttle --remote={{user name}}@{{sshserver}} {{0.0.0.0/0}}
  exclude {{192.168.0.1/24}}
  ```

- Use the tproxy method to forward all IPv4 and IPv6 traffic:

  ```
  sshuttle --method=tproxy --remote={{user name}}@{{sshserver}}
  {{0.0.0.0/0}} {{::/0}} --exclude={{your_local_ip_address}}
  exclude={{ssh_server_ip_address}}
  ```

sstat

View information about running jobs.

More information: https://slurm.schedmd.com/sstat.html.

- Display status information of a comma-separated list of jobs:

```
sstat --jobs={{job_id}}
```

- Display job ID, average CPU and average virtual memory size of a comma-separated list of jobs, with pipes as column delimiters:

```
sstat --parsable --jobs={{job_id}} --format={{JobID}},{{AveCPU}},{{AveVMSize}}
```

- Display list of fields available:

```
sstat --helpformat
```

steghide

Steganography tool for JPEG, BMP, WAV and AU file formats.

More information: https://github.com/Stefano Deluono/steghide

- Embed data in a PNG image, prompting for a passphrase:

  ```
  steghide embed --coverfile {{path/to/image.png}} --embedfile {{path/to/data.txt}}
  ```

- Extract data from a WAV audio file:

  ```
  steghide extract --stegofile {{path/to/sound.wav}}
  ```

- Display file information, trying to detect an embedded file:

  ```
  steghide info {{path/to/file.jpg}}
  ```

- Embed data in a JPEG image, using maximum compression:

  ```
  steghide embed --coverfile {{path/to/image.jpg}} --embedfile {{path/to/data.txt}} --compress {{9}}
  ```

- Get the list of supported encryption algorithms and modes:

  ```
  steghide encinfo
  ```

- Embed encrypted data in a JPEG image, e.g. with Blowfish in CBC mode:

  ```
  steghide embed --coverfile {{path/to/image.jpg}} --embedfile {{path/to/data.txt}} --encryption {{blowfish|...}} {{cbc|...}}
  ```

strace

Troubleshooting tool for tracing system calls.

- Start tracing a specific process by its PID:

  ```
  strace -p {{pid}}
  ```

- Trace a process and filter output by system call:

  ```
  strace -p {{pid}} -e {{system_call_name}}
  ```

- Count time, calls, and errors for each system call and report a summary on program exit:

  ```
  strace -p {{pid}} -c
  ```

- Show the time spent in every system call:

  ```
  strace -p {{pid}} -T
  ```

- Start tracing a program by executing it:

  ```
  strace {{program}}
  ```

- Start tracing file operations of a program:

  ```
  strace -e trace=file {{program}}
  ```

stress

A tool to stress test CPU, memory, and IO on a Linux system.

- Spawn 4 workers to stress test CPU:

  ```
  stress -c {{4}}
  ```

- Spawn 2 workers to stress test IO and timeout after 5 seconds:

  ```
  stress -i {{2}} -t {{5}}
  ```

- Spawn 2 workers to stress test memory (each worker allocates 256 M bytes):

  ```
  stress -m {{2}} --vm-bytes {{256M}}
  ```

- Spawn 2 workers spinning on write()/unlink() (each worker writes 1 G bytes):

  ```
  stress -d {{2}} --hdd-bytes {{1GB}}
  ```

swapoff

Disables device or file for swapping.

- Disable a given swap partition:

  ```
  sw apoff {{/dev /sdb 7 }}
  ```

- Disable a given swap file:

  ```
  sw apoff {{path/to/file}}
  ```

- Disable all swap areas:

  ```
  sw apoff - a
  ```

- Disable swap by label of a device or file:

  ```
  sw apoff - L  {{sw ap1 }}
  ```

swapon

Enables device or file for swapping.

- Get swap information:

  ```
  swapon -s
  ```

- Enable a given swap partition:

  ```
  swapon {{/dev/sdb7}}
  ```

- Enable a given swap file:

  ```
  swapon {{path/to/file}}
  ```

- Enable all swap areas:

  ```
  swapon -a
  ```

- Enable swap by label of a device or file:

  ```
  swapon -L {{swap1}}
  ```

swupd

Package management utility for Clear Linux.

More information: https://docs.01.org/clearlinux/latest/guides/clear/swupd.html

- Update to latest version:

  ```
  sudo swupd update
  ```

- Show current version, and check whether a newer one exists:

  ```
  swupd check-update
  ```

- List installed bundles:

  ```
  swupd bundle-list
  ```

- Locate the bundle where a wanted package exists:

  ```
  swupd search -b {{package}}
  ```

- Install a new bundle:

  ```
  sudo swupd bundle-add {{bundle}}
  ```

- Remove a bundle:

  ```
  sudo swupd bundle-remove {{bundle}}
  ```

- Correct broken or missing files:

  ```
  sudo swupd verify
  ```

sxiv

Simple X Image Viewer.

More information: https://github.com/muennich/sxiv.

- Open an image:

  ```
  sxiv {{path/to/file}}
  ```

- Open an image in fullscreen mode:

  ```
  sxiv -f {{path/to/file}}
  ```

- Open a newline-separated list of images, reading filenames from standard input:

  ```
  echo {{path/to/file}} | sxiv -i
  ```

- Open a space-separated list of images as a slideshow:

  ```
  sxiv -S {{seconds}} {{path/to/file}}
  ```

- Open a space-separated list of images in thumbnail mode:

  ```
  sxiv -t {{path/to/file}}
  ```

synopkg

Package management utility for Synology DiskStation Manager.

More information: https://www.synology.com/dsm.

- List the names of installed packages:

  ```
  synopkg list --name
  ```

- List packages which depend on a specific package:

  ```
  synopkg list --depend-on {{package}}
  ```

- Start/Stop a package:

  ```
  sudo synopkg {{start|stop}} {{package}}
  ```

- Print the status of a package:

  ```
  synopkg status {{package}}
  ```

- Uninstall a package:

  ```
  sudo synopkg uninstall {{package}}
  ```

- Check if updates are available for a package:

  ```
  synopkg check update {{package}}
  ```

- Upgrade all packages to the latest version:

  ```
  sudo synopkg upgradeall
  ```

- Install a package from a synopkg file:

  ```
  sudo synopkg install {{path/to/package.spk}}
  ```

synoupgrade

Upgrade a Synology DiskStation Manager (DSM) from the command-line.

More information: https://www.synology.com/dsm.

- Check if upgrades are available:

  ```
  sudo synoupgrade --check
  ```

- Check for patches without upgrading the DSM version:

  ```
  sudo synoupgrade --check -smallupdate
  ```

- Download the latest upgrade available (use `--download-smallupdate` for patches):

  ```
  sudo synoupgrade --download
  ```

- Start the upgrade process:

  ```
  sudo synoupgrade --start
  ```

- Upgrade to the latest version automatically:

  ```
  sudo synoupgrade --auto
  ```

- Apply patches without upgrading the DSM version automatically:

  ```
  sudo synoupgrade --auto-smallupdate
  ```

- Upgrade the DSM using a patch file (should be an absolute path):

  ```
  sudo synoupgrade --patch {{/path/to/file.pat}}
  ```

- Display help:

  ```
  synoupgrade
  ```

sysctl

List and change kernel runtime variables.

- Show all available variables and their values:

  ```
  sysctl -a
  ```

- Set a changeable kernel state variable:

  ```
  sysctl -w {{section.tunable}}={{value}}
  ```

- Get currently open file handlers:

  ```
  sysctl fs.file-nr
  ```

- Get limit for simultaneous open files:

  ```
  sysctl fs.file-max
  ```

- Apply changes from `/etc/sysctl.conf`:

  ```
  sysctl -p
  ```

systemctl

Control the systemd system and service manager.

More information: https://www.freedesktop.org/software/systemd/man/systemctl.html.

- List failed units:

```
system ctl - - failed
```

- S tart/S top/R estart/R eload a service:

```
system ctl {{start| stop| restart| reload}} {{u nit}}
```

- S how the status of a unit:

```
system ctl statu s {{u nit}}
```

- E nable/Disable a unit to be started on bootup:

```
system ctl {{enab le| disab le}} {{u nit}}
```

- Mask/U nmask a unit to prevent enablement and manual activation:

```
system ctl {{m ask | u nm ask }} {{u nit}}
```

- R eload systemd, scanning for new or changed units:

```
system ctl daem on- reload
```

- Check if a unit is active:

```
system ctl is- activ e {{u nit}}
```

- Check if a unit is enabled:

```
system ctl is- enab led {{u nit}}
```

systemd-analyze

Show timing details about the boot process of units (services, mount points, devices, sockets).

- List time of each unit to start up:

  ```
  systemd-analyze blame
  ```

- Print a tree of the time critical chain of units:

  ```
  systemd-analyze critical-chain
  ```

- Create an SVG file showing when each system service started, highlighting the time that they spent on initialization:

  ```
  systemd-analyze plot > {{path/to/file.svg}}
  ```

- Plot a dependency graph and convert it to an SVG file:

  ```
  systemd-analyze dot | dot -Tsvg > {{path/to/file.svg}}
  ```

taskset

Get or set a process' CPU affinity or start a new process with a defined CPU affinity.

- Get a running process' CPU affinity by PID:

  ```
  taskset --pid --cpu-list {{pid}}
  ```

- Set a running process' CPU affinity by PID:

  ```
  taskset --pid --cpu-list {{cpu_id}} {{pid}}
  ```

- Start a new process with affinity for a single CPU:

  ```
  taskset --cpu-list {{cpu_id}} {{command}}
  ```

- Start a new process with affinity for multiple non-sequential CPUs:

  ```
  taskset --cpu-list {{cpu_id_1}} {{cpu_id_2}} {{cpu_id_3}}
  ```

- Start a new process with affinity for CPUs 1 through 4:

  ```
  taskset --cpu-list {{cpu_id_1}}, {{cpu_id_4}}
  ```

tcpflow

Capture TCP traffic for debugging and analysis.

- Show all data on the given interface and port:

```
tcpflow -c -i {{eth0}} port {{80}}
```

tcpkill

Kills specified in-progress TCP connections.

- Kill in-progress connections at a specified interface, host and port:

```
tcpkill -i {{eth1}} host {{192.95.4.27}} and port {{2266}}
```

tcptraceroute

A traceroute implementation using TCP packets.

More information: https://github.com/mct/tcptraceroute.

- Trace the route to a host:

  ```
  tcptraceroute {{host}}
  ```

- Specify the destination port and packet length in bytes:

  ```
  tcptraceroute {{host}} {{destination_port}} {{packet_length}}
  ```

- Specify the local source port and source address:

  ```
  tcptraceroute {{host}} -p {{source_port}} -s
  {{source_address}}
  ```

- Set the first and maximum TTL:

  ```
  tcptraceroute {{host}} -f {{first_ttl}} -m {{max_ttl}}
  ```

- Specify the wait time and number of queries per hop:

  ```
  tcptraceroute {{host}} -w {{wait_time}} -q
  {{number_of_queries}}
  ```

- Specify the interface:

  ```
  tcptraceroute {{host}} -i {{interface}}
  ```

terminator

Arrange multiple G N O ME terminals in one window.

More information: https://gnome-terminator.org/.

- S tart terminator window:

  `term inator`

- S tart with a fullscreen window:

  `term inator - f`

- S plit terminals horiz ontally:

  `C trl +  S hift +  O`

- S plit terminals vertically:

  `C trl +  S hift +  E`

- O pen new tab:

  `C trl +  S hift +  T`

thunar

Graphical file manager for XFCE desktop environments.

More information: https://docs.xfce.org/xfce/thunar/start.

- Open a new window showing the current directory:

  ```
  thunar
  ```

- Open the bulk rename utility:

  ```
  thunar --bulk-rename
  ```

- Close all open thunar windows:

  ```
  thunar --quit
  ```

tic

Compile terminfo and install for ncurses.

More information: https://pubs.opengroup.org/onlinepubs/0 0 7 9 0 8 7 9 9 /xcurses/terminfo.html.

- Compile and install terminfo for a terminal:

```
tic - xe {{term inal}} {{path/to/term inal. info}}
```

- Check terminfo file for errors:

```
tic - c {{path/to/term inal. info}}
```

- P rint database locations:

```
tic - D
```

timedatectl

Control the system time and date.

More information: https://manned.org/timedatectl.1.

- Check the current system clock time:

  ```
  timedatectl
  ```

- Set the local time of the system clock directly:

  ```
  timedatectl set-time "{{yyyy-MM-dd hh:mm:ss}}"
  ```

- List available timezones:

  ```
  timedatectl list-timezones
  ```

- Set the system timezone:

  ```
  timedatectl set-timezone {{timezone}}
  ```

- Enable Network Time Protocol (NTP) synchronization:

  ```
  timedatectl set-ntp on
  ```

timeshift

System restore utility.

More information: https://github.com/teejee2008/timeshift

- List snapshots:

```
sudo timeshift --list
```

- Create a new snapshot (if scheduled):

```
sudo timeshift --check
```

- Create a new snapshot (even if not scheduled):

```
sudo timeshift --create
```

- Restore a snapshot (selecting which snapshot to restore interactively):

```
sudo timeshift --restore
```

- Restore a specific snapshot:

```
sudo timeshift --restore --snapshot '{{snapshot}}'
```

- Delete a specific snapshot:

```
sudo timeshift --delete --snapshot '{{snapshot}}'
```

tlp-stat

A tool to generate TLP status reports.

See also `tlp`

More information: https://linrunner.de/tlp/usage/tlp-stat.

- Generate status report with configuration and all active settings:

  ```
  sudo tlp-stat
  ```

- Show battery information:

  ```
  sudo tlp-stat -b
  ```

- Show configuration:

  ```
  sudo tlp-stat -c
  ```

tlp

Advanced power management for Linux.

S ee also `t lp- s t.a t`

More information: https://linrunner.de/tlp/.

- Apply settings (according to the actual power source):

  ```
  su do tlp start
  ```

- Apply battery settings (ignoring the actual power source):

  ```
  su do tlp b at
  ```

- Apply AC settings (ignoring the actual power source):

  ```
  su do tlp ac
  ```

toilet

A tool to display ASCII-art fonts.

More information: http://caca.zoy.org/wiki/toilet

- Generate ASCII art for a given text:

  ```
  toilet {{input_text}}
  ```

- Generate ASCII art using a custom font file:

  ```
  toilet {{input_text}} -f {{font_filename}}
  ```

- Generate ASCII art using a filter:

  ```
  toilet {{input_text}} --filter {{filter_name}}
  ```

- Show available toilet filters:

  ```
  toilet --filter list
  ```

tomb

Manage encrypted storage directories that can be safely transported and hidden in a filesystem.

More information: https://www.dyne.org/software/tomb/.

- Create a new tomb with an initial size of 100 MB:

```
tomb dig -s {{100}} {{encrypted_directory.tomb}}
```

- Create a new key file that can be used to lock a tomb; user will be prompted for a password for the key:

```
tomb forge {{encrypted_directory.tomb.key}}
```

- Forcefully create a new key, even if the tomb isn't allowing key forging (due to swap):

```
tomb forge {{encrypted_directory.tomb.key}} -f
```

- Initialize and lock an empty tomb using a key made with `forge`:

```
tomb lock {{encrypted_directory.tomb}} -k
{{encrypted_directory.tomb.key}}
```

- Mount a tomb (by default in `/media`) using its key, making it usable as a regular filesystem directory:

```
tomb open {{encrypted_directory.tomb}} -k
{{encrypted_directory.tomb.key}}
```

- Close a tomb (fails if the tomb is being used by a process):

```
tomb close {{encrypted_directory.tomb}}
```

- Forcefully close all open tombs, killing any applications using them:

```
tomb slam all
```

- List all open tombs:

```
tomb list
```

top

Display dynamic real-time information about running processes.

More information: https://manned.org/top.

- Start top:

```
top
```

- Do not show any idle or zombie processes:

```
top -i
```

- Show only processes owned by given user:

```
top -u {{username}}
```

- Sort processes by a field:

```
top -o {{field_name}}
```

- Show the individual threads of a given process:

```
top -Hp {{process_id}}
```

- Show only the processes with the given PID(s), passed as a comma-separated list. (Normally you wouldn't know PIDs off hand. This example picks the PIDs from the process name):

```
top -p $(pgrep -d ',' {{process_name}})
```

- Get help about interactive commands:

```
?
```

tracepath

Trace the path to a network host discovering MTU along this path.

More information: https://manned.org/tracepath.

- A preferred way to trace the path to a host:

  ```
  tracepath -p {{33434}} {{host}}
  ```

- Specify the initial destination port, useful with non-standard firewall settings:

  ```
  tracepath -p {{destination_port}} {{host}}
  ```

- Print both hostnames and numerical IP addresses:

  ```
  tracepath -b {{host}}
  ```

- Specify a maximum TTL (number of hops):

  ```
  tracepath -m {{max_hops}} {{host}}
  ```

- Specify the initial packet length (defaults to 65535 for IPv4 and 128000 for IPv6

  ```
  tracepath -l {{packet_length}} {{host}}
  ```

- Use only IPv6 addresses:

  ```
  tracepath -6 {{host}}
  ```

trap

Automatically execute commands after receiving signals by processes or the operating system.

Can be used to perform cleanups for interruptions by the user or other actions.

More information: https://manned.org/trap.

- List available signals to set traps for:

```
trap -l
```

- List active traps for the current shell:

```
trap -p
```

- Set a trap to execute commands when one or more signals are detected:

```
trap 'echo "Caught signal {{SIGHUP}}"' {{SIGHUP}}
```

- Remove active traps:

```
trap - {{SIGHUP}} {{SIGINT}}
```

trash

A CLI for managing your trashcan / recycling bin.

More information: https://github.com/andreafrancia/trash-cli.

- Delete a file (send to trash):

  ```
  trash {{path/to/file}}
  ```

- List files in trash:

  ```
  trash-list
  ```

- Restore file from trash:

  ```
  trash-restore
  ```

- Empty trash:

  ```
  trash-empty
  ```

- Empty trash, keeping files trashed less than {{10}} days ago:

  ```
  trash-empty {{10}}
  ```

- Remove all files named 'foo' from the trash:

  ```
  trash-rm foo
  ```

- Remove all files with a given original location:

  ```
  trash-rm {{/absolute/path/to/file_or_directory}}
  ```

tree

Show the contents of the current directory as a tree.

More information: http://mama.indstate.edu/users/ice/tree/.

- Print files and directories up to 'num' levels of depth (where 1 means the current directory):

  ```
  tree -L {{num}}
  ```

- Print directories only:

  ```
  tree -d
  ```

- Print hidden files too with colorization on:

  ```
  tree -a -C
  ```

- Print the tree without indentation lines, showing the full path instead (use `-N` to not escape non-printable characters):

  ```
  tree -i -f
  ```

- Print the size of each file and the cumulative size of each directory, in human-readable format:

  ```
  tree -s -h --du
  ```

- Print files within the tree hierarchy, using a wildcard (glob) pattern, and pruning out directories that don't contain matching files:

  ```
  tree -P '{{*.txt}}' --prune
  ```

- Print directories within the tree hierarchy, using the wildcard (glob) pattern, and pruning out directories that aren't ancestors of the wanted one:

  ```
  tree -P {{directory_name}} --matchdirs --prune
  ```

- Print the tree ignoring the given directories:

  ```
  tree -I '{{directory_name1|directory_name2}}'
  ```

trizen

Arch Linux utility for building packages from the Arch User Repository (AUR).

More information: https://github.com/trizen/trizen

- Synchronize and update all AUR packages:

  ```
  trizen -Syua
  ```

- Install a new package:

  ```
  trizen -S {{package}}
  ```

- Remove a package and its dependencies:

  ```
  trizen -Rs {{package}}
  ```

- Search the package database for a keyword:

  ```
  trizen -Ss {{keyword}}
  ```

- Show information about a package:

  ```
  trizen -Si {{package}}
  ```

- List installed packages and versions:

  ```
  trizen -Qe
  ```

tshark

Packet analysis tool, CLI version of Wireshark.

More information: https://tshark.dev/.

- Monitor everything on localhost:

```
tshark
```

- Only capture packets matching a specific capture filter:

```
tshark -f '{{udp port 53}}'
```

- Only show packets matching a specific output filter:

```
tshark -Y '{{http.request.method == "GET"}}'
```

- Decode a TCP port using a specific protocol (e.g. HTTP):

```
tshark -d tcp.port==={{8888}},{{http}}
```

- Specify the format of captured output:

```
tshark -T {{json|text|ps|…}}
```

- Select specific fields to output:

```
tshark -T {{fields|ek|jsonl|pdml}} -e {{http.request.method}}
-e {{ip.src}}
```

- Write captured packet to a file:

```
tshark -w {{path/to/file}}
```

- Analyze packets from a file:

```
tshark -r {{filename}}.pcap
```

tune2fs

Adjust parameters of an ext2, ext3 or ext4 filesystem.

May be used on mounted filesystems.

More information: https://manned.org/tune2fs

- Set the max number of counts before a filesystem is checked to 2:

  ```
  tune2fs -c {{2}} {{/dev/sdX1}}
  ```

- Set the filesystem label to MY_LABEL:

  ```
  tune2fs -L {{'MY_LABEL'}} {{/dev/sdX1}}
  ```

- Enable discard and user-specified extended attributes for a filesystem:

  ```
  tune2fs -o {{discard,user_xattr}} {{/dev/sdX1}}
  ```

- Enable journaling for a filesystem:

  ```
  tune2fs -O^{{nobarrier}} {{/dev/sdX1}}
  ```

tuxi

A CLI tool that scrapes G oogle search results and S E R P s that provides instant and concise answers.

More information: https://github.com/B ugswriter/tuxi

- Make a search using G oogle:

  ```
  tu xi {{search_term s}}
  ```

- Display the search results in [r] aw format (no pretty output, no colors):

  ```
  tu xi - r {{search_term s}}
  ```

- Display only search results (silences " Did you mean? " , greetings and usage):

  ```
  tu xi - q {{search_term s}}
  ```

- Display help:

  ```
  tu xi - h
  ```

udisksctl

A command-line program used to interact with the udisksd daemon process.

More information: http://storaged.org/doc/udisks2-api/latest/udisksctl.1.html

- Show high-level information about disk drives and block devices:

  ```
  udisksctl status
  ```

- Show detailed information about a device:

  ```
  udisksctl info --block-device {{/dev/sdX}}
  ```

- Show detailed information about a device partition:

  ```
  udisksctl info --block-device {{/dev/sdX N}}
  ```

- Mount a device partition and prints the mount point:

  ```
  udisksctl mount --block-device {{/dev/sdX N}}
  ```

- Unmount a device partition:

  ```
  udisksctl unmount --block-device {{/dev/sdX N}}
  ```

- Monitor the daemon for events:

  ```
  udisksctl monitor
  ```

ufw

Uncomplicated Firewall.

Frontend for iptables aiming to make configuration of a firewall easier.

More information: https://wiki.ubuntu.com/UncomplicatedFirewall

- Enable ufw:

  ```
  ufw enable
  ```

- Disable ufw:

  ```
  ufw disable
  ```

- Show ufw rules, along with their numbers:

  ```
  ufw status numbered
  ```

- Allow incoming traffic on port 5432 on this host with a comment identifying the service:

  ```
  ufw allow {{5432}} comment "{{service}}"
  ```

- Allow only TCP traffic from 192.168.0.4 to any address on this host, on port 22:

  ```
  ufw allow proto {{tcp}} from {{192.168.0.4}} to {{any}} port {{22}}
  ```

- Deny traffic on port 80 on this host:

  ```
  ufw deny {{80}}
  ```

- Deny all UDP traffic to port 22:

  ```
  ufw deny proto {{udp}} from {{any}} to {{any}} port {{22}}
  ```

- Delete a particular rule. The rule number can be retrieved from the `ufw status numbered` command:

  ```
  ufw delete {{rule_number}}
  ```

ul

Performs the underlining of a text.

Each character in a given string must be underlined separately.

More information: https://manned.org/ul.

- Display the contents of the file with underlines where applicable:

  ```
  ul {{file.txt}}
  ```

- Display the contents of the file with underlines made of dashes `-`:

  ```
  ul -i {{file.txt}}
  ```

unix2dos

Change Unix-style line endings to DOS-style.

Replaces CR with CRLF.

More information: https://waterlan.home.xs4all.nl/dos2unix.html

- Change the line endings of a file:

  ```
  unix2dos {{filename}}
  ```

- Create a copy with DOS-style line endings:

  ```
  unix2dos -n {{filename}} {{new_filename}}
  ```

unix2mac

Change Unix-style line endings to macOS-style.

Replaces CR with LF.

More information: https://waterlan.home.xs4all.nl/dos2unix.html

- Change the line endings of a file:

  ```
  unix2mac {{filename}}
  ```

- Create a copy with macOS-style line endings:

  ```
  unix2mac -n {{filename}} {{new_filename}}
  ```

unset

Remove shell variables or functions.

More information: https://manned.org/unset.

- Remove the variable `foo`, or if the variable doesn't exist, remove the function `foo`:

  `unset {{foo}}`

- Remove the variables foo and bar:

  `unset -v {{foo}} {{bar}}`

- Remove the function my_func:

  `unset -f {{my_func}}`

unshadow

Utility provided by the John the Ripper project to obtain the traditional Unix password file if the system uses shadow passwords.

More information: https://www.openwall.com/john/

- Combine the `/etc/shadow` and `/etc/passwd` of the current system:

  ```
  sudo unshadow /etc/passwd /etc/shadow
  ```

- Combine two arbitrary shadow and password files:

  ```
  sudo unshadow {{path/to/passwd}} {{path/to/shadow}}
  ```

update-alternatives

A convenient tool for maintaining symbolic links to determine default commands.

More information: https://manned.org/update-alternatives.

- Add a symbolic link:

```
su do u pdate- alternativ es - - install {{path/to/sym link }}
{{com m and_nam e}} {{path/to/com m and_b inary}} {{priority}}
```

- Configure a symbolic link for `j av a`:

```
su do u pdate- alternativ es - - config  {{j av a}}
```

- R emove a symbolic link:

```
su do u pdate- alternativ es - - rem ov e {{j av a}} {{/opt/j av a/
j dk 1 . 8 . 0 _1 0 2 /b in/j av a}}
```

- Display information about a specified command:

```
u pdate- alternativ es - - display {{j av a}}
```

- Display all commands and their current selection:

```
u pdate- alternativ es - - g et- selections
```

update-rc.d

Install and remove services which are System-V style init script links.

Init scripts are in the `/etc/init.d/`

More information: https://manned.org/update-rc.d.

- Install a service:

  ```
  update-rc.d {{mysql}} defaults
  ```

- Enable a service:

  ```
  update-rc.d {{mysql}} enable
  ```

- Disable a service:

  ```
  update-rc.d {{mysql}} disable
  ```

- Forcibly remove a service:

  ```
  update-rc.d -f {{mysql}} remove
  ```

updatedb

Create or update the database used by `locate`.

It is usually run daily by cron.

More information: https://manned.org/updatedb.

- Refresh database content:

  ```
  sudo updatedb
  ```

- Display file names as soon as they are found:

  ```
  sudo updatedb --verbose
  ```

upower

System utility to provide power and battery information and statistics.

More information: https://upower.freedesktop.org/docs/upower.1.html

- Display power and battery information:

```
upower --dump
```

- List all power devices:

```
upower --enumerate
```

- Watch for and print power status changes:

```
upower --monitor
```

- Watch for and print detailed power status changes:

```
upower --monitor-detail
```

- Display version:

```
upower --version
```

uprecords

Displays a summary of historical uptime records.

- Display a summary of the top 1 0 historical uptime records:

  ```
  u precords
  ```

- Display the top 2 5 records:

  ```
  u precords - m {{2 5 }}
  ```

- Display the downtime between reboots instead of the kernel version:

  ```
  u precords - d
  ```

- S how the most recent reboots:

  ```
  u precords - B
  ```

- Don' t truncate information:

  ```
  u precords - w
  ```

urxvt

Rxvt-unicode.

A customizable terminal emulator.

- Open a new urxvt window:

  ```
  urxvt
  ```

- Run in a specific directory:

  ```
  urxvt -cd {{path/to/directory}}
  ```

- Run a command in a new urxvt window:

  ```
  urxvt -e {{command and}}
  ```

- Run a command and keep the window open:

  ```
  urxvt --hold -e {{command and}}
  ```

- Run a command within the `sh` shell:

  ```
  urxvt -e {{sh}} -c {{command and}}
  ```

useradd

Create a new user.

More information: https://manned.org/useradd.

- Create new user:

```
u seradd {{nam e}}
```

- Create new user with a default home directory:

```
u seradd - - create- hom e {{nam e}}
```

- Create new user with specified shell:

```
u seradd - - shell {{path/to/shell}} {{nam e}}
```

- Create new user belonging to additional groups (mind the lack of whitespace):

```
u seradd - - g rou ps {{g rou p1 , g rou p2 }} {{nam e}}
```

- Create new system user without a home directory:

```
u seradd - - no- create- hom e - - system  {{nam e}}
```

userdel

R emove a user account or remove a user from a group.

N ote: all commands must be executed as root.

More information: https://manned.org/userdel.

- R emove a user:

  ```
  u serdel {{nam e}}
  ```

- R emove a user along with their home directory and mail spool:

  ```
  u serdel - - rem ov e {{nam e}}
  ```

- R emove a user from a group:

  ```
  u serdel {{nam e}} {{g rou p}}
  ```

- R emove a user in other root directory:

  ```
  u serdel - - root {{path/to/other/root}} {{nam e}}
  ```

usermod

Modifies a user account.

More information: https://manned.org/usermod.

- Change a user's name:

```
usermod -l {{new name}} {{user}}
```

- Add user to supplementary groups (mind the whitespace):

```
usermod -a -G {{group1,group2}} {{user}}
```

- Create a new home directory for a user and move their files to it:

```
usermod -m -d {{path/to/home}} {{user}}
```

utmpdump

Dump and load btmp, utmp and wtmp accounting files.

- Dump the `/v ar/log /w tm p` file to the standard output as plain text:

  ```
  u tm pdu m p {{/v ar/log /w tm p}}
  ```

- Load a previously dumped file into `/v ar/log /w tm p`

  ```
  u tm pdu m p - r {{du m pfile}} >  {{/v ar/log /w tm p}}
  ```

uuid

Generate and decode Universally Unique Identifiers (UUID).

See also `uuidgen`

More information: https://manned.org/uuid.

- Generate a UUIDv1 (based on time and system's hardware address, if present):

  ```
  uuid
  ```

- Generate a UUIDv4 (based on random data):

  ```
  uuid -v {{4}}
  ```

- Generate multiple UUIDv4 identifiers at once:

  ```
  uuid -v {{4}} -n {{number_of_uuids}}
  ```

- Generate a UUIDv4 and specify the output format:

  ```
  uuid -v {{4}} -F {{BIN|STR|SIV}}
  ```

- Generate a UUIDv4 and write the output to a file:

  ```
  uuid -v {{4}} -o {{path/to/file}}
  ```

- Generate a UUIDv5 (based on the supplied object name) with a specified namespace prefix:

  ```
  uuid -v {{5}} ns:{{DNS|URL|OID|X500}} {{object_name}}
  ```

- Decode a given UUID:

  ```
  uuid -d {{uuid}}
  ```

uuidd

Daemon for generating U U IDs.

More information: https://manned.org/uuidd.

- G enerate a random U U ID:

```
u u idd - - random
```

- G enerate a bulk number of random U U IDs:

```
u u idd - - random  - - u u ids {{nu m b er_of_u u ids}}
```

- G enerate a time-based U U ID, based on the current time and MAC address of the system:

```
u u idd - - tim e
```

uuidgen

Generate unique identifiers (UUIDs).

See also: **uuid**

More information: https://manned.org/uuidgen.

- Create a random UUIDv4:

  ```
  uuidgen --random
  ```

- Create a UUIDv1 based on the current time:

  ```
  uuidgen --time
  ```

- Create a UUIDv5 of the name with a specified namespace prefix:

  ```
  uuidgen --sha1 --namespace {{@dns|@url|@oid|@x500}} --name
  {{object_name}}
  ```

uvcdynctrl

A libwebcam command-line tool to manage dynamic controls in uvcvideo.

- List all available cameras:

  ```
  uvcdynctrl -l
  ```

- Specify the device to use (defaults to `video0`):

  ```
  uvcdynctrl -d {{device_name}}
  ```

- List available controls:

  ```
  uvcdynctrl -c
  ```

- Set a new control value (for negative values, add -- before {{-value}}):

  ```
  uvcdynctrl -s {{control_name}} {{value}}
  ```

- Get the current control value:

  ```
  uvcdynctrl -g {{control_name}}
  ```

- Save the state of the current controls to a file:

  ```
  uvcdynctrl -W {{filename}}
  ```

- Load the state of the controls from a file:

  ```
  uvcdynctrl -L {{filename}}
  ```

veracrypt

Free and open source disk encryption software.

More information: https://www.veracrypt.fr/code/VeraCrypt/plain/doc/html/Documentation.html.

- Create a new volume through a text user interface and use `/dev/urandom` as a source of random data:

  ```
  veracrypt --text --create --random-source={{/dev/urandom}}
  ```

- Decrypt a volume interactively through a text user interface and mount it to a directory:

  ```
  veracrypt --text {{path/to/volume}} {{path/to/mount_point}}
  ```

- Decrypt a partition using a keyfile and mount it to a directory:

  ```
  veracrypt --keyfiles={{path/to/keyfile}} {{/dev/sdXN}} {{path/to/mount_point}}
  ```

- Dismount a volume on the directory it is mounted to:

  ```
  veracrypt --dismount {{path/to/mounted_point}}
  ```

vgchange

Change the attributes of a Logical Volume Manager (LVM) volume group.

See also: `lvm`.

More information: https://manned.org/vgchange.

- Change the activation status of logical volumes in all volume groups:

  ```
  sudo vgchange --activate {{y|n}}
  ```

- Change the activation status of logical volumes in the specified volume group (determine with `vgscan`):

  ```
  sudo vgchange --activate {{y|n}} {{volume_group}}
  ```

vgcreate

Create volume groups combining multiple mass-storage devices.

See also: `lvm`.

More information: https://man7.org/linux/man-pages/man8/vgcreate.8.html.

- Create a new volume group called vg1 using the `/dev/sda1` device:

  ```
  vg create {{vg1}} {{/dev/sda1}}
  ```

- Create a new volume group called vg1 using multiple devices:

  ```
  vg create {{vg1}} {{/dev/sda1}} {{/dev/sdb1}} {{/dev/sdc1}}
  ```

vgdisplay

Display information about Logical Volume Manager (LVM) volume groups.

See also: `lvm`.

More information: https://man7.org/linux/man-pages/man8/vgdisplay.8.html

- Display information about all volume groups:

  ```
  sudo vgdisplay
  ```

- Display information about volume group vg1:

  ```
  sudo vgdisplay {{vg1}}
  ```

vgs

Display information about volume groups.

See also: `lvm`.

More information: https://man7.org/linux/man-pages/man8/vgs.8.html

- Display information about volume groups:

  ```
  vgs
  ```

- Display all volume groups:

  ```
  vgs -a
  ```

- Change default display to show more details:

  ```
  vgs -v
  ```

- Display only specific fields:

  ```
  vgs -o {{field_name_1}},{{field_name_2}}
  ```

- Append field to default display:

  ```
  vgs -o +{{field_name}}
  ```

- Suppress heading line:

  ```
  vgs --noheadings
  ```

- Use separator to separate fields:

  ```
  vgs --separator=
  ```

vgscan

Scan for volume groups on all supported Logical Volume Manager (LVM) block devices.

See also: `lvm`, `vgchange`.

More information: https://manned.org/vgscan.

- Scan for volume groups and print information about each group found:

  `sudo vgscan`

- Scan for volume groups and add the special files in `/dev`, if they don't already exist, needed to access the logical volumes in the found groups :

  `sudo vgscan --mknodes`

viewnior

Simple and elegant image viewer.

- View an image:

  ```
  viewnior {{path/to/image.ext}}
  ```

- View in fullscreen mode:

  ```
  viewnior --fullscreen {{path/to/image.ext}}
  ```

- View fullscreen in slideshow mode:

  ```
  viewnior --slideshow {{path/to/image.ext}}
  ```

vipw

Edit the password file.

More information: https://manned.org/vipw.

- Edit the password file:

  ```
  vipw
  ```

- Display the current version of `vipw`

  ```
  vipw --version
  ```

virt-manager

CLI launcher for virt-manager, a desktop user interface for managing KVM and Xen virtual machines and LXC containers.

More information: https://manpages.ubuntu.com/manpages/man1/virt-manager.1.html

- Launch virt-manager:

```
virt-manager
```

- Connect to a hypervisor:

```
virt-manager --connect {{hypervisor_uri}}
```

- Don't fork virt-manager process into background on startup:

```
virt-manager --no-fork
```

- Print debug output:

```
virt-manager --debug
```

- Open the "New VM" wizard:

```
virt-manager --show-domain-creator
```

- Show domain details window:

```
virt-manager --show-domain-editor {{name|id|uuid}}
```

- Show domain performance window:

```
virt-manager --show-domain-performance {{name|id|uuid}}
```

- Show connection details window:

```
virt-manager --show-host-summary
```

vmstat

Report information about processes, memory, paging, block IO, traps, disks and CPU activity.

More information: https://manned.org/vmstat.

- Display virtual memory statistics:

```
vmstat
```

- Display reports every 2 seconds for 5 times:

```
vmstat {{2}} {{5}}
```

vmware-checkvm

Checks to see if the current host is a V Mware V M or not.

- R eturn the current V Mware software version (exit status determines whether the system is a V M or not):

  ```
  v m w are- check v m
  ```

- R eturn the V Mware hardware version:

  ```
  v m w are- check v m  - h
  ```

vncserver

Launches a V N C (V irtual N etwork Computing) desktop.

- Launch a V N C S erver on next available display:

  ```
  v ncserv er
  ```

- Launch a V N C S erver with specific screen geometry:

  ```
  v ncserv er - - g eom etry {{w idth}}x{{heig ht}}
  ```

- Kill an instance of V N C S erver running on a specific display:

  ```
  v ncserv er - - k ill : {{display_nu m b er}}
  ```

vncviewer

Launches a V N C (V irtual N etwork Computing) client.

- Launch a V N C client which connects to a host on a given display:

  ```
  v ncv iew er {{host}}: {{display_nu m b er}}
  ```

- Launch in full-screen mode:

  ```
  v ncv iew er - F u llS creen {{host}}: {{display_nu m b er}}
  ```

- Launch a V N C client with a specific screen geometry:

  ```
  v ncv iew er - - g eom etry {{w idth}}x{{heig ht}} {{host}}:
  {{display_nu m b er}}
  ```

- Launch a V N C client which connects to a host on a given port:

  ```
  v ncv iew er {{host}}: : {{port}}
  ```

vnstat

A console-based network traffic monitor.

- Display traffic summary for all interfaces:

  ```
  v nstat
  ```

- Display traffic summary for a specific network interface:

  ```
  v nstat - i {{eth0 }}
  ```

- Display live stats for a specific network interface:

  ```
  v nstat - l - i {{eth0 }}
  ```

- S how traffic statistics on an hourly basis for the last 2 4 hours using a bar graph:

  ```
  v nstat - hg
  ```

- Measure and show average traffic for 3 0 seconds:

  ```
  v nstat - tr {{3 0 }}
  ```

vnstati

PNG image output support for vnStat.

More information: https://manned.org/vnstati.

- Output a summary of the last 2: months, days, and all-time:

```
vnstati --summary --iface {{network_interface}} --output {{path/to/output.png}}
```

- Output the 10 most traffic-intensive days of all time:

```
vnstati --top10 --iface {{network_interface}} --output {{path/to/output.png}}
```

- Output monthly traffic statistics from the last 12 months:

```
vnstati --months --iface {{network_interface}} --output {{path/to/output.png}}
```

- Output hourly traffic statistics from the last 24 hours:

```
vnstati --hours --iface {{network_interface}} --output {{path/to/output.png}}
```

vpnc

A V P N client for the Cisco 3 0 0 0 V P N Concentrator.

- Connect with a defined configuration file:

  ```
  su do v pnc {{config _file}}
  ```

- T erminate the previously created connection:

  ```
  su do v pnc- disconnect
  ```

vrms

Report non-free packages installed on Debian-based OSes.

More information: https://debian.pages.debian.net/vrms/.

- List non-free and contrib packages (and their description):

  ```
  vrms
  ```

- Only output the package names:

  ```
  vrms --sparse
  ```

W

Display who is logged in and their processes.

More information: https://www.geeksforgeeks.org/w-command-in-linux-with-examples/.

- Display information about all users who are currently logged in:

```
w
```

- Display information about a specific user:

```
w {{user}}
```

- Display information without including the header:

```
w --no-header
```

- Display information without including the login, JCPU and PCPU columns:

```
w --short
```

wall

Write a message on the terminals of users currently logged in.

- Send a message:

  ```
  echo "{{message}}" | wall
  ```

- Send a message from a file:

  ```
  wall {{file}}
  ```

- Send a message with timeout (default 300):

  ```
  wall -t {{seconds}} {{file}}
  ```

watch

Execute a command repeatedly, and monitor the output in full-screen mode.

- Monitor files in the current directory:

  ```
  watch {{ls}}
  ```

- Monitor disk space and highlight the changes:

  ```
  watch -d {{df}}
  ```

- Monitor "node" processes, refreshing every 3 seconds:

  ```
  watch -n {{3}} "{{ps aux | grep node}}"
  ```

wg-quick

Quickly set up WireGuard tunnels based on config files.

More information: https://www.wireguard.com/quickstart/

- Set up a VPN tunnel:

```
wg-quick up {{interface_name}}
```

- Delete a VPN tunnel:

```
wg-quick down {{interface_name}}
```

wg

Manage the configuration of WireGuard interfaces.

More information: https://www.wireguard.com/quickstart/

- Check status of currently active interfaces:

```
sudo wg
```

- Print a new private key:

```
wg genkey
```

- Print a new public key:

```
echo {{private_key}} | wg pubkey
```

- Generate a public and private key:

```
wg genkey | tee {{privatekey.txt}} | wg pubkey > {{publickey.txt}}
```

whatis

Display one-line descriptions from manual pages.

- Display a description from a man page:

  ```
  w hatis {{com m and}}
  ```

- Don' t cut the description off at the end of the line:

  ```
  w hatis - - long {{com m and}}
  ```

- Display descriptions for all commands matching a glob:

  ```
  w hatis - - w ildcard {{net* }}
  ```

- S earch man page descriptions with a regular expression:

  ```
  w hatis - - reg ex ' {{w ish[ 0 - 9 ] \ . [ 0 - 9 ] }}'
  ```

whereis

Locate the binary, source, and manual page files for a command.

- Locate binary, source and man pages for ssh:

  ```
  w hereis {{ssh}}
  ```

- Locate binary and man pages for ls:

  ```
  w hereis - b m  {{ls}}
  ```

- Locate source of gcc and man pages for G it:

  ```
  w hereis - s {{g cc}} - m  {{g it}}
  ```

- Locate binaries for gcc in /u sr/b in only:

  ```
  w hereis - b  - B  {{/u sr/b in/}} - f {{g cc}}
  ```

- Locate unusual binaries (those that have more or less than one binary on the system):

  ```
  w hereis - u  *
  ```

- Locate binaries that have unusual manual entries (binaries that have more or less than one manual installed):

  ```
  w hereis - u  - m  *
  ```

whiptail

Display text-based dialog boxes from shell scripts.

- Display a simple message:

```
whiptail --title "{{title}}" --msgbox "{{message}}" {{height_in_chars}} {{width_in_chars}}
```

- Display a boolean choice, returning the result through the exit code:

```
whiptail --title "{{title}}" --yesno "{{message}}" {{height_in_chars}} {{width_in_chars}}
```

- Customise the text on the yes / no buttons:

```
whiptail --title "{{title}}" --yes-button "{{text}}" --no-button "{{text}}" --yesno "{{message}}" {{height_in_chars}} {{width_in_chars}}
```

- Display a text input box:

```
{{resulting_variable}}="$(whiptail --title "{{title}}" --inputbox "{{message}}" {{height_in_chars}} {{width_in_chars}} {{default_text}} 3>&1 1>&2 2>&3)"
```

- Display a password input box:

```
{{resulting_variable}}="$(whiptail --title "{{title}}" --passwordbox "{{message}}" {{height_in_chars}} {{width_in_chars}} 3>&1 1>&2 2>&3)"
```

- Display a multiple-choice menu:

```
{{resulting_variable}}=$(whiptail --title "{{title}}" --menu "{{message}}" {{height_in_chars}} {{width_in_chars}} {{menu_display_height}} "{{value_1}}" "{{display_text_1}}" "{{value_n}}" "{{display_text_n}}" ..... 3>&1 1>&2 2>&3
```

wifi-menu

Interactively connect to a wireless network.

More information: https://manned.org/wifi-menu.

- S et up a wireless connection interactively:

```
w ifi- m enu
```

- Interactively set up a connection to a network and obscure the password:

```
w ifi- m enu  - - ob scu re
```

- Display help:

```
w ifi- m enu  - - help
```

wine

Run Windows programs on Unix.

More information: https://wiki.winehq.org/

- Run `ipconfig.exe` program:

  `wine {{ipconfig}} {{/all}}`

- Run `cmd.exe` in background:

  `wine start {{cmd}}`

- Run Windows-like Package Manager:

  `wine uninstaller`

- Install MSI packages:

  `wine msiexec /i {{package}}`

winetricks

Manage Wine virtual Windows environments.

More information: https://wiki.winehq.org/Winetricks

- Start a graphical setup at the default Wine location:

  ```
  winetricks
  ```

- Specify a custom Wine directory to run Winetricks in:

  ```
  WINEPREFIX={{path/to/wine_directory}} winetricks
  ```

- Install a Windows DLL or component to the default Wine directory:

  ```
  winetricks {{package}}
  ```

wipefs

Wipe filesystem, raid, or partition-table signatures from a device.

- Display signatures for specified device:

  ```
  su do w ipefs {{/dev /sdX }}
  ```

- Wipe all available signatures for specified device:

  ```
  su do w ipefs - - all {{/dev /sdX }}
  ```

- P erform dry run:

  ```
  su do w ipefs - - all - - no- act {{/dev /sdX }}
  ```

- F orce wipe, even if the filesystem is mounted:

  ```
  su do w ipefs - - all - - force {{/dev /sdX }}
  ```

wmctrl

CLI for X Window Manager.

- List all windows, managed by the window manager:

  ```
  wmctrl -l
  ```

- Switch to the first window whose (partial) title matches:

  ```
  wmctrl -a {{window_title}}
  ```

- Move a window to the current workspace, raise it and give it focus:

  ```
  wmctrl -R {{window_title}}
  ```

- Switch to a workspace:

  ```
  wmctrl -s {{workspace_number}}
  ```

- Select a window and toggle fullscreen:

  ```
  wmctrl -r {{window_title}} -b toggle,fullscreen
  ```

- Select a window a move it to a workspace:

  ```
  wmctrl -r {{window_title}} -t {{workspace_number}}
  ```

wodim

Command (aliased as **`cdrecord`** on some systems) for recording data to CDs or DVDs.

Some invocations of wodim can cause destructive actions, such as erasing all the data on a disc.

- Display optical drives available to `wodim`:

  ```
  wodim --devices
  ```

- Record ("burn") an audio-only disc:

  ```
  wodim dev=/dev/{{optical_drive}} -audio {{track*.cdaudio}}
  ```

- Burn a file to a disc, ejecting the disc once done (some recorders require this):

  ```
  wodim -eject dev=/dev/{{optical_drive}} -data {{file.iso}}
  ```

- Burn a file to the disc in an optical drive, potentially writing to multiple discs in succession:

  ```
  wodim -tao dev=/dev/{{optical_drive}} -data {{file.iso}}
  ```

wol

Client for sending Wake-on-LAN magic packets.

More information: https://sourceforge.net/projects/wake-on-lan/

- Send a WoL packet to a device:

  ```
  wol {{mac_address}}
  ```

- Send a WoL packet to a device in another subnet based on its IP:

  ```
  wol --ipaddr={{ip_address}} {{mac_address}}
  ```

- Send a WoL packet to a device in another subnet based on its hostname:

  ```
  wol --host={{hostname}} {{mac_address}}
  ```

- Send a WoL packet to a specific port on a host:

  ```
  wol --port={{port_number}} {{mac_address}}
  ```

- Read hardware addresses, IP addresses/hostnames, optional ports and SecureON passwords from a file:

  ```
  wol --file={{path/to/file}}
  ```

- Turn on verbose output:

  ```
  wol --verbose {{mac_address}}
  ```

wpa_cli

Add and configure wlan interfaces.

- Scan for available networks:

  ```
  wpa_cli scan
  ```

- Show scan results:

  ```
  wpa_cli scan_results
  ```

- Add a network:

  ```
  wpa_cli add_network {{number}}
  ```

- Set a network's SSID:

  ```
  wpa_cli set_network {{number}} ssid "{{SSID}}"
  ```

- Enable network:

  ```
  wpa_cli enable_network {{number}}
  ```

- Save config:

  ```
  wpa_cli save_config
  ```

wpa_ passphrase

G enerate a WP A-P S K key from an AS CII passphrase for a given S S ID.

- Compute and display the WP A-P S K key for a given S S ID reading the passphrase from stdin:

  ```
  w pa_passphrase {{S S I D }}
  ```

- Compute and display WP A-P S K key for a given S S ID specifying the passphrase as an argument:

  ```
  w pa_passphrase {{S S I D }} {{passphrase}}
  ```

wtf

Show the expansions of acronyms.

More information: https://manpages.debian.org/bsdgames/wtf.6.en.html

- Expand a given acronym:

  ```
  wtf {{IMO}}
  ```

- Specify a computer related search type:

  ```
  wtf -t {{comp}} {{WWW}}
  ```

x0 vncserver

TigerVNC Server for X displays.

More information: https://tigervnc.org/doc/x0vncserver.html

- Start a VNC server using a passwordfile:

  ```
  x0vncserver -display {{:0}} -passwordfile {{path/to/file}}
  ```

- Start a VNC server using a specific port:

  ```
  x0vncserver -display {{:0}} -rfbport {{port}}
  ```

x1 1 vnc

A V N C server that will enable V N C on an existing display server.

B y default, the server will automatically terminate once all clients disconnect from it.

- Launch a V N C server that allows multiple clients to connect:

  ```
  x1 1 v nc - shared
  ```

- Launch a V N C server in view-only mode, and which won' t terminate once the last client disconnects:

  ```
  x1 1 v nc - forev er - v iew only
  ```

- Launch a V N C server on a specific display and screen (both starting at index z ero):

  ```
  x1 1 v nc - display : {{display}}. {{screen}}
  ```

- Launch a V N C server on the third display' s default screen:

  ```
  x1 1 v nc - display : {{2 }}
  ```

- Launch a V N C server on the first display' s second screen:

  ```
  x1 1 v nc - display : {{0 }}. {{1 }}
  ```

xar

Manage .xar archives.

- Create a xar archive of all files in a given directory:

  ```
  xar -cf {{archive.xar}} {{path/to/directory}}
  ```

- List the contents of a given xar archive:

  ```
  xar -tf {{archive.xar}}
  ```

- Extract the contents of a given xar archive to the current directory:

  ```
  xar -xf {{archive.xar}}
  ```

xbacklight

Utility to adjust backlight brightness using the RandR extension.

More information: https://gitlab.freedesktop.org/xorg/app/xbacklight.

- Get the current screen brightness as a percentage:

```
xbacklight
```

- Set the screen brightness to 40%:

```
xbacklight -set {{40}}
```

- Increase current brightness by 25%:

```
xbacklight -inc {{25}}
```

- Decrease current brightness by 75%:

```
xbacklight -dec {{75}}
```

- Increase backlight to 100%, over 60 seconds (value given in ms), using 60 steps:

```
xbacklight -set {{100}} -time {{60000}} -steps {{60}}
```

xbps

The X Binary Package System (or xbps) is the binary package system used by Void Linux.

More information: https://github.com/void-linux/xbps.

- Install packages and synchronize them with the remote repository:

  ```
  xbps-install --synchronize {{package name1}} {{package name2}}
  ```

- Search for a package in the remote repository:

  ```
  xbps-query --repository -s {{package name}}
  ```

- Remove a package, leaving all of its dependencies installed:

  ```
  xbps-remove {{package name}}
  ```

- Remove a package and all of its dependencies recursively that are not required by other packages:

  ```
  xbps-remove --recursive {{package name}}
  ```

- Synchronize your repository databases and update your system and dependencies:

  ```
  xbps-install --synchronize -u
  ```

- Remove packages that were installed as dependencies and aren't currently needed:

  ```
  xbps-remove --remove-orphans
  ```

- Remove obsolete packages from the cache:

  ```
  xbps-remove --clean-cache
  ```

xclip

X 1 1 clipboard manipulation tool, similar to `xsel`.

Handles the X primary and secondary selections, plus the system clipboard (`Ctrl+C`/`Ctrl+V`).

- Copy the output from a command to the X 1 1 primary selection area (clipboard):

  ```
  echo 123 | xclip
  ```

- Copy the output from a command to a given X 1 1 selection area:

  ```
  echo 123 | xclip -selection {{primary|secondary|clipboard}}
  ```

- Copy the output from a command to the system clipboard, using short notation:

  ```
  echo 123 | xclip -sel clip
  ```

- Copy the contents of a file into the system clipboard:

  ```
  xclip -sel clip {{input_file.txt}}
  ```

- Copy the contents of a PNG image into the system clipboard (can be pasted in other programs correctly):

  ```
  xclip -sel clip -t image/png {{input_file.png}}
  ```

- Copy the user input in the console into the system clipboard:

  ```
  xclip -i
  ```

- Paste the contents of the X 1 1 primary selection area to the console:

  ```
  xclip -o
  ```

- Paste the contents of the system clipboard to the console:

  ```
  xclip -o -sel clip
  ```

xclock

Display the time in analog or digital form.

- Display an analog clock:

```
xclock
```

- Display a 24-hour digital clock with the hour and minute fields only:

```
xclock -digital -brief
```

- Display a digital clock using an strftime format string (see strftime(3)):

```
xclock -digital -strftime {{format}}
```

- Display a 24-hour digital clock with the hour, minute and second fields that updates every second:

```
xclock -digital -strftime '%H:%M:%S' -update 1
```

- Display a 12-hour digital clock with the hour and minute fields only:

```
xclock -digital -twelve -brief
```

xcursorgen

Create an X cursor file from a collection of P N G images.

If `- - pref ix` is omitted, the image files must be located in the current working directory.

More information: https://manned.org/xcursorgen.1.

- Create an X cursor file using a config file:

  ```
  xcu rsorg en {{path/to/config . cu rsor}} {{path/to/ou tpu t_file}}
  ```

- Create an X cursor file using a config file and specify the path to the image files:

  ```
  xcu rsorg en - - prefix {{path/to/im ag e_directory/}} {{path/to/
  config . cu rsor}} {{path/to/ou tpu t_file}}
  ```

- Create an X cursor file using a config file and write the output to stdout:

  ```
  xcu rsorg en {{path/to/config . cu rsor}}
  ```

xdg-mime

Query and manage MIME types according to the XDG standard.

More information: https://portland.freedesktop.org/doc/xdg-mime.html.

- Display the MIME type of a file:

```
xdg-mime query filetype {{path/to/file}}
```

- Display the default application for opening PNG images:

```
xdg-mime query default {{image/png}}
```

- Display the default application for opening a specific file:

```
xdg-mime query default $(xdg-mime query filetype {{path/to/file}})
```

- Set imv as the default application for opening PNG and JPEG images:

```
xdg-mime default {{imv.desktop}} {{image/png}} {{image/jpeg
```

xdg-open

Opens a file or URL in the user's preferred application.

More information: https://man.archlinux.org/man/xdg-open.1.

- Open the current directory in the default file explorer:

```
xdg-open .
```

- Open an URL in the default browser:

```
xdg-open {{https://example.com}}
```

- Open an image in the default image viewer:

```
xdg-open {{path/to/image}}
```

- Open a PDF in the default PDF viewer:

```
xdg-open {{path/to/pdf}}
```

- Display help:

```
xdg-open --help
```

xdotool

Command-line automation for X 1 1 .

- R etrieve the X -Windows window ID of the running F irefox window(s):

  ```
  xdotool search - - onlyv isib le - - nam e {{firefox}}
  ```

- Click the right mouse button:

  ```
  xdotool click {{3 }}
  ```

- G et the id of the currently active window:

  ```
  xdotool g etactiv ew indow
  ```

- F ocus on the window with id of 1 2 3 4 5 :

  ```
  xdotool w indow focu s - - sync {{1 2 3 4 5 }}
  ```

- T ype a message, with a 5 0 0 ms delay for each letter:

  ```
  xdotool type - - delay {{5 0 0 }} " H ello w orld"
  ```

- P ress the enter key:

  ```
  xdotool k ey {{K P _E nter}}
  ```

xeyes

Display eyes on the screen that follow the mouse cursor.

- Launch xeyes on the local machine' s default display:

  ```
  xeyes
  ```

- Launch xeyes on a remote machine' s display 0 , screen 0 :

  ```
  xeyes - display {{rem ote_host}}: {{0 }}. {{0 }}
  ```

xfce4 -screenshooter

The XFCE4 screenshot tool.

More information: https://docs.xfce.org/apps/xfce4 -screenshooter/start

- Launch the screenshooter GUI:

  ```
  xfce4 - screenshooter
  ```

- Take a screenshot of the entire screen and launch the GUI to ask how to proceed:

  ```
  xfce4 - screenshooter - - fu llscreen
  ```

- Take a screenshot of the entire screen and save it in the specified directory:

  ```
  xfce4 - screenshooter - - fu llscreen - - sav e {{path/to/directory}}
  ```

- Wait some time before taking the screenshot:

  ```
  xfce4 - screenshooter - - delay {{seconds}}
  ```

- Take a screenshot of a region of the screen (select using the mouse):

  ```
  xfce4 - screenshooter - - reg ion
  ```

- Take a screenshot of the active window, and copy it to the clipboard:

  ```
  xfce4 - screenshooter - - w indow  - - clipb oard
  ```

- Take a screenshot of the active window, and open it with a chosen program:

  ```
  xfce4 - screenshooter - - w indow  - - open {{g im p}}
  ```

xfce4-terminal

The XFCE4 terminal emulator.

More information: https://docs.xfce.org/apps/xfce4-terminal/start

- Open a new terminal window:

  ```
  xfce4-terminal
  ```

- Set the initial title:

  ```
  xfce4-terminal --initial-title "{{initial_title}}"
  ```

- Open a new tab in the current terminal window:

  ```
  xfce4-terminal --tab
  ```

- Execute a command in a new terminal window:

  ```
  xfce4-terminal --command "{{command_with_args}}"
  ```

- Keep the terminal around after the executed command finishes executing:

  ```
  xfce4-terminal --command "{{command_with_args}}" --hold
  ```

- Open multiple new tabs, executing a command in each:

  ```
  xfce4-terminal --tab --command "{{command_a}}" --tab --command "{{command_b}}"
  ```

xinput

List available input devices, query information about a device and change input device settings.

- List all input devices:

  ```
  xinput list
  ```

- Disable an input:

  ```
  xinput disable {{id}}
  ```

- Enable an input:

  ```
  xinput enable {{id}}
  ```

- Disconnect an input from its master:

  ```
  xinput float {{id}}
  ```

- Reattach an input as slave to a master:

  ```
  xinput reattach {{id}} {{master_id}}
  ```

- List settings of an input device:

  ```
  xinput list-props {{id}}
  ```

- Change a setting of an input device:

  ```
  xinput set-prop {{id}} {{setting_id}} {{value}}
  ```

xman

Manual page viewer for X Window System.

- Start xman in three-button window:

  ```
  xman
  ```

- Open the manual page output stored in a given file:

  ```
  xman -helpfile {{filename}}
  ```

- Show both manual page and directory:

  ```
  xman -bothshown
  ```

xmount

Convert on-the-fly between multiple input and output hard disk image types with optional write cache support.

Creates a virtual file system using FUSE (Filesystem in Userspace) that contains a virtual representation of the input image.

More information: https://manned.org/xmount.

- Mount a `.raw` image file into a DMG container file:

  ```
  xmount --in {{raw}} {{path/to/image.dd}} --out {{dmg}} {{mountpoint}}
  ```

- Mount an EWF image file with write-cache support into a VHD file to boot from:

  ```
  xmount --cache {{path/to/cache.ovl}} --in {{ewf}} {{path/to/image.E??}} --out {{vhd}} {{mountpoint}}
  ```

- Mount the first partition at sector 2048 into a new `.raw` image file:

  ```
  xmount --offset {{2048}} --in {{raw}} {{path/to/image.dd}} --out {{raw}} {{mountpoint}}
  ```

xrandr

Set the size, orientation and/or reflection of the outputs for a screen.

- Display the current state of the system (known screens, resolutions, ...):

```
xrandr --query
```

- Disable disconnected outputs and enable connected ones with default settings:

```
xrandr --auto
```

- Change the resolution and update frequency of DisplayPort 1 to 1920x1080,60H

```
xrandr --output {{DP1}} --mode {{1920x1080}} --rate {{60
```

- Set the resolution of HDMI2 to 1280x1024 and put it on the right of DP1:

```
xrandr --output {{HDMI2}} --mode {{1280x1024}} --right
{{DP1}}
```

- Disable the VGA1 output:

```
xrandr --output {{VGA1}} --off
```

- Set brightness for LVDS1 to 50%:

```
xrandr --output {{LVDS1}} --brightness {{0.5}}
```

- See display hardware information:

```
xrandr -q
```

xsel

X11 selection and clipboard manipulation tool.

- Use a command's output as input of the clip[b]oard (equivalent to `Ctrl + C`):

  ```
  echo 123 | xsel -ib
  ```

- Use the contents of a file as input of the clipboard:

  ```
  cat {{file}} | xsel -ib
  ```

- Output the clipboard's contents into the terminal (equivalent to `Ctrl + V`):

  ```
  xsel -ob
  ```

- Output the clipboard's contents into a file:

  ```
  xsel -ob > {{file}}
  ```

- Clear the clipboard:

  ```
  xsel -cb
  ```

- Output the X11 primary selection's contents into the terminal (equivalent to a mouse middle-click):

  ```
  xsel -op
  ```

xsetwacom

Command-line tool to change settings for Wacom pen tablets at runtime.

- List all the available wacom devices. The device name is in the first column:

  ```
  xsetwacom list
  ```

- Set Wacom area to specific screen. Get name of the screen with `xrandr`:

  ```
  xsetwacom set "{{device name}}" MapToOutput {{screen}}
  ```

- Set mode to relative (like a mouse) or absolute (like a pen) mode:

  ```
  xsetwacom set "{{device name}}" Mode "{{Relative|Absolute}}"
  ```

- Rotate the input (useful for tablet-PC when rotating screen) by 0|90|180|270 degrees from "natural" rotation:

  ```
  xsetwacom set "{{device name}}" Rotate {{none|half|cw|ccw}}
  ```

- Set button to only work when the tip of the pen is touching the tablet:

  ```
  xsetwacom set "{{device name}}" TabletPCButton "on"
  ```

xterm

A terminal emulator for the X Window System.

- Open the terminal with a title of `Example`:

  ```
  xterm -T {{Example}}
  ```

- Open the terminal in fullscreen mode:

  ```
  xterm -fullscreen
  ```

- Open the terminal with a dark blue background and yellow foreground (font color):

  ```
  xterm -bg {{dark blue}} -fg {{yellow}}
  ```

- Open the terminal with 100 characters per line and 35 lines, in screen position x=200 px, y=20 px:

  ```
  xterm -geometry {{100}}x{{35}}+{{200}}+{{20}}
  ```

- Open the terminal using a Serif font and a font size equal to 20:

  ```
  xterm -fa {{'Serif'}} -fs {{20}}
  ```

xtrlock

Lock the X display until the user supplies their password.

- Lock the display and show a padlock instead of the cursor:

  ```
  xtrlock
  ```

- Display a blank screen as well as the padlock cursor:

  ```
  xtrlock -b
  ```

- Fork the xtrlock process and return immediately:

  ```
  xtrlock -f
  ```

xvfb-run

Run a command in a virtual X server environment.

More information: https://www.x.org/wiki/.

- Run the specified command in a virtual X server:

```
xvfb-run {{command}}
```

- Try to get a free server number, if the default (99) is not available:

```
xvfb-run --auto-servernum {{command}}
```

- Pass arguments to the X vfb server:

```
xvfb-run --server-args="{{-screen 0 1024x768x24}}" {{command}}
```

yank

Read input from stdin and display a selection interface that allows a field to be selected and copied to the clipboard.

- Yank using the default delimiters (\f, \n, \r, \s, \t):

  ```
  {{sudo dmesg}} | yank
  ```

- Yank an entire line:

  ```
  {{sudo dmesg}} | yank -l
  ```

- Yank using a specific delimiter:

  ```
  {{echo hello=world}} | yank -d {{=}}
  ```

- Only yank fields matching a specific pattern:

  ```
  {{ps ux}} | yank -g "{{[0-9]+}}"
  ```

yaourt

Arch Linux utility for building packages from the Arch User Repository.

- Synchronize and update all packages (including AUR):

  ```
  yaourt -Syua
  ```

- Install a new package (includes AUR):

  ```
  yaourt -S {{package name}}
  ```

- Remove a package and its dependencies (includes AUR packages):

  ```
  yaourt -Rs {{package name}}
  ```

- Search the package database for a keyword (including AUR):

  ```
  yaourt -Ss {{package name}}
  ```

- List installed packages, versions, and repositories (AUR packages will be listed under the repository name 'local'):

  ```
  yaourt -Q
  ```

yay

Y et Another Y ogurt: A utility for Arch Linux to build and install packages from the Arch U ser R epository.

Also see **pa cma .n**

More information: https://github.com/J guer/yay

- Interactively search and install packages from the repos and AU R :

  ```
  yay {{pack ag e_nam e| search_term }}
  ```

- S ynchroniz e and update all packages from the repos and AU R :

  ```
  yay
  ```

- S ynchroniz e and update only AU R packages:

  ```
  yay - S u a
  ```

- Install a new package from the repos and AU R :

  ```
  yay - S {{pack ag e_nam e}}
  ```

- R emove an installed package and both its dependencies and configuration files:

  ```
  yay - R ns {{pack ag e_nam e}}
  ```

- S earch the package database for a keyword from the repos and AU R :

  ```
  yay - S s {{k eyw ord}}
  ```

- S how statistics for installed packages and system health:

  ```
  yay - P s
  ```

yetris

Clone of the game T etris in the terminal.

More information: https://github.com/alexdantas/yetris.

- S tart a tetris game:

  ```
  yetris
  ```

- N avigate the piece horiz ontally:

  ```
  {{L eft| R ig ht}} arrow k ey
  ```

- R otate the piece clockwise or counterclockwise:

  ```
  {{x| z}}
  ```

- H old a piece (only one allowed at a time):

  ```
  c
  ```

- S oft drop the piece:

  ```
  D ow n arrow k ey
  ```

- H ard drop the piece:

  ```
  S paceb ar
  ```

- P ause/unpause the game:

  ```
  p
  ```

- Q uit the game:

  ```
  q
  ```

yum

Package management utility for RHEL, Fedora, and CentOS (for older versions).

More information: https://man7.org/linux/man-pages/man8/yum.8.html

- Install a new package:

```
yum install {{package}}
```

- Install a new package and assume yes to all questions (also works with update, great for automated updates):

```
yum -y install {{package}}
```

- Find the package that provides a particular command:

```
yum provides {{command}}
```

- Remove a package:

```
yum remove {{package}}
```

- Display available updates for installed packages:

```
yum check-update
```

- Upgrade installed packages to newest available versions:

```
yum upgrade
```

z enity

Display dialogs from the command-line/shell scripts.

R eturn user-inserted values or 1 if error.

- Display the default q uestion dialog:

```
zenity - - q u estion
```

- Display an info dialog displaying the text " H ello! " :

```
zenity - - info - - text= " {{H ello! }}"
```

- Display a name/password form and output the data separated by " ; " :

```
zenity - - form s - - add- entry= " {{N am e}}" - - add-
passw ord= " {{P assw ord}}" - - separator= " {{; }}"
```

- Display a file selection form in which the user can only select directories:

```
zenity - - file- selection - - directory
```

- Display a progress bar which updates its message every second and show a progress percent:

```
{{( echo " # 1 " ; sleep 1 ; echo " 5 0 " ; echo " # 2 " ; sleep 1 ; ech
" 1 0 0 " ) }} | zenity - - prog ress
```

zgrep

Grep text patterns from files within compressed file (equivalent to grep -Z).

- Grep a pattern in a compressed file (case-sensitive):

```
zgrep {{pattern}} {{path/to/compressed/file}}
```

- Grep a pattern in a compressed file (case-insensitive):

```
zgrep -i {{pattern}} {{path/to/compressed/file}}
```

- Output count of lines containing matched pattern in a compressed file:

```
zgrep -c {{pattern}} {{path/to/compressed/file}}
```

- Display the lines which don't have the pattern present (Invert the search function):

```
zgrep -v {{pattern}} {{path/to/compressed/file}}
```

- Grep a compressed file for multiple patterns:

```
zgrep -e "{{pattern_1}}" -e "{{pattern_2}}" {{path/to/compressed/file}}
```

- Use extended regular expressions (supporting `?`, `+`, `{}`, `()` and `|`):

```
zgrep -E {{regular_expression}} {{path/to/file}}
```

- Print 3 lines of [C]ontext around, [B]efore, or [A]fter each match:

```
zgrep -{{C|B|A}} {{3}} {{pattern}} {{path/to/compressed/file}}
```

z ile

Z ile is a lightweight clone of the E macs text editor.

More information: https://www.gnu.org/software/z ile/

- S tart a buffer for temporary notes, which won' t be saved:

  ```
  zile
  ```

- O pen a file:

  ```
  zile {{path/to/file}}
  ```

- S ave a file:

  ```
  C trl + X , C trl + S
  ```

- Q uit:

  ```
  C trl + X , C trl + C
  ```

- O pen a file at a specified line number:

  ```
  zile + {{line_nu m b er}} {{path/to/file}}
  ```

- U ndo changes:

  ```
  C trl + X , U
  ```

zramctl

Setup and control zram devices.

Use `mkfs` or `mkswap` to format zram devices to partitions.

- Check if zram is enabled:

  ```
  lsmod | grep -i zram
  ```

- Enable zram with a dynamic number of devices (use `zramctl` to configure devices further):

  ```
  sudo modprobe zram
  ```

- Enable zram with exactly 2 devices:

  ```
  sudo modprobe zram num_devices={{2}}
  ```

- Find and initialise the next free zram device to a 2 GB virtual drive using LZ4 compression:

  ```
  sudo zramctl --find --size {{2GB}} --algorithm {{lz4}}
  ```

- List currently initialised devices:

  ```
  zramctl
  ```

zypper

SUSE & openSUSE package management utility.

More information: https://en.opensuse.org/SDB:Zypper_manual

- Synchronize list of packages and versions available:

  ```
  zypper refresh
  ```

- Install a new package:

  ```
  zypper install {{package}}
  ```

- Remove a package:

  ```
  zypper remove {{package}}
  ```

- Upgrade installed packages to newest available versions:

  ```
  zypper update
  ```

- Search package via keyword:

  ```
  zypper search {{keyword}}
  ```

- Show information related to configured repositories:

  ```
  zypper repos --sort-by-priority
  ```

O sx

afinfo

Audio file metadata parser for O S X .

B uilt-in command of O S X .

- Display info of a given audio file:

  ```
  afinfo {{path/to/file}}
  ```

- P rint a one line description of the audio file:

  ```
  afinfo - b {{path/to/file}}
  ```

- P rint metadata info and contents of the audio file' s InfoDictionary:

  ```
  afinfo - i {{path/to/file}}
  ```

- P rint output in xml format:

  ```
  afinfo - x {{path/to/file}}
  ```

- P rint warnings for the audio file if any:

  ```
  afinfo - - w arning s {{path/to/file}}
  ```

- Display help for full usage:

  ```
  afinfo - h
  ```

afplay

Command-line audio player.

- Play a sound file (waits until playback ends):

  ```
  afplay {{path/to/file}}
  ```

- Play a sound file at 2x speed (playback rate):

  ```
  afplay --rate {{2}} {{path/to/file}}
  ```

- Play a sound file at half speed:

  ```
  afplay --rate {{0.5}} {{path/to/file}}
  ```

- Play the first N seconds of a sound file:

  ```
  afplay --time {{seconds}} {{path/to/file}}
  ```

airport

Wireless network configuration utility.

- S how current wireless status information:

  ```
  airport -I
  ```

- S niff wireless traffic on channel 1 :

  ```
  airport sniff {{1}}
  ```

- S can for available wireless networks:

  ```
  airport -s
  ```

- Disassociate from current airport network:

  ```
  sudo airport -z
  ```

apachectl

Apache HTTP Server control interface for macOS.

- Start the `org.apache.httpd` launchd job:

  ```
  apachectl start
  ```

- Stop the launchd job:

  ```
  apachectl stop
  ```

- Stop, then start launchd job:

  ```
  apachectl restart
  ```

arch

Display the name of the system architecture, or run a command under a different architecture.

See also: **uname**

- Display the system's architecture:

  ```
  arch
  ```

- Run a command using x86_64:

  ```
  arch -x86_64 {{command}}
  ```

archey

Simple tool for stylishly displaying system information.

- Show system information:

  ```
  archey
  ```

- Show system information without colored output:

  ```
  archey --nocolor
  ```

- Show system information, using MacPorts instead of Homebrew:

  ```
  archey --macports
  ```

- Show system information without IP address check:

  ```
  archey --offline
  ```

as

Portable GNU assembler.

Primarily intended to assemble output from **`gcc`** to be used by **`ld`**.

- Assemble a file, writing the output to `a.out`:

  ```
  as {{file.s}}
  ```

- Assemble the output to a given file:

  ```
  as {{file.s}} -o {{out.o}}
  ```

- Generate output faster by skipping whitespace and comment preprocessing. (Should only be used for trusted compilers):

  ```
  as -f {{file.s}}
  ```

- Include a given path to the list of directories to search for files specified in `.include` directives:

  ```
  as -I {{path/to/directory}} {{file.s}}
  ```

asr

Restore (copy) a disk image onto a volume.

The command name stands for Apple Software Restore.

- Restore a disk image to a target volume:

  ```
  sudo asr restore --source {{image_name}}.dmg --target {{path/to/volume}}
  ```

- Erase the target volume before restoring:

  ```
  sudo asr restore --source {{image_name}}.dmg --target {{path/to/volume}} --erase
  ```

- Skip verification after restoring:

  ```
  sudo asr restore --source {{image_name}}.dmg --target {{path/to/volume}} --noverify
  ```

- Clone volumes without the use of an intermediate disk image:

  ```
  sudo asr restore --source {{path/to/volume}} --target {{path/to/cloned_volume}}
  ```

base64

Encode and decode using Base64 representation.

- Encode a file:

```
base64 --input={{plain_file}}
```

- Decode a file:

```
base64 --decode --input={{base64_file}}
```

- Encode from stdin:

```
echo -n {{plain_text}} | base64
```

- Decode from stdin:

```
echo -n {{base64_text}} | base64 --decode
```

bless

Set volume boot capability and startup disk options.

More information: https://ss64.com/osx/bless.html

- Bless a volume with only Mac OS X or Darwin, and create the BootX and `boot.efi` files as needed:

  ```
  bless --folder "{{/Volumes/Macintosh HD/System/Library/CoreServices}}" --bootinfo --bootefi
  ```

- Set a volume containing either Mac OS 9 and Mac OS X to be the active volume:

  ```
  bless --mount "{{/Volumes/Macintosh HD}}" --setBoot
  ```

- Set the system to NetBoot and broadcast for an available server:

  ```
  bless --netboot --server {{bsdp://255.255.255.255}}
  ```

- Gather information about the currently selected volume (as determined by the firmware), suitable for piping to a program capable of parsing Property Lists:

  ```
  bless --info --plist
  ```

brew bundle

Bundler for Homebrew, Homebrew Cask and the Mac App Store.

More information: https://github.com/Homebrew/homebrew-bundle

- Install packages from a Brewfile at the current path:

  ```
  brew bundle
  ```

- Install packages from a specific Brewfile at a specific path:

  ```
  brew bundle --file={{path/to/file}}
  ```

- Create a Brewfile from all installed packages:

  ```
  brew bundle dump
  ```

- Uninstall all formulae not listed in the Brewfile:

  ```
  brew bundle cleanup --force
  ```

- Check if there is anything to install or upgrade in the Brewfile:

  ```
  brew bundle check
  ```

- Output a list of all entries in the Brewfile:

  ```
  brew bundle list --all
  ```

brew cask

Package manager for macOS applications distributed as binaries.

More information: https://github.com/Homebrew/homebrew-cask

- Search for formulas and casks:

```
brew search {{text}}
```

- Install a cask:

```
brew cask install {{cask_name}}
```

- List all installed casks:

```
brew list --cask
```

- List installed casks that have newer versions available:

```
brew outdated --cask
```

- Upgrade an installed cask (if no cask name is given, all installed casks are upgraded):

```
brew upgrade --cask {{cask_name}}
```

- Uninstall a cask:

```
brew cask uninstall {{cask_name}}
```

- Uninstall a cask and remove related settings and files:

```
brew cask zap {{cask_name}}
```

- Display information about a given cask:

```
brew cask info {{cask_name}}
```

brightness

G et and set the brightness level of all internal and certain external displays.

- S how current brightness:

  ```
  b rig htness - l
  ```

- S et the brightness to 1 0 0 % ::

  ```
  b rig htness {{1 }}
  ```

- S et the brightness to 5 0 % ::

  ```
  b rig htness {{0 . 5 }}
  ```

caffeinate

Prevent mac from sleeping.

- Prevent from sleeping for 1 hour (3600 seconds):

  ```
  caffeinate -u -t {{3600}}
  ```

- Prevent from sleeping until a command completes:

  ```
  caffeinate -s {{command}}
  ```

- Prevent from sleeping until you type Ctrl-C:

  ```
  caffeinate -i
  ```

cal

Prints calendar information.

- Display a calendar for the current month:

  ```
  cal
  ```

- Display previous, current and next month:

  ```
  cal -3
  ```

- Display a calendar for a specific month (1-12 or name):

  ```
  cal -m {{month}}
  ```

- Display a calendar for the current year:

  ```
  cal -y
  ```

- Display a calendar for a specific year (4 digits):

  ```
  cal {{year}}
  ```

- Display a calendar for a specific month and year:

  ```
  cal {{month}} {{year}}
  ```

- Display date of Easter (Western Christian churches) in a given year:

  ```
  ncal -e {{year}}
  ```

carthage

A dependency management tool for Cocoa applications.

- Download the latest version of all dependencies mentioned in Cartfile, and build them:

  ```
  carthage update
  ```

- Update dependencies, but only build for iOS:

  ```
  carthage update --platform ios
  ```

- Update dependencies, but don't build any of them:

  ```
  carthage update --no-build
  ```

- Download and rebuild the current version of dependencies (without updating them):

  ```
  carthage bootstrap
  ```

- Rebuild a specific dependency:

  ```
  carthage build {{dependency}}
  ```

chflags

Change file or directory flags.

- Set the `hidden` flag for a file:

  ```
  chflags {{hidden}} {{path/to/file}}
  ```

- Unset the `hidden` flag for a file:

  ```
  chflags {{nohidden}} {{path/to/file}}
  ```

- Recursively set the `uchg` flag for a directory:

  ```
  chflags -R {{uchg}} {{path/to/directory}}
  ```

- Recursively unset the `uchg` flag for a directory:

  ```
  chflags -R {{nouchg}} {{path/to/directory}}
  ```

codesign

Create and manipulate code signatures for macOS.

- Sign an application with a certificate:

  ```
  codesign -s "{{My Company Name}}" {{path/to/App.app}}
  ```

- Verify the certificate of an application:

  ```
  codesign -v {{path/to/App.app}}
  ```

compgen

A built-in command for auto-completion in bash, which is called on pressing TAB key twice.

- List all commands that you could run:

  ```
  compgen -c
  ```

- List all aliases:

  ```
  compgen -a
  ```

- List all functions that you could run:

  ```
  compgen -A function
  ```

- Show shell reserved key words:

  ```
  compgen -k
  ```

- See all available commands/aliases starting with 'ls':

  ```
  compgen -ac {{ls}}
  ```

csshX

Cluster S S H tool for macO S .

More information: https://github.com/brockgr/csshx.

- Connect to multiple hosts:

```
csshX {{hostnam e1 }} {{hostnam e2 }}
```

- Connect to multiple hosts with a given S S H key:

```
csshX {{u ser@ hostnam e1 }} {{u ser@ hostnam e2 }} ' - - ssh_arg s' ' -
{{path/to/ssh_k ey. pem }}'
```

- Connect to a pre-defined cluster from `/etc/clu sters`

```
csshX clu ster1
```

dark-mode

Control macOS dark mode from the command-line.

More information: https://github.com/sindresorhus/dark-mode.

- Toggle dark mode (turn it on if it's currently off, off if it's currently on):

  ```
  dark-mode
  ```

- Turn dark mode on:

  ```
  dark-mode on
  ```

- Turn dark mode off:

  ```
  dark-mode off
  ```

- Check if dark mode is on:

  ```
  dark-mode status
  ```

date

Set or display the system date.

- Display the current date using the default locale's format:

```
date +"%c"
```

- Display the current date in UTC and ISO 8601 format:

```
date -u +"%Y-%m-%dT%H:%M:%SZ"
```

- Display the current date as a Unix timestamp (seconds since the Unix epoch):

```
date +%s
```

- Display a specific date (represented as a Unix timestamp) using the default format:

```
date -r 1473305798
```

dd

Convert and copy a file.

- Make a bootable usb drive from an isohybrid file (such like `archlinux-xxx.iso`):

  ```
  dd if={{file.iso}} of=/dev/{{usb_drive}}
  ```

- Clone a drive to another drive with 4MB block and ignore error:

  ```
  dd if=/dev/{{source_drive}} of=/dev/{{dest_drive}} bs=4M conv=noerror
  ```

- Generate a file of 100 random bytes by using kernel random driver:

  ```
  dd if=/dev/urandom of={{random_file}} bs=100 count=1
  ```

- Benchmark the write performance of a disk:

  ```
  dd if=/dev/zero of={{file_1GB}} bs=1024 count=1000000
  ```

defaults

Read and write macOS user configuration for applications.

More information: https://ss64.com/osx/defaults.html

- Read system defaults for an application option:

```
defaults read {{application}} {{option}}
```

- Read default values for an application option:

```
defaults read -app {{application}} {{option}}
```

- Search for a keyword in domain names, keys, and values:

```
defaults find {{keyword}}
```

- Write the default value of an application option:

```
defaults write {{application}} {{option}} {{-type}} {{value}}
```

- Speed up Mission Control animations:

```
defaults write com.apple.Dock expose-animation-duration -float 0.1
```

- Delete all defaults of an application:

```
defaults delete {{application}}
```

diskutil

Utility to manage local disks and volumes.

- List all currently available disks, partitions and mounted volumes:

  ```
  diskutil list
  ```

- Repair the filesystem data structures of a volume:

  ```
  diskutil repairVolume {{/dev/disk X}}
  ```

- Unmount a volume:

  ```
  diskutil unmountDisk {{/dev/disk X}}
  ```

- Eject a CD/DVD (unmount first):

  ```
  diskutil eject {{/dev/disk 1}}
  ```

ditto

Copy files and directories.

- Overwrite contents of destination directory with contents of source directory:

  ```
  ditto {{path/to/source}} {{path/to/destination}}
  ```

- Print a line to the Terminal window for every file that's being copied:

  ```
  ditto -V {{path/to/source}} {{path/to/destination}}
  ```

- Copy a given file or directory, while retaining the original file permissions:

  ```
  ditto -rsrc {{path/to/source}} {{path/to/destination}}
  ```

dmesg

Write the kernel messages to standard output.

- S how kernel messages:

  ```
  dm esg
  ```

- S how how much physical memory is available on this system:

  ```
  dm esg | g rep - i m em ory
  ```

- S how kernel messages 1 page at a time:

  ```
  dm esg | less
  ```

dot_clean

Merge ._* files with corresponding native files.

More information: https://ss64.com/osx/dot_clean.html

- Merge all ._* files recursively:

```
dot_clean {{path/to/directory}}
```

- Don't recursively merge all ._* in a directory (flat merge):

```
dot_clean -f {{path/to/directory}}
```

- Merge and delete all ._* files:

```
dot_clean -m {{path/to/directory}}
```

- Only delete ._* files if there's a matching native file:

```
dot_clean -n {{path/to/directory}}
```

- Follow symlinks:

```
dot_clean -s {{path/to/directory}}
```

- Print verbose output:

```
dot_clean -v {{path/to/directory}}
```

drutil

Interact with DVD burners.

- Eject a disk from the drive:

  ```
  drutil eject
  ```

- Burn a directory as an ISO 9660 filesystem onto a DVD. Don't verify and eject when complete:

  ```
  drutil burn -noverify -eject -iso9660
  ```

du

Disk usage: estimate and summarize file and directory space usage.

- List the sizes of a directory and any subdirectories, in the given unit (KB/MB/GB):

  ```
  du -{{k|m|g}} {{path/to/directory}}
  ```

- List the sizes of a directory and any subdirectories, in human-readable form (i.e. auto-selecting the appropriate unit for each size):

  ```
  du -h {{path/to/directory}}
  ```

- Show the size of a single directory, in human readable units:

  ```
  du -sh {{path/to/directory}}
  ```

- List the human-readable sizes of a directory and of all the files and directories within it:

  ```
  du -ah {{path/to/directory}}
  ```

- List the human-readable sizes of a directory and any subdirectories, up to N levels deep:

  ```
  du -h -d {{N}} {{path/to/directory}}
  ```

- List the human-readable size of all `.jpg` files in subdirectories of the current directory, and show a cumulative total at the end:

  ```
  du -ch */*.jpg
  ```

duti

Set default applications for document types and URL schemes on macOS.

- Set Safari as the default handler for HTML documents:

  ```
  duti -s {{com.apple.Safari}} {{public.html}} all
  ```

- Set VLC as the default viewer for files with `.m4v` extensions:

  ```
  duti -s {{org.videolan.vlc}} {{m4v}} viewer
  ```

- Set Finder as the default handler for the ftp:// URL scheme:

  ```
  duti -s {{com.apple.Finder}} {{ftp}}
  ```

- Display information about the default application for a given extension:

  ```
  duti -x {{ext}}
  ```

- Display the default handler for a given UTI:

  ```
  duti -d {{uti}}
  ```

- Display all handlers of a given UTI:

  ```
  duti -l {{uti}}
  ```

eval

Execute arguments as a single command in the current shell and return its result.

- Call `echo` with the "foo" argument:

  ```
  eval "{{echo foo}}"
  ```

- Set a variable in the current shell:

  ```
  eval "{{foo=bar}}"
  ```

export

Command to mark shell variables in the current environment to be exported with any newly forked child processes.

- Set a new environment variable:

  ```
  export {{VARIABLE}}={{value}}
  ```

- Remove an environment variable:

  ```
  export -n {{VARIABLE}}
  ```

- Append something to the PATH variable:

  ```
  export PATH=$PATH:{{path/to/append}}
  ```

fc

Open the most recent command and edit it.

- Open in the default system editor:

  ```
  fc
  ```

- Specify an editor to open with:

  ```
  fc -e {{'emacs'}}
  ```

- List recent commands from history:

  ```
  fc -l
  ```

fdesetup

Set and retrieve FileVault related information.

- List current FileVault enabled users:

  ```
  sudo fdesetup list
  ```

- Get current FileVault status:

  ```
  fdesetup status
  ```

- Add FileVault enabled user:

  ```
  sudo fdesetup add -usertoadd user1
  ```

- Enable FileVault:

  ```
  sudo fdesetup enable
  ```

- Disable FileVault:

  ```
  sudo fdesetup disable
  ```

feh

Lightweight image viewing utility.

- View images locally or using a URL:

  ```
  feh {{path/to/images}}
  ```

- View images recursively:

  ```
  feh --recursive {{path/to/images}}
  ```

- View images without window borders:

  ```
  feh --borderless {{path/to/images}}
  ```

- Exit after the last image:

  ```
  feh --cycle-once {{path/to/images}}
  ```

- Set the slideshow cycle delay:

  ```
  feh --slideshow-delay {{seconds}} {{path/to/images}}
  ```

- Set your wallpaper (centered, filled, maximized, scaled or tiled):

  ```
  feh --bg-{{center|fill|max|scale|tile}} {{path/to/image}}
  ```

file

Determine file type.

- G ive a description of the type of the specified file. Works fine for files with no file extension:

  ```
  file {{filenam e}}
  ```

- Look inside a z ipped file and determine the file type(s) inside:

  ```
  file - z {{foo. zip}}
  ```

- Allow file to work with special or device files:

  ```
  file - s {{filenam e}}
  ```

- Don' t stop at first file type match; keep going until the end of the file:

  ```
  file - k  {{filenam e}}
  ```

- Determine the mime encoding type of a file:

  ```
  file - I  {{filenam e}}
  ```

fileicon

A macOS CLI to manage custom file and folder icons.

More information: https://github.com/mklement0/fileicon

- Set a custom icon for a specific file or directory:

  ```
  fileicon set {{path/to/file_or_directory}} {{path/to/icon.png}}
  ```

- Remove a custom icon from a specific file or directory:

  ```
  fileicon rm {{path/to/file_or_directory}}
  ```

- Save the custom icon of a file or directory as a `.icns` file into the current directory:

  ```
  fileicon get {{path/to/file_or_directory}}
  ```

- Test if a specific file or directory has a custom icon:

  ```
  fileicon test {{path/to/file_or_directory}}
  ```

fsck

Check the integrity of a filesystem or repair it. The filesystem should be unmounted at the time the command is run.

It is a wrapper that calls **`fsck.hfs`**, **`fsck.affs`**, **`fsck.msdos`**, **`fsck.exfat`**, and **`fsck.udf`** as needed.

- Check filesystem `/dev/sdX`, reporting any damaged blocks:

  ```
  fsck {{/dev/sdX}}
  ```

- Check filesystem `/dev/sdX` only if it is clean, reporting any damaged blocks and interactively letting the user choose to repair each one:

  ```
  fsck -f {{/dev/sdX}}
  ```

- Check filesystem `/dev/sdX` only if it is clean, reporting any damaged blocks and automatically repairing them:

  ```
  fsck -fy {{/dev/sdX}}
  ```

- Check filesystem `/dev/sdX`, reporting whether it has been cleanly unmounted:

  ```
  fsck -q {{/dev/sdX}}
  ```

G etF ileInfo

G et information about a file in an H F S + directory.

- Display information about a given file:

```
G etF ileI nfo {{path/to/filenam e}}
```

- Display the date and time a given file was created:

```
G etF ileI nfo - d {{path/to/filenam e}}
```

- Display the date and time a given file was last modified:

```
G etF ileI nfo - m  {{path/to/filenam e}}
```

- Display the creator of a given file:

```
G etF ileI nfo - c {{path/to/filenam e}}
```

hdiutil

Utility to create and manage disk images.

- Mount an image:

  ```
  hdiutil attach {{path/to/image_file}}
  ```

- Unmount an image:

  ```
  hdiutil detach /Volumes/{{volume_name}}
  ```

- List mounted images:

  ```
  hdiutil info
  ```

- Create an ISO image from the contents of a directory:

  ```
  hdiutil makehybrid -o {{path/to/output_file}} {{path/to/directory}}
  ```

head

Output the first part of files.

- Output the first few lines of a file:

  ```
  head -n {{count_of_lines}} {{filename}}
  ```

- Output the first few bytes of a file:

  ```
  head -c {{number_in_bytes}} {{filename}}
  ```

hexdump

An ASCII, decimal, hexadecimal, octal dump.

- Print the hexadecimal representation of a file:

  ```
  hexdump {{file}}
  ```

- Display the input offset in hexadecimal and its ASCII representation in two columns:

  ```
  hexdump -C {{file}}
  ```

- Display the hexadecimal representation of a file, but interpret only n bytes of the input:

  ```
  hexdump -C -n{{number_of_bytes}} {{file}}
  ```

hostname

Show or set the system's host name.

- Show current host name:

  ```
  hostnam e
  ```

- Set current host name:

  ```
  hostnam e {{new _hostnam e}}
  ```

icalB uddy

Command-line utility for printing events and tasks from the macO S calendar database.

More information: https://hasseg.org/icalB uddy/

- S how events later today:

  ```
  icalB u ddy - n ev entsT oday
  ```

- S how uncompleted tasks:

  ```
  icalB u ddy u ncom pletedT ask s
  ```

- S how a formatted list separated by calendar for all events today:

  ```
  icalB u ddy - f - sc ev entsT oday
  ```

- S how tasks for a specified number of days:

  ```
  icalB u ddy - n task sD u eB efore: today+ {{days}}
  ```

- S how events in a time range:

  ```
  icalB u ddy ev entsF rom : ' {{start_date}}' to: ' {{end_date}}'
  ```

imgcat

A utility to display images directly on the command-line.

R eq uires a compatible terminal such as iT erm2 .

- Display an image on the command-line:

  ```
  im g cat {{filenam e}}
  ```

indent

Change the appearance of a C/C++ program by inserting or deleting whitespace.

More information: https://www.freebsd.org/cgi/man.cgi?query=indent

- Format C/C++ source according to the Berkeley style:

```
indent {{path/to/source.c}} {{path/to/indented_source.c}} -nbad -nbap -bc -br -c33 -cd33 -cdb -ce -ci4 -cli0 -di16 -fcb -i4 -ip -l75 -lp -npcs -nprs -psl -sc -nsob -ts8
```

- Format C/C++ source according to the style of Kernighan & Ritchie (K&R):

```
indent {{path/to/source.c}} {{path/to/indented_source.c}} -nbad -bap -nbc -br -c33 -cd33 -ncdb -ce -ci4 -cli0 -cs -di1 -nfc1 -nfcb -i4 -nip -l75 -lp -npcs -nprs -npsl -nsc -nsob
```

istats

CLI tool that shows statistics such as CPU temperature, fan speeds and battery status.

More information: https://github.com/Chris911/iStats

- Show all the stats:

  ```
  istats
  ```

- Show all CPU stats:

  ```
  istats cpu
  ```

- Show all fan stats:

  ```
  istats fan
  ```

- Scan and print temperatures:

  ```
  istats scan
  ```

launchctl

A command-line interface to Apple's `launchd` manager for launch daemons (system-wide services) and launch agents (per-user programs).

`launchd` loads XML-based `*.plist` files placed in the appropriate locations, and runs the corresponding commands according to their defined schedule.

- Activate a user-specific agent to be loaded into `launchd` whenever the user logs in:

  ```
  launchctl load ~/Library/LaunchAgents/{{my_script}}.plist
  ```

- Activate an agent which requires root privileges to run and/or should be loaded whenever any user logs in (note the absence of `~` in the path):

  ```
  sudo launchctl load /Library/LaunchAgents/
  {{root_script}}.plist
  ```

- Activate a system-wide daemon to be loaded whenever the system boots up (even if no user logs in):

  ```
  sudo launchctl load /Library/LaunchDaemons/
  {{system_daemon}}.plist
  ```

- Show all loaded agents/daemons, with the PID if the process they specify is currently running, and the exit code returned the last time they ran:

  ```
  launchctl list
  ```

- Unload a currently loaded agent, e.g. to make changes (note: the plist file is automatically loaded into `launchd` after a reboot and/or logging in):

  ```
  launchctl unload ~/Library/LaunchAgents/{{my_script}}.plist
  ```

- Manually run a known (loaded) agent/daemon, even if it is not the right time (note: this command uses the agent's label, rather than the filename):

  ```
  launchctl start {{my_script}}
  ```

- Manually kill the process associated with a known agent/daemon, if it is running:

  ```
  launchctl stop {{my_script}}
  ```

lldb

The LLVM Low-Level Debugger.

- Debug an executable:

  ```
  lldb {{executable}}
  ```

- Attach `lldb` to a running process with a given PID:

  ```
  lldb -p {{pid}}
  ```

- Wait for a new process to launch with a given name, and attach to it:

  ```
  lldb -w -n {{process_name}}
  ```

locate

Find filenames quickly.

- Look for pattern in the database. Note: the database is recomputed periodically (usually weekly or daily):

  ```
  locate {{pattern}}
  ```

- Look for a file by its exact filename (a pattern containing no globbing characters is interpreted as `*pattern*`):

  ```
  locate * /{{filename}}
  ```

- Recompute the database. You need to do it if you want to find recently added files:

  ```
  sudo /usr/libexec/locate.updatedb
  ```

logger

Add messages to syslog (/var/log/syslog).

- Log a message to syslog:

```
logger {{message}}
```

- Take input from stdin and log to syslog:

```
echo {{log_entry}} | logger
```

- Send the output to a remote syslog server running at a given port. Default port is 514:

```
echo {{log_entry}} | logger -h {{hostname}} -P {{port}}
```

- Use a specific tag for every line logged. Default is the name of logged in user:

```
echo {{log_entry}} | logger -t {{tag}}
```

- Log messages with a given priority. Default is `user.notice`. See `man logger` for all priority options:

```
echo {{log_entry}} | logger -p {{user.warning}}
```

look

Look for lines in sorted file.

- Look for lines which begins with the given prefix:

  ```
  look {{prefix}} {{file}}
  ```

- Look for lines ignoring case:

  ```
  look -f {{prefix}} {{file}}
  ```

m

Swiss Army Knife for macOS.

- Get the battery status:

  ```
  m battery status
  ```

- Turn off bluetooth:

  ```
  m bluetooth off
  ```

- List available filesystems for formatting:

  ```
  m disk filesystems
  ```

- Enable Dock's auto hide feature:

  ```
  m dock autohide YES
  ```

- Disable the firewall:

  ```
  m firewall disable
  ```

mas

Command-line interface for the Mac App Store.

More information: https://github.com/mas-cli/mas.

- Sign into the Mac App Store for the first time:

  ```
  mas signin {{user@example.com}}
  ```

- Show all installed applications and their product identifiers:

  ```
  mas list
  ```

- Search for an application, displaying the price alongside the results:

  ```
  mas search {{application}} --price
  ```

- Install or update an application:

  ```
  mas install {{product_identifier}}
  ```

- Install all pending updates:

  ```
  mas upgrade
  ```

md5

Calculate MD5 cryptographic checksums.

- Calculate the MD5 checksum for a file:

  ```
  m d5 {{filenam e}}
  ```

- Calculate MD5 checksums for multiple files:

  ```
  m d5 {{filenam e1 }} {{filenam e2 }}
  ```

- O utput only the md5 checksum (no filename):

  ```
  m d5 - q {{filenam e}}
  ```

- P rint a checksum of the given string:

  ```
  m d5 - s {{string }}
  ```

mdfind

List files matching a given query.

- Find a file by its name:

  ```
  mdfind -name {{file}}
  ```

- Find a file by its content:

  ```
  mdfind {{query}}
  ```

- Find a file containing a string, in a given directory:

  ```
  mdfind -onlyin {{directory}} {{query}}
  ```

mdls

Display the metadata attributes for a file.

- Display the list of metadata attributes for file:

  ```
  m dls {{path/to/file}}
  ```

- Display a specific metadata attribute:

  ```
  m dls - nam e {{attrib u te}} {{path/to/file}}
  ```

mdutil

Manage the metadata stores used by Spotlight for indexing.

- Show the indexing status of the startup volume:

  ```
  mdutil -s {{/}}
  ```

- Turn on/off the Spotlight indexing for a given volume:

  ```
  mdutil -i {{on|off}} {{path/to/volume}}
  ```

- Turn on/off indexing for all volumes:

  ```
  mdutil -a -i {{on|off}}
  ```

- Erase the metadata stores and restart the indexing process:

  ```
  mdutil -E {{path/to/volume}}
  ```

mkfile

Create one or more empty files of any size.

- Create an empty file of 15 kilobytes:

  ```
  mkfile -n {{15k}} {{filename}}
  ```

- Create a file of a given size and unit (bytes, KB, MB, GB):

  ```
  mkfile -n {{size}}{{b|k|m|g}} {{filename}}
  ```

- Create two files of 4 megabytes each:

  ```
  mkfile -n {{4m}} {{first_filename}} {{second_filename}}
  ```

n

Tool to manage multiple node versions.

- Install a given version of node. If the version is already installed, it will be activated:

  ```
  n {{version}}
  ```

- Display installed versions and interactively activate one of them:

  ```
  n
  ```

- Remove a version:

  ```
  n rm {{version}}
  ```

- Execute a file with a given version:

  ```
  n use {{version}} {{file.js}}
  ```

- Output binary path for a version:

  ```
  n bin {{version}}
  ```

netstat

Displays network-related information such as open connections, open socket ports, etc.

More information: https://www.unix.com/man-page/osx/1 /netstat

- List all ports:

  ```
  netstat - a
  ```

- List all listening ports:

  ```
  netstat - l
  ```

- List listening T CP ports:

  ```
  netstat - t
  ```

- Display P ID and program names for a specific protocol:

  ```
  netstat - p {{protocol}}
  ```

- P rint the routing table:

  ```
  netstat - nr
  ```

nettop

Display updated information about the network.

- Monitor TCP and UDP sockets from all interfaces:

  ```
  nettop
  ```

- Monitor TCP sockets from Loopback interfaces:

  ```
  nettop -m {{tcp}} -t {{loopback}}
  ```

- Monitor a specific process:

  ```
  nettop -p {{process_id|process_name}}
  ```

- Display a per-process summary:

  ```
  nettop -P
  ```

- Print 10 samples of network information:

  ```
  nettop -l {{10}}
  ```

- Monitor changes every 5 seconds:

  ```
  nettop -d -s {{5}}
  ```

- While running nettop, list interactive commands:

  ```
  h
  ```

- Display help:

  ```
  nettop -h
  ```

networksetup

Configuration tool for Network System Preferences.

- List available network service providers (Ethernet, Wi-Fi, Bluetooth, etc):

  ```
  networksetup -listallnetworkservices
  ```

- Show network settings for a particular networking device:

  ```
  networksetup -getinfo "{{Wi-Fi}}"
  ```

- Get currently connected Wi-Fi network name (Wi-Fi device usually en0 or en1):

  ```
  networksetup -getairportnetwork {{en0}}
  ```

- Connect to a particular Wi-Fi network:

  ```
  networksetup -setairportnetwork {{en0}} "{{Airport Network SSID}}" {{password}}
  ```

nm

List symbol names in object files.

- List global (extern) functions in a file (prefixed with T):

```
nm -g {{file.o}}
```

- List only undefined symbols in a file:

```
nm -u {{file.o}}
```

- List all symbols, even debugging symbols:

```
nm -a {{file.o}}
```

- Demangle C++ symbols (make them readable):

```
nm -demangle {{file.o}}
```

oathtool

OATH one-time password tool.

- Generate TOTP token (behaves like Google Authenticator):

  ```
  oathtool --totp --base32 {{secret}}
  ```

- Generate a TOTP token for a specific time:

  ```
  oathtool --totp --now {{2004-02-29 16:21:42}} --base3
  {{secret}}
  ```

- Validate a TOTP token:

  ```
  oathtool --totp --base32 {{secret}} {{token}}
  ```

open

Opens files, directories and applications.

- Open a file with the associated application:

  ```
  open {{file.ext}}
  ```

- Run a graphical macOS application:

  ```
  open -a {{Application}}
  ```

- Run a graphical macOS app based on the bundle identifier (refer to `osascript` for an easy way to get this):

  ```
  open -b {{com.domain.application}}
  ```

- Open the current directory in Finder:

  ```
  open .
  ```

- Reveal a file in Finder:

  ```
  open -R {{path/to/file}}
  ```

- Open all the files of a given extension in the current directory with the associated application:

  ```
  open {{*.ext}}
  ```

opensnoop

Tool that tracks file opens on your system.

- Print all file opens as they occur:

  ```
  sudo opensnoop
  ```

- Track all file opens by a process by name:

  ```
  sudo opensnoop -n {{process_name}}
  ```

- Track all file opens by a process by PID:

  ```
  sudo opensnoop -p {{PID}}
  ```

- Track which processes open a specified file:

  ```
  sudo opensnoop -f {{path/to/file}}
  ```

osascript

Run AppleScript or JavaScript for Automation (JXA) from the command-line.

- Run an AppleScript command:

  ```
  osascript -e '{{say "Hello world"}}'
  ```

- Run multiple AppleScript commands:

  ```
  osascript -e '{{say "Hello"}}' -e '{{say "world"}}'
  ```

- Run a compiled (`*.scpt`), bundled (`*.scptd`), or plaintext (`*.applescript`) AppleScript file:

  ```
  osascript {{path/to/apple.scpt}}
  ```

- Get the bundle identifier of an application (useful for `open -b`):

  ```
  osascript -e 'id of app "{{Application}}"'
  ```

- Run a JavaScript command:

  ```
  osascript -l JavaScript -e '{{console.log("Hello world");}}'
  ```

- Run a JavaScript file:

  ```
  osascript -l JavaScript {{path/to/script.js}}
  ```

pbcopy

Place standard output in the clipboard.

- Place the contents of a file in the clipboard:

  ```
  pb copy < {{file}}
  ```

- Place the results of a command in the clipboard:

  ```
  find . -type t -name "*.png" | pb copy
  ```

pbpaste

Send the contents of the clipboard to standard output.

- Write the contents of the clipboard to a file:

  ```
  pbpaste > {{file}}
  ```

- Use the contents of the clipboard as input to a command:

  ```
  pbpaste | grep foo
  ```

pdfgrep

Search text in PDF files.

- Find lines that match pattern in a PDF:

  ```
  pdfgrep {{pattern}} {{file.pdf}}
  ```

- Include file name and page number for each matched line:

  ```
  pdfgrep --with-filename --page-number {{pattern}} {{file.pdf}}
  ```

- Do a case insensitive search for lines that begin with "foo" and return the first 3 matches:

  ```
  pdfgrep --max-count {{3}} --ignore-case {{'^foo'}} {{file.pdf}}
  ```

- Find pattern in files with a `.pdf` extension in the current directory recursively:

  ```
  pdfgrep --recursive {{pattern}}
  ```

- Find pattern on files that match a specific glob in the current directory recursively:

  ```
  pdfgrep --recursive --include {{'*book.pdf'}} {{pattern}}
  ```

ping

Send ICMP ECHO_REQUEST packets to network hosts.

- Ping the specified host:

  ```
  ping {{host}}
  ```

- Ping a host a specific number of times:

  ```
  ping -c {{count}} {{host}}
  ```

- Ping `host`, specifying the interval in `seconds` between requests (default is 1 second):

  ```
  ping -i {{seconds}} {{host}}
  ```

- Ping `host` without trying to lookup symbolic names for addresses:

  ```
  ping -n {{host}}
  ```

- Ping `host` and ring the bell when a packet is received (if your terminal supports it):

  ```
  ping -a {{host}}
  ```

- Ping `host` and prints the time a packet was received (this option is an Apple addition):

  ```
  ping --apple-time {{host}}
  ```

pkgutil

Query and manipulate Mac OS X Installer packages and receipts.

- List package IDs for all installed packages:

  ```
  pkgutil --pkgs
  ```

- Verify cryptographic signatures of a package file:

  ```
  pkgutil --check-signature {{path/to/filename.pkg}}
  ```

- List all the files for an installed package given its ID:

  ```
  pkgutil --files {{com.microsoft.Word}}
  ```

- Extract the contents of a package file into a directory:

  ```
  pkgutil --expand-full {{path/to/filename.pkg}} {{path/to/directory}}
  ```

plutil

View, convert, validate, or edit property list ("plist") files.

- Display the contents of one or more plist files in human-readable format:

  ```
  plutil -p {{file1.plist file2.plist ...}}
  ```

- Convert one or more plist files to XML format, overwriting the original files in-place:

  ```
  plutil -convert xml1 {{file1.plist file2.plist ...}}
  ```

- Convert one or more plist files to binary format, overwriting the original files in-place:

  ```
  plutil -convert binary1 {{file1.plist file2.plist ...}}
  ```

- Convert a plist file to a different format, writing to a new file:

  ```
  plutil -convert {{xml1|binary1|json|swift|objc}} {{path/to/file.plist}} -o {{path/to/new_file.plist}}
  ```

- Convert a plist file to a different format, writing to stdout:

  ```
  plutil -convert {{xml1|binary1|json|swift|objc}} {{path/to/file.plist}} -o -
  ```

pmset

Configure macOS power management settings, as one might do in System Preferences > Energy Saver.

Commands that modify settings must begin with **`sudo`**

- Display the current power management settings:

  ```
  pmset -g
  ```

- Display the current power source and battery levels:

  ```
  pmset -g batt
  ```

- Put display to sleep immediately:

  ```
  pmset displaysleepnow
  ```

- Set display to never sleep when on charger power:

  ```
  sudo pmset -c displaysleep 0
  ```

- Set display to sleep after 15 minutes when on battery power:

  ```
  sudo pmset -b displaysleep 15
  ```

- Schedule computer to automatically wake up every weekday at 9 AM:

  ```
  sudo pmset repeat wake MTWRF 09:00:00
  ```

- Restore to system defaults:

  ```
  sudo pmset -a displaysleep 10 disksleep 10 sleep 30 womp 1
  ```

pod

Dependency manager for Swift and Objective-C Cocoa projects.

- Create a Podfile for the current project with the default contents:

  ```
  pod init
  ```

- Download and install all pods defined in the Podfile (that haven't been installed before):

  ```
  pod install
  ```

- List all available pods:

  ```
  pod list
  ```

- Show the outdated pods (of those currently installed):

  ```
  pod outdated
  ```

- Update all currently installed pods to their newest version:

  ```
  pod update
  ```

- Update a specific (previously installed) pod to its newest version:

  ```
  pod update {{pod_name}}
  ```

- Remove CocoaPods from a Xcode project:

  ```
  pod deintegrate {{xcode_project}}
  ```

port

Package manager for macOS.

- Search for a package:

  ```
  port search {{search_term}}
  ```

- Install a package:

  ```
  sudo port install {{package_name}}
  ```

- List installed packages:

  ```
  port installed
  ```

- Update port and fetch latest list of available packages:

  ```
  sudo port selfupdate
  ```

- Upgrade outdated packages:

  ```
  sudo port upgrade outdated
  ```

- Remove old versions of installed packages:

  ```
  sudo port uninstall inactive
  ```

pwgen

G enerate pronounceable passwords.

- G enerate random password with s[y] mbols:

  ```
  pw g en - y {{leng th}}
  ```

- G enerate secure, hard-to-memoriz e passwords:

  ```
  pw g en - s {{leng th}}
  ```

- G enerate password with at least one capital letter in them:

  ```
  pw g en - c {{leng th}}
  ```

q lmanage

Q uickLook server tool.

- Display Q uickLook for one or multiple files:

  ```
  q lm anag e - p {{filenam e}} {{filenam e2 }}
  ```

- Compute 3 0 0 px wide P N G thumbnails of all J P E G s in the current directory and put them in a directory:

  ```
  q lm anag e {{* . j pg }} - t - s {{3 0 0 }} {{path/to/directory}}
  ```

- R eset Q uickLook:

  ```
  q lm anag e - r
  ```

readlink

Follow symlinks and get symlink information.

More information: https://www.gnu.org/software/coreutils/readlink.

- Print the absolute path which the symlink points to:

```
readlink {{path/to/symlink}}
```

reboot

Reboot the system.

- Reboot immediately:

  ```
  sudo reboot
  ```

- Reboot immediately without gracefully shutting down:

  ```
  sudo reboot -q
  ```

rename

R ename a file or group of files with a regular expression.

- R eplace`from` with `to` in the filenames of the specified files:

  ```
  renam e ' s/{{from }}/{{to}}/'  {{* . txt}}
  ```

route

Manually manipulate the routing tables.

Necessitates to be root.

- Add a route to a destination through a gateway:

  ```
  sudo route add {{dest_ip_address}} {{gateway_address}}
  ```

- Add a route to a /24 subnet through a gateway:

  ```
  sudo route add {{subnet_ip_address}}/24 {{gateway_address}}
  ```

- Run in test mode (does not do anything, just print):

  ```
  sudo route -t add {{dest_ip_address}}/24 {{gateway_address}}
  ```

- Remove all routes:

  ```
  sudo route flush
  ```

- Delete a specific route:

  ```
  sudo route delete {{dest_ip_address}}/24
  ```

- Lookup and display the route for a destination (hostname or IP address):

  ```
  sudo route get {{destination}}
  ```

rubocop

Lint Ruby files.

- Check all files in the current directory (including subdirectories):

  ```
  rubocop
  ```

- Check one or more specific files or directories:

  ```
  rubocop {{path/to/file}} {{path/to/directory}}
  ```

- Write output to file:

  ```
  rubocop --out {{path/to/file}}
  ```

- View list of cops (linter rules):

  ```
  rubocop --show-cops
  ```

- Exclude a cop:

  ```
  rubocop --except {{cop_1}} {{cop_2}}
  ```

- Run only specified cops:

  ```
  rubocop --only {{cop_1}} {{cop_2}}
  ```

- Auto-correct files (experimental):

  ```
  rubocop --auto-correct
  ```

S afeE j ectG P U

E j ect a G P U safely.

V isit the man page for more info.

- E j ect all G P U s:

  ```
  S afeE j ectG P U E j ect
  ```

- List all G P U s attached:

  ```
  S afeE j ectG P U g pu s
  ```

- List apps using a G P U :

  ```
  S afeE j ectG P U g pu id {{G P U _I D }} apps
  ```

- G et the status of a G P U :

  ```
  S afeE j ectG P U g pu id {{G P U _I D }} statu s
  ```

- E j ect a G P U :

  ```
  S afeE j ectG P U g pu id {{G P U _I D }} E j ect
  ```

- Launch an app on a G P U :

  ```
  S afeE j ectG P U g pu id {{G P U _I D }} L au nchO nG P U {{path/to/A pp.
  ```

say

Converts text to speech.

- Say a phrase aloud:

```
say "{{I like to ride my bike.}}"
```

- Read a file aloud:

```
say -f {{filename.txt}}
```

- Say a phrase with a custom voice and speech rate:

```
say -v {{voice}} -r {{words_per_minute}} "{{I'm sorry Dave, I
can't let you do that.}}"
```

- List the available voices:

```
say -v "?"
```

- Create an audio file of the spoken text:

```
say -o {{filename.aiff}} "{{Here's to the Crazy Ones.}}"
```

screencapture

Utility to take screenshots and screen recordings.

- Take a screenshot and save it to a file:

  ```
  screencapture {{path/to/file.png}}
  ```

- Take a screenshot including the mouse cursor:

  ```
  screencapture -C {{path/to/file.png}}
  ```

- Take a screenshot and open it in Preview, instead of saving:

  ```
  screencapture -P
  ```

- Take a screenshot of a selected rectangular area:

  ```
  screencapture -i {{path/to/file.png}}
  ```

- Take a screenshot after a delay:

  ```
  screencapture -T {{seconds}} {{path/to/file.png}}
  ```

- Make a screen recording and save it to a file:

  ```
  screencapture -v {{path/to/file.mp4}}
  ```

scutil

Manage system configuration parameters.

Necessitates to be root when setting configuration.

- Display DNS Configuration:

  ```
  scutil --dns
  ```

- Display proxy configuration:

  ```
  scutil --proxy
  ```

- Get computer name:

  ```
  scutil --getComputerName
  ```

- Set computer name:

  ```
  sudo scutil --setComputerName {{computer_name}}
  ```

- Get hostname:

  ```
  scutil --getHostName
  ```

- Set hostname:

  ```
  scutil --setHostName {{hostname}}
  ```

security

Administer Keychains, keys, certificates and the S ecurity framework.

More information: https://ss6 4 .com/osx/security.html

- List the available keychains:

  `secu rity list- k eychains`

- Delete a specific keychain:

  `secu rity delete- k eychain {{path}}`

- Create a keychain:

  `secu rity create- k eychain - p {{passw ord}} {{nam e. k eychain}}`

sed

Edit text in a scriptable manner.

More information: https://ss64.com/osx/sed.html

- Replace the first occurrence of a string in a file, and print the result:

```
sed 's/{{find}}/{{replace}}/' {{filename}}
```

- Replace all occurrences of an extended regular expression in a file:

```
sed -E 's/{{regular_expression}}/{{replace}}/g' {{filename}}
```

- Replace all occurrences of a string [in] a file, overwriting the file (i.e. in-place):

```
sed -i '' 's/{{find}}/{{replace}}/g' {{filename}}
```

- Replace only on lines matching the line pattern:

```
sed '/{{line_pattern}}/s/{{find}}/{{replace}}/' {{filename}}
```

- Print only text between n-th line till the next empty line:

```
sed -n '{{line_number}},/^$/p' {{filename}}
```

- Apply multiple find-replace expressions to a file:

```
sed -e 's/{{find}}/{{replace}}/' -e 's/{{find}}/{{replace}}/' {{filename}}
```

- Replace separator / by any other character not used in the find or replace patterns, e.g., # :

```
sed 's#{{find}}#{{replace}}#' {{filename}}
```

- [d]elete the line at the specific line number [in] a file, overwriting the file:

```
sed -i '' '{{line_number}}d' {{filename}}
```

shuf

Generate random permutations.

- Randomize the order of lines in a file and output the result:

  ```
  shuf {{filename}}
  ```

- Only output the first 5 entries of the result:

  ```
  shuf -n {{5}} {{filename}}
  ```

- Write output to another file:

  ```
  shuf {{filename}} -o {{output_filename}}
  ```

- Generate random numbers in range 1-10:

  ```
  shuf -i {{1-10}}
  ```

shutdown

Shutdown and reboot the system.

- Power off (halt) immediately:

  ```
  shutdown -h now
  ```

- Sleep immediately:

  ```
  shutdown -s now
  ```

- Reboot immediately:

  ```
  shutdown -r now
  ```

- Reboot in 5 minutes:

  ```
  shutdown -r +{{5}}
  ```

- Power off (halt) at 1:00 pm (Uses 24h clock):

  ```
  shutdown -h {{1300}}
  ```

- Reboot on May 10th 2042 at 11:30 am (Input format: YYMMDDHHMM):

  ```
  shutdown -r {{4205101130}}
  ```

sips

Apple Scriptable Image Processing System.

Raster/Query images and ColorSync ICC Profiles.

- Specify an output directory so that originals do not get modified:

  ```
  sips --out {{path/to/out_dir}}
  ```

- Resample image at specified size, Image aspect ratio may be altered:

  ```
  sips -z {{1920}} {{300}} {{image.ext}}
  ```

- Resample image so height and width aren't greater than specified size (notice the capital Z):

  ```
  sips -Z {{1920}} {{300}} {{image.ext}}
  ```

- Resample all images in a directory to fit a width of 960 px (honoring aspect ratio):

  ```
  sips --resampleWidth {{960}} {{path/to/images}}
  ```

- Convert an image from CMYK to RGB:

  ```
  sips --matchTo '/System/Library/ColorSync/Profiles/Generic
  RGB Profile.icc' {{path/to/image.ext}} {{path/to/out_dir}}
  ```

- Remove ColorSync ICC profile from an image:

  ```
  sips -d profile --deleteColorManagementProperties {{path/to/
  image.ext}}
  ```

softwareupdate

A tool for updating macOS App Store apps via the command-line.

- List all available updates:

  ```
  softwareupdate -l
  ```

- Download and install all updates:

  ```
  softwareupdate -ia
  ```

- Download and install all recommended updates:

  ```
  softwareupdate -ir
  ```

- Download and install a specific app:

  ```
  softwareupdate -i {{update_name}}
  ```

split

Split a file into pieces.

- Split a file, each split having 10 lines (except the last split):

  ```
  split -l {{10}} {{filename}}
  ```

- Split a file by a regular expression. The matching line will be the first line of the next output file:

  ```
  split -p {{cat|^[dh]og}} {{filename}}
  ```

- Split a file with 512 bytes in each split (except the last split; use 512k for kilobytes and 512m for megabytes):

  ```
  split -b {{512}} {{filename}}
  ```

spotify

A command-line interface to S potify.

More information: https://github.com/hnarayanan/shpotify.

- F ind a song by name and play it:

  ```
  spotify play {{song _nam e}}
  ```

- F ind a playlist by name and play it:

  ```
  spotify play list {{playlist_nam e}}
  ```

- P ause (or resume) playback:

  ```
  spotify pau se
  ```

- S kip to the next song in a playlist:

  ```
  spotify next
  ```

- Change volume:

  ```
  spotify v ol {{u p| dow n| v alu e}}
  ```

- S how the playback status and song details:

  ```
  spotify statu s
  ```

ssh-add

Manage loaded ssh keys in the ssh-agent.

E nsure that ssh-agent is up and running for the keys to be loaded in it.

- Add the default ssh keys in `~/.ssh` to the ssh-agent:

  ```
  ssh-add
  ```

- Add a specific key to the ssh-agent:

  ```
  ssh-add {{path/to/private_key}}
  ```

- List fingerprints of currently loaded keys:

  ```
  ssh-add -l
  ```

- Delete a key from the ssh-agent:

  ```
  ssh-add -d {{path/to/private_key}}
  ```

- Delete all currently loaded keys from the ssh-agent:

  ```
  ssh-add -D
  ```

- Add a key to the ssh-agent and the keychain:

  ```
  ssh-add -K {{path/to/private_key}}
  ```

sshuttle

Transparent proxy server that tunnels traffic over an SSH connection.

Doesn't require admin, or any special setup on the remote SSH server.

More information: https://github.com/sshuttle/sshuttle.

- Forward all IPv4 TCP traffic via a remote SSH server:

  ```
  sshuttle --remote={{username}}@{{sshserver}} {{0.0.0.0/0}}
  ```

- Forward all IPv4 TCP and DNS traffic:

  ```
  sshuttle --dns --remote={{username}}@{{sshserver}}
  {{0.0.0.0/0}}
  ```

- Use the tproxy method to forward all IPv4 and IPv6 traffic:

  ```
  sudo sshuttle --method=tproxy --remote={{username}}
  @{{sshserver}} {{0.0.0.0/0}} {{::/0}} --
  exclude={{your_local_ip_address}} --
  exclude={{ssh_server_ip_address}}
  ```

stat

Display file status.

- Show file properties such as size, permissions, creation and access dates among others:

  ```
  stat {{file}}
  ```

- Same as above but verbose (more similar to linux' `stat`):

  ```
  stat -x {{file}}
  ```

- Show only octal file permissions:

  ```
  stat -f %Mp%Lp {{file}}
  ```

- Show owner and group of the file:

  ```
  stat -f "%Su %Sg" {{file}}
  ```

- Show the size of the file in bytes:

  ```
  stat -f "%z %N" {{file}}
  ```

sw_vers

Print macOS operating system version information.

- Print all available information (OS name, version number, and build):

  ```
  sw_vers
  ```

- Print only the version number of the operating system:

  ```
  sw_vers -productVersion
  ```

- Print only the build identifier:

  ```
  sw_vers -buildVersion
  ```

sysctl

Access kernel state information.

- Show all available variables and their values:

  ```
  sysctl -a
  ```

- Show Apple model identifier:

  ```
  sysctl -n hw.model
  ```

- Show CPU model:

  ```
  sysctl -n machdep.cpu.brand_string
  ```

- Show available CPU features (MMX, SSE, SSE2, SSE3, AES, etc):

  ```
  sysctl -n machdep.cpu.features
  ```

- Set a changeable kernel state variable:

  ```
  sysctl -w {{section.tunable}}={{value}}
  ```

system_profiler

Report system hardware and software configuration.

- Display a full system profiler report which can be opened by System Profiler.app:

  ```
  system_profiler -xml > MyReport.spx
  ```

- Display a hardware overview (Model, CPU, Memory, Serial, etc):

  ```
  system_profiler SPHardwareDataType
  ```

- Print the system serial number:

  ```
  system_profiler SPHardwareDataType | grep "Serial Number (system)" | awk '{print $4}'
  ```

systemsetup

Configure System Preferences machine settings.

- Enable remote login (SSH):

  ```
  systemsetup -setremoteloginon
  ```

- Specify TimeZone, NTP Server and enable network time:

  ```
  systemsetup -settimezone {{US/Pacific}} -setnetworktimeserver
  {{us.pool.ntp.org}} -setusingnetworktime on
  ```

- Make the machine never sleep and automatically restart on power failure or kernel panic:

  ```
  systemsetup -setsleep off -setrestartpowerfailure on -
  setrestartfreeze on
  ```

- List valid startup disks:

  ```
  systemsetup -liststartupdisks
  ```

- Specify a new startup disk:

  ```
  systemsetup -setstartupdisk {{path}}
  ```

textutil

Used to manipulate text files of various formats.

- Display information about `foo.rtf`:

  ```
  textutil -info {{foo.rtf}}
  ```

- Convert `foo.rtf` into `foo.html`:

  ```
  textutil -convert {{html}} {{foo.rtf}}
  ```

- Convert rich text to normal text:

  ```
  textutil {{foo.rtf}} -convert {{txt}}
  ```

- Convert `foo.txt` into `foo.rtf`, using Times 10 for the font:

  ```
  textutil -convert {{rtf}} -font {{Times}} -fontsize {{10}}
  {{foo.txt}}
  ```

- Load all RTF files in the current directory, concatenates their contents, and writes the result out as `index.html` with the HTML title set to "Several Files":

  ```
  textutil -cat {{html}} -title "Several Files" -output
  {{index.html}} *.rtf
  ```

tmutil

Utility for managing Time Machine backups. Most verbs require root privileges.

More information: https://ss64.com/osx/tmutil.html

- Set a HFS+ drive as the backup destination:

```
sudo tmutil setdestination {{path/to/disk_mount_point}}
```

- Set a AFP share or SMB share as the backup destination:

```
sudo tmutil setdestination {{protocol://user[:password]@host/share}}
```

- Append the given destination to the list of destinations:

```
sudo tmutil setdestination -a {{destination}}
```

- Enable automatic backups:

```
sudo tmutil enable
```

- Disable automatic backups:

```
sudo tmutil disable
```

- Start a backup, if one is not running already, and release control of the shell:

```
sudo tmutil startbackup
```

- Start a backup and block until the backup is finished:

```
sudo tmutil startbackup -b
```

- Stop a backup:

```
sudo tmutil stopbackup
```

top

Display dynamic real-time information about running processes.

- Start top, all options are available in the interface:

  ```
  top
  ```

- Start top sorting processes by internal memory size (default order - process ID):

  ```
  top -o mem
  ```

- Start top sorting processes first by CPU, then by running time:

  ```
  top -o cpu -O time
  ```

- Start top displaying only processes owned by given user:

  ```
  top -user {{user_name}}
  ```

- Get help about interactive commands:

  ```
  ?
  ```

trap

Automatically execute commands after receiving signals by processes or the operating system.

Can be used to perform cleanups for interruptions by the user or other actions.

- List available signals to set traps for:

  ```
  trap -l
  ```

- List active traps for the current shell:

  ```
  trap -p
  ```

- Set a trap to execute commands when one or more signals are detected:

  ```
  trap 'echo "Caught signal {{SIGHUP}}"' {{SIGHUP}}
  ```

- Remove active traps:

  ```
  trap - {{SIGHUP}} {{SIGINT}}
  ```

tree

Show the contents of the current directory as a tree.

More information: http://mama.indstate.edu/users/ice/tree/.

- Print files and directories up to 'num' levels of depth (where 1 means the current directory):

  ```
  tree -L {{num}}
  ```

- Print directories only:

  ```
  tree -d
  ```

- Print hidden files too with colorization on:

  ```
  tree -a -C
  ```

- Print the tree without indentation lines, showing the full path instead (use `-N` to not escape whitespace and special characters):

  ```
  tree -i -f
  ```

- Print the size of each node next to it, in human-readable format, with directories displaying their cumulative size (as in the `du` command):

  ```
  tree -s -h --du
  ```

- Print files within the tree hierarchy, using a wildcard (glob) pattern, and pruning out directories that don't contain matching files:

  ```
  tree -P '{{*.txt}}' --prune
  ```

- Print directories within the tree hierarchy, using the wildcard (glob) pattern, and pruning out directories that aren't ancestors of the wanted one:

  ```
  tree -P {{directory_name}} --matchdirs --prune
  ```

- Print the tree ignoring the given directories:

  ```
  tree -I '{{directory_name1|directory_name2}}'
  ```

uname

Print details about the current machine and the operating system running on it.

Note: for additional information about the operating system, try the `sw_vers` command.

More information: https://ss64.com/osx/uname.html

- Print kernel name:

  `uname`

- Print system architecture and processor information:

  `uname -mp`

- Print kernel name, kernel release and kernel version:

  `uname -srv`

- Print system hostname:

  `uname -n`

- Print all available system information:

  `uname -a`

uptime

Tell how long the system has been running and other information.

- Print current time, uptime, number of logged-in users and other information:

  ```
  uptime
  ```

uuidgen

Generate new UUID (Universally Unique IDentifier) strings.

More information: https://www.ss64.com/osx/uuidgen.html

- Generate a UUID string:

```
uuidgen
```

valet

A Laravel development environment that allows hosting sites via local tunnels on **`http://<example>.test`**

More information: https://laravel.com/docs/8.x/valet

- Start the valet daemon:

  ```
  valet start
  ```

- Register the current working directory as a path that Valet should search for sites:

  ```
  valet park
  ```

- View 'parked' paths:

  ```
  valet paths
  ```

- Serve a single site instead of an entire directory:

  ```
  valet link app-name
  ```

- Share a project via an Ngrok tunnel:

  ```
  valet share
  ```

W

S how who is logged on and what they are doing.

P rint user login, T T Y , remote host, login time, idle time, current process.

- S how logged-in users info:

  ```
  w
  ```

- S how logged-in users info without a header:

  ```
  w -h
  ```

- S how info about logged-in users, sorted by their idle time:

  ```
  w -i
  ```

wacaw

A little command-line tool for macOS that allows you to capture both still pictures and video from an attached camera.

More information: http://webcam-tools.sourceforge.net.

- Take a picture from webcam:

```
wacaw {{filename}}
```

- Record a video:

```
wacaw --video {{filename}} -D {{duration_in_seconds}}
```

- Take a picture with custom resolution:

```
wacaw -x {{width}} -y {{height}} {{filename}}
```

- Copy image just taken to clipboard:

```
wacaw --to-clipboard
```

- List the devices available:

```
wacaw -L
```

whatis

T ool that searches a set of database files containing short descriptions of system commands for keywords.

More information: http://www.linfo.org/whatis.html.

- S earch for information about keyword:

  ```
  w hatis {{k eyw ord}}
  ```

- S earch for information about multiple keywords:

  ```
  w hatis {{first_k eyw ord}} {{second_k eyw ord}}
  ```

whence

A zsh builtin to indicate how a given command would be interpreted.

- Interpret {{command}}, with expansion if defined as an `alias` (similar to the `command -v` builtin):

  ```
  whence {{command}}
  ```

- Display type of {{command}}, with location if defined as a function, or binary (equivalent to the `type` and `command -V` builtins):

  ```
  whence -v {{command}}
  ```

- Same as above, except display content of shell functions instead of location (equivalent to `which` builtin):

  ```
  whence -c {{command}}
  ```

- Same as above, but show all occurrences on command path (equivalent to the `where` builtin):

  ```
  whence -ca {{command}}
  ```

- Search only the `PATH` for {{command}}, ignoring builtins, aliases or shell functions (equivalent to the `where` command):

  ```
  whence -p {{command}}
  ```

whereis

Locate the binary, source, and manual page files for a command.

- Locate binary, source and man pages for ssh:

  ```
  w hereis {{ssh}}
  ```

- Locate binary and man pages for ls:

  ```
  w hereis - b m  {{ls}}
  ```

- Locate source of gcc and man pages for G it:

  ```
  w hereis - s {{g cc}} - m  {{g it}}
  ```

- Locate binaries for gcc in `/u sr/b in`only:

  ```
  w hereis - b  - B  {{/u sr/b in/}} - f {{g cc}}
  ```

wifi-password

Get the password of the wifi.

More information: https://github.com/rauchg/wifi-password.

- Get the password for the wifi you are currently logged onto:

  ```
  wifi-password
  ```

- Get the password for the wifi with a specific SSID:

  ```
  wifi-password {{ssid}}
  ```

- Print only the password as output:

  ```
  wifi-password -q
  ```

xar

Manage .xar archives.

More information: https://manned.org/xar.

- Create a xar archive of all files in a given directory:

```
xar -cf {{archive.xar}} {{path/to/directory}}
```

- List[t] the contents of a given xar archive:

```
xar -tf {{archive.xar}}
```

- Extract the contents of a given xar archive to the current directory:

```
xar -xf {{archive.xar}}
```

xattr

Utility to work with extended filesystem attributes.

- List key:value extended attributes for a given file:

  ```
  xattr -l {{file}}
  ```

- Write an attribute for a given file:

  ```
  xattr -w {{attribute_key}} {{attribute_value}} {{file}}
  ```

- Delete an attribute from a given file:

  ```
  xattr -d {{com.apple.quarantine}} {{file}}
  ```

- Delete all extended attributes from a given file:

  ```
  xattr -c {{file}}
  ```

- Recursively delete an attribute in a given directory:

  ```
  xattr -rd {{attribute_key}} {{directory}}
  ```

xcode-select

Switch between different versions of Xcode and the included developer tools.

Also used to update the path to Xcode if it is moved after installation.

- Install Xcode's command-line tools:

  ```
  xcode-select --install
  ```

- Select a given path as the active developer directory:

  ```
  xcode-select -s {{path/to/Xcode.app/Contents/Developer}}
  ```

- Select a given Xcode instance and use its developer directory as the active one:

  ```
  xcode-select -s {{path/to/Xcode.app}}
  ```

- Print the currently selected developer directory:

  ```
  xcode-select -p
  ```

- Discard any user-specified developer directory so that it will be found via the default search mechanism:

  ```
  sudo xcode-select -r
  ```

xcodebuild

Build Xcode projects.

- Build workspace:

  ```
  xcodebuild -workspace {{workspace_name.workspace}} -scheme
  {{scheme_name}} -configuration {{configuration_name}} clean
  build SYMROOT={{SYMROOT_path}}
  ```

- Build project:

  ```
  xcodebuild -target {{target_name}} -configuration
  {{configuration_name}} clean build SYMROOT={{SYMROOT_path
  ```

- Show SDKs:

  ```
  xcodebuild -showsdks
  ```

xctool

Tool for building Xcode projects.

More information: https://github.com/facebook/xctool.

- Build a single project without any workspace:

  ```
  xctool -project {{YourProject.xcodeproj}} -scheme
  {{YourScheme}} build
  ```

- Build a project that is part of a workspace:

  ```
  xctool -workspace {{YourWorkspace.xcworkspace}} -scheme
  {{YourScheme}} build
  ```

- Clean, build and execute all the tests:

  ```
  xctool -workspace {{YourWorkspace.xcworkspace}} -scheme
  {{YourScheme}} clean build test
  ```

xed

Opens files for editing in Xcode.

- Open file in Xcode:

  ```
  xed {{file1}}
  ```

- Open file(s) in Xcode, create if it doesn't exist:

  ```
  xed -c {{filename1}}
  ```

- Open a file in Xcode and jump to line number 75:

  ```
  xed -l 75 {{filename}}
  ```

xip

Create or expand compressed files in a secure xip archive.

O nly archives signed by Apple are trusted, so this tool should not be used to create archives.

- E xpand the archive into the current working directory:

  ```
  xip - - expand {{path/to/file. xip}}
  ```

xsltproc

Transform XML with XSLT to produce output (usually HTML or XML).

- Transform an XML file with a specific XSLT stylesheet:

  ```
  xsltproc --output {{output.html}} {{stylesheet.xslt}} {{xmlfile.xml}}
  ```

- Pass a value to a parameter in the stylesheet:

  ```
  xsltproc --output {{output.html}} --stringparam {{name}} {{value}} {{stylesheet.xslt}} {{xmlfile.xml}}
  ```

yaa

Create and manipulate YAA archives.

- Create an archive from a directory:

```
yaa archive -d {{path/to/directory}} -o {{path/to/output.yaa}}
```

- Create an archive from a file:

```
yaa archive -i {{path/to/file}} -o {{path/to/output.yaa}}
```

- Extract an archive to the current directory:

```
yaa extract -i {{path/to/archive.yaa}}
```

- List the contents of an archive:

```
yaa list -i {{path/to/archive.yaa}}
```

- Create an archive with a specific compression algorithm:

```
yaa archive -a {{algorithm}} -d {{path/to/directory}} -o {{path/to/output.yaa}}
```

- Create an archive with an 8 MB block size:

```
yaa archive -b {{8m}} -d {{path/to/directory}} -o {{path/to/output.yaa}}
```

yabai

A tiling window manager for macOS based on binary space partitioning.

More information: https://github.com/koekeishiya/yabai.

- Set the layout to bsp:

  ```
  yabai -m config layout {{bsp}}
  ```

- Set the window gap to 10pt:

  ```
  yabai -m config window_gap {{10}}
  ```

- Enable opacity:

  ```
  yabai -m config window_opacity on
  ```

- Disable window shadow:

  ```
  yabai -m config window_shadow off
  ```

- Enable status bar:

  ```
  yabai -m config status_bar on
  ```

yank

Read input from stdin and display a selection interface that allows a field to be selected and copied to the clipboard.

- Yank using the default delimiters (\f, \n, \r, \s, \t):

  ```
  {{sudo dmesg}} | yank
  ```

- Yank an entire line:

  ```
  {{sudo dmesg}} | yank -l
  ```

- Yank using a specific delimiter:

  ```
  {{echo hello=world}} | yank -d {{=}}
  ```

- Only yank fields matching a specific pattern:

  ```
  {{ps ux}} | yank -g "{{[0-9]+}}"
  ```

S u nos

devfsadm

Administration command for `/dev` Maintains the `/dev` namespace.

More information: https://www.unix.com/man-page/sunos/1_m/devfsadm

- Scan for new disks:

  ```
  devfsadm -c disk
  ```

- Cleanup any dangling /dev links and scan for new device:

  ```
  devfsadm -C -v
  ```

- Dry-run - output what would be changed but make no modifications:

  ```
  devfsadm -C -v -n
  ```

dmesg

Write the kernel messages to standard output.

More information: https://www.unix.com/man-page/sunos/1_m/dmesg

- Show kernel messages:

```
dmesg
```

- Show how much physical memory is available on this system:

```
dmesg | grep -i memory
```

- Show kernel messages 1 page at a time:

```
dmesg | less
```

prctl

Get or set the resource controls of running processes,.

Tasks, and projects.

More information: https://www.unix.com/man-page/sunos/1/prctl

- Examine process limits and permissions:

  ```
  prctl {{PID}}
  ```

- Examine process limits and permissions in machine parseable format:

  ```
  prctl -P {{PID}}
  ```

- Get specific limit for a running process:

  ```
  prctl -n process.max-file-descriptor {{PID}}
  ```

prstat

Report active process statistics.

More information: https://www.unix.com/man-page/sunos/1_m/prstat

- Examine all processes and reports statistics sorted by CPU usage:

  ```
  prstat
  ```

- Examine all processes and reports statistics sorted by memory usage:

  ```
  prstat -s rss
  ```

- Report total usage summary for each user:

  ```
  prstat -t
  ```

- Report microstate process accounting information:

  ```
  prstat -m
  ```

- Print out a list of top 5 CPU using processes every second:

  ```
  prstat -c -n 5 -s cpu 1
  ```

snoop

N etwork packet sniffer.

S unO S eq uivalent of tcpdump.

More information: https://www.unix.com/man-page/sunos/1 m/snoop

- Capture packets on a specific network interface:

```
snoop - d {{e1 0 0 0 g 0 }}
```

- S ave captured packets in a file instead of displaying them:

```
snoop - o {{filenam e}}
```

- Display verbose protocol layer summary of packets from a file:

```
snoop - V - i {{filenam e}}
```

- Capture network packets that come from a hostname and go to a given port:

```
snoop to port {{port}} from host {{hostnam e}}
```

- Capture and show an hex-dump of network packets exchanged between two IP addresses:

```
snoop - x0 - p4 {{ip_address_1 }} {{ip_address_2 }}
```

svcadm

Manipulate service instances.

More information: https://www.unix.com/man-page/linux/1_m/svcadm

- Enable a service in the service database:

  ```
  svcadm enable {{service name}}
  ```

- Disable service:

  ```
  svcadm disable {{service name}}
  ```

- Restart a running service:

  ```
  svcadm restart {{service name}}
  ```

- Command service to re-read configuration files:

  ```
  svcadm refresh {{service name}}
  ```

- Clear a service from maintenance state and command it to start:

  ```
  svcadm clear {{service name}}
  ```

svccfg

Import, export, and modify service configurations.

More information: https://www.unix.com/man-page/linux/1_m/svccfg

- V alidate configuration file:

```
sv ccfg  v alidate {{sm f. xm l}}
```

- E xport service configurations to file:

```
sv ccfg  export {{serv icenam e}} >  {{sm f. xm l}}
```

- Import/update service configurations from file:

```
sv ccfg  im port {{sm f. xm l}}
```

SVCS

List information about running services.

More information: https://www.unix.com/man-page/linux/1 /svcs

- List all running services:

```
sv cs
```

- List services that are not running:

```
sv cs - v x
```

- List information about a service:

```
sv cs apache
```

- S how location of service log file:

```
sv cs - L  apache
```

- Display end of a service log file:

```
tail $ ( sv cs - L  apache)
```

truss

T roubleshooting tool for tracing system calls.

S unO S eq uivalent of strace.

More information: https://www.unix.com/man-page/linux/1 /truss

- S tart tracing a program by executing it, following all child processes:

```
tru ss - f {{prog ram }}
```

- S tart tracing a specific process by its P ID:

```
tru ss - p {{pid}}
```

- S tart tracing a program by executing it, showing arguments and environment variables:

```
tru ss - a - e {{prog ram }}
```

- Count time, calls, and errors for each system call and report a summary on program exit:

```
tru ss - c - p {{pid}}
```

- T race a process filtering output by system call:

```
tru ss - p {{pid}} - t {{system _call_nam e}}
```

Windows

assoc

Display or modify file extension associations.

More information: https://docs.microsoft.com/windows-server/administration/windows-commands/assoc.

- Display all associated filetypes:

```
assoc
```

- Display the associated filetype for a specific extension:

```
assoc {{. txt}}
```

- Modify the associated filetype for a specific extension:

```
assoc {{. txt}}= {{txtfile}}
```

attrib

Displays or changes file and directory attributes.

More information: https://docs.microsoft.com/windows-server/administration/windows-commands/attrib.

- Display the attributes of the files in the current directory:

```
attrib
```

- Display the attributes of the files in the current directory and sub-directories:

```
attrib /S
```

- Display the attributes of the files and directories in the current directory and sub-directories:

```
attrib /S /D
```

- Add the read-only attribute to a file:

```
attrib +R {{document.txt}}
```

- Remove the system and hidden attributes of a file:

```
attrib -S -H {{document.txt}}
```

- Add the hidden attribute to a directory:

```
attrib +H {{path\to\directory}}
```

az copy

A file transfer tool for uploading to Azure Cloud Storage Accounts.

More information: https://docs.microsoft.com/azure/storage/common/storage-use-azcopy-v10

- Log in to an Azure Tenant:

  ```
  azcopy login
  ```

- Upload a local file:

  ```
  azcopy copy '{{path/to/source/file}}' 'https://{{storage_account_name}}.blob.core.windows.net/{{container_name}}/{{blob_name}}'
  ```

- Upload files with `.txt` and `.jpg` extensions:

  ```
  azcopy copy '{{path/to/source}}' 'https://{{storage_account_name}}.blob.core.windows.net/{{container_name}}' --include-pattern '{{*.txt;*.jpg}}'
  ```

- Copy a container directly between two Azure storage accounts:

  ```
  azcopy copy 'https://{{source_storage_account_name}}.blob.core.windows.net/{{container_name}}' 'https://{{destination_storage_account_name}}.blob.core.windows.net/{{container_name}}'
  ```

- Synchronize a local directory and delete files in the destination if they no longer exist in the source:

  ```
  azcopy sync '{{path/to/source}}' 'https://{{storage_account_name}}.blob.core.windows.net/{{container_name}}' --recursive --delete-destination=true
  ```

- Display detailed usage information:

  ```
  azcopy --help
  ```

cd

Displays the name of or changes the current working directory.

More information: https://docs.microsoft.com/windows-server/administration/windows-commands/cd.

- Go to a directory in the same drive:

  ```
  cd {{path/to/directory}}
  ```

- Display the name of the current directory:

  ```
  cd
  ```

- Go up to the parent of the current directory:

  ```
  cd . .
  ```

- Go to a directory in a different drive:

  ```
  cd {{path/to/directory}} /d
  ```

chkdsk

Check file system and volume metadata for errors.

More information: https://docs.microsoft.com/windows-server/administration/windows-commands/chkdsk.

- Specify the drive letter (followed by a colon), mount point, or volume name to check:

```
chkdsk {{volume}}
```

- Fix errors on a specific volume:

```
chkdsk {{volume}} /f
```

- Dismount a specific volume before checking:

```
chkdsk {{volume}} /x
```

- Change the log file size to the specified size (only for NTFS):

```
chkdsk /l{{size}}
```

choco-apikey

Manage API keys for Chocolatey sources.

More information: https://chocolatey.org/docs/commands-apikey.

- Display a list of sources and their API keys:

  ```
  choco apikey
  ```

- Display a specific source and its API key:

  ```
  choco apikey --source "{{source_url}}"
  ```

- Set an API key for a source:

  ```
  choco apikey --source "{{source_url}}" --key "{{api_key}}"
  ```

- Remove an API key for a source:

  ```
  choco apikey --source "{{source_url}}" --remove
  ```

choco feature

Interact with features with Chocolatey.

More information: https://chocolatey.org/docs/commands-feature.

- Display a list of available features:

  ```
  choco featu re list
  ```

- E nable a feature:

  ```
  choco featu re enab le - - nam e {{nam e}}
  ```

- Disable a feature:

  ```
  choco featu re disab le - - nam e {{nam e}}
  ```

choco info

Display detailed information about a package with Chocolatey.

More information: https://chocolatey.org/docs/commands-info.

- Display information on a specific package:

  ```
  choco info {{package}}
  ```

- Display information for a local package only:

  ```
  choco info {{package}} --local-only
  ```

- Specify a custom source to receive packages information from:

  ```
  choco info {{package}} --source {{source_url|alias}}
  ```

- Provide a username and password for authentication:

  ```
  choco info {{package}} --user {{username}} --password {{password}}
  ```

choco install

Install one or more packages with Chocolatey.

More information: https://chocolatey.org/docs/commands-install.

- Install one or more space-separated packages:

  ```
  choco install {{pack ag e( s) }}
  ```

- Install packages from a custom configuration file:

  ```
  choco install {{path/to/pack ag es. config }}
  ```

- Install a specific nuspec or nupkg file:

  ```
  choco install {{path/to/file}}
  ```

- Install a specific version of a package:

  ```
  choco install {{pack ag e}} - - v ersion {{v ersion}}
  ```

- Allow installing multiple versions of a package:

  ```
  choco install {{pack ag e}} - - allow - m u ltiple
  ```

- Confirm all prompts automatically:

  ```
  choco install {{pack ag e}} - - yes
  ```

- S pecify a custom source to receive packages from:

  ```
  choco install {{pack ag e}} - - sou rce {{sou rce_u rl| alias}}
  ```

- P rovide a username and password for authentication:

  ```
  choco install {{pack ag e}} - - u ser {{u sernam e}} - - passw ord
  {{passw ord}}
  ```

choco list

Display a list of packages with Chocolatey.

More information: https://chocolatey.org/docs/commands-list.

- Display all available packages:

  ```
  choco list
  ```

- Display all locally installed packages:

  ```
  choco list - - local- only
  ```

- Display a list including local programs:

  ```
  choco list - - inclu de- prog ram s
  ```

- Display only approved packages:

  ```
  choco list - - approv ed- only
  ```

- S pecify a custom source to display packages from:

  ```
  choco list - - sou rce {{sou rce_u rl| alias}}
  ```

- P rovide a username and password for authentication:

  ```
  choco list - - u ser {{u sernam e}} - - passw ord {{passw ord}}
  ```

choco new

Generate new package specification files with Chocolatey.

More information: https://chocolatey.org/docs/commands-new.

- Create a new package skeleton:

```
choco new {{package name}}
```

- Create a new package with a specific version:

```
choco new {{package name}} --version {{version}}
```

- Create a new package with a specific maintainer name:

```
choco new {{package name}} --maintainer {{maintainer_name}}
```

- Create a new package in a custom output directory:

```
choco new {{package name}} --output-directory {{path/to/directory}}
```

- Create a new package with specific 32-bit and 64-bit installer URLs:

```
choco new {{package name}} url="{{url}}" url64="{{url}}"
```

choco outdated

Check for outdated packages with Chocolatey.

More information: https://chocolatey.org/docs/commands-outdated.

- Display a list of outdated packages in table format:

  ```
  choco ou tdated
  ```

- Ignore pinned packages in the output:

  ```
  choco ou tdated - - ig nore- pinned
  ```

- S pecify a custom source to check packages from:

  ```
  choco ou tdated - - sou rce {{sou rce_u rl| alias}}
  ```

- P rovide a username and password for authentication:

  ```
  choco ou tdated - - u ser {{u sernam e}} - - passw ord {{passw ord}}
  ```

choco pack

Package a NuGet specification into a nupkg file.

More information: https://chocolatey.org/docs/commands-pack.

- Package a NuGet specification to a nupkg file:

  ```
  choco pack {{path/to/specification}}
  ```

- Package a NuGet specification specifying the version of the resulting file:

  ```
  choco pack {{path/to/specification}} --version {{version}}
  ```

- Package a NuGet specification to a specific directory:

  ```
  choco pack {{path/to/specification}} --outputdirectory {{path/to/outputdirectory}}
  ```

choco pin

Pin a package at a specific version with Chocolatey.

Pinned packages are skipped automatically when upgrading.

More information: https://chocolatey.org/docs/commands-pin.

- Display a list of pinned packages and their versions:

  ```
  choco pin list
  ```

- Pin a package at its current version:

  ```
  choco pin add --name {{package}}
  ```

- Pin a package at a specific version:

  ```
  choco pin add --name {{package}} --version {{version}}
  ```

- Remove a pin for a specific package:

  ```
  choco pin remove --name {{package}}
  ```

choco search

Search for a local or remote package with Chocolatey.

More information: https://chocolatey.org/docs/commands-search.

- Search for a package:

```
choco search {{query}}
```

- Search for a package locally:

```
choco search {{query}} --local-only
```

- Only include exact matches in the results:

```
choco search {{query}} --exact
```

- Confirm all prompts automatically:

```
choco search {{query}} --yes
```

- Specify a custom source to search for packages in:

```
choco search {{query}} --source {{source_url|alias}}
```

- Provide a username and password for authentication:

```
choco search {{query}} --user {{username}} --password {{password}}
```

choco source

Manage sources for packages with Chocolatey.

More information: https://chocolatey.org/docs/commands-source.

- List currently available sources:

  ```
  choco sou rce list
  ```

- Add a new package source:

  ```
  choco sou rce add - - nam e {{nam e}} - - sou rce {{u rl}}
  ```

- Add a new package source with credentials:

  ```
  choco sou rce add - - nam e {{nam e}} - - sou rce {{u rl}} - - u ser
  {{u sernam e}} - - passw ord {{passw ord}}
  ```

- Add a new package source with a client certificate:

  ```
  choco sou rce add - - nam e {{nam e}} - - sou rce {{u rl}} - - cert
  {{path/to/certificate}}
  ```

- E nable a package source:

  ```
  choco sou rce enab le - - nam e {{nam e}}
  ```

- Disable a package source:

  ```
  choco sou rce disab le - - nam e {{nam e}}
  ```

- R emove a package source:

  ```
  choco sou rce rem ov e - - nam e {{nam e}}
  ```

choco uninstall

U ninstall one or more packages with Chocolatey.

More information: https://chocolatey.org/docs/commands-uninstall.

- U ninstall one or more space-separated packages:

  ```
  choco u ninstall {{pack ag e( s) }}
  ```

- U ninstall a specific version of a package:

  ```
  choco u ninstall {{pack ag e}} - - v ersion {{v ersion}}
  ```

- Confirm all prompts automatically:

  ```
  choco u ninstall {{pack ag e}} - - yes
  ```

- R emove all dependencies when uninstalling:

  ```
  choco u ninstall {{pack ag e}} - - rem ov e- dependencies
  ```

- U ninstall all packages:

  ```
  choco u ninstall all
  ```

choco upgrade

Upgrade one or more packages with Chocolatey.

More information: https://chocolatey.org/docs/commands-upgrade.

- Upgrade one or more space-separated packages:

  ```
  choco upgrade {{package(s)}}
  ```

- Upgrade to a specific version of a package:

  ```
  choco upgrade {{package}} --version {{version}}
  ```

- Upgrade all packages:

  ```
  choco upgrade all
  ```

- Upgrade all except specified comma-separated packages:

  ```
  choco upgrade all --except "{{package(s)}}"
  ```

- Confirm all prompts automatically:

  ```
  choco upgrade {{package}} --yes
  ```

- Specify a custom source to receive packages from:

  ```
  choco upgrade {{package}} --source {{source_url|alias}}
  ```

- Provide a username and password for authentication:

  ```
  choco upgrade {{package}} --user {{username}} --password {{password}}
  ```

choco

A command-line interface for the Chocolatey package manager.

See `choco install`, `choco upgrade` and other pages for additional information.

More information: https://chocolatey.org.

- Execute Chocolatey command:

  `choco {{command}}`

- Call general help:

  `choco -?`

- Call help on a specific command:

  `choco {{command}} -?`

- Check the Chocolatey version:

  `choco --version`

choice

Prompts the user to select one item from a list of single-character choices in a batch program, and then returns the index of the selected choice.

If used without parameters, choice displays the default choices Y and N .

More information: https://docs.microsoft.com/en-us/windows-server/administration/windows-commands/choice.

- A,B and C as list of choices to be used:

  ```
  choice /c {{A B C }}
  ```

- Use the default [Y ,N] list of choices:

  ```
  choice
  ```

- Specify that the choices are case-sensitive:

  ```
  choice /C S {{A aB b }}
  ```

- Specify the number of seconds to pause before using the default choice specified by `/d`:

  ```
  choice /C {{A aB b }} /t {{3 }} /d {{b }}
  ```

- Specify a message to display before the list of choices. If `/m` is not specified, only the choice prompt is displayed:

  ```
  choice /m {{m essag e}} /C {{A B C }}
  ```

- Display help message:

  ```
  choice /?
  ```

cinst

This command is an alias of `choco install`.

- View documentation for the original command:

  `tldr choco install`

cipher

Encrypt or decrypt files on NTFS drives.

More information: https://docs.microsoft.com/windows-server/administration/windows-commands/cipher.

- Encrypt a file or directory:

```
cipher /e: {{path/to/file_or_directory}}
```

- Decrypt a file or directory:

```
cipher /d: {{path/to/file_or_directory}}
```

- Securely remove a file or directory:

```
cipher /w : {{path/to/file_or_directory}}
```

clip

Copy input content to the Windows clipboard.

More information: https://docs.microsoft.com/windows-server/administration/windows-commands/clip.

- Pipe command-line output to the Windows clipboard:

```
{{dir}} | clip
```

- Copy the contents of a file to the Windows clipboard:

```
clip < {{path/to/file.ext}}
```

- Copy text with a trailing newline to the Windows clipboard:

```
echo {{some text}} | clip
```

- Copy text without a trailing newline to the Windows clipboard:

```
echo | set /p="some text" | clip
```

clist

This command is an alias of `choco list`

- View documentation for the original command:

  ```
  tldr choco list
  ```

cls

Clears the screen.

More information: https://docs.microsoft.com/windows-server/administration/windows-commands/cls.

- Clear the screen:

```
cls
```

cmd

The Windows command interpreter.

More information: https://docs.microsoft.com/windows-server/administration/windows-commands/cmd.

- Start a new instance of the command interpreter:

  ```
  cmd
  ```

- Run the specified command and then exit:

  ```
  cmd /c "{{command}}"
  ```

- Run the specified command and then enter an interactive shell:

  ```
  cmd /k "{{command}}"
  ```

- Disable the usage of `echo` in command output:

  ```
  cmd /q
  ```

- Enable or disable command extensions:

  ```
  cmd /e:{{on|off}}
  ```

- Enable or disable file or directory autocompletion:

  ```
  cmd /f:{{on|off}}
  ```

- Enable or disable environment variable expansion:

  ```
  cmd /v:{{on|off}}
  ```

- Force output to use Unicode encoding:

  ```
  cmd /u
  ```

cmstp

A command-line tool for managing connection service profiles.

More information: https://docs.microsoft.com/windows-server/administration/windows-commands/cmstp.

- Install a specific profile:

  ```
  cm stp " {{path/to/profile}}"
  ```

- Install without creating a desktop shortcut:

  ```
  cm stp /ns " {{path/to/profile}}"
  ```

- Install without checking for dependencies:

  ```
  cm stp /nf " {{path/to/profile}}"
  ```

- O nly install for the current user:

  ```
  cm stp /su  " {{path/to/profile}}"
  ```

- Install for all users (req uires administrator privileges):

  ```
  cm stp /au  " {{path/to/profile}}"
  ```

- Install silently without any prompts:

  ```
  cm stp /s " {{path/to/profile}}"
  ```

- U ninstall a specific profile:

  ```
  cm stp /u  " {{path/to/profile}}"
  ```

- U ninstall silently without a confirmation prompt:

  ```
  cm stp /u  /s " {{path/to/profile}}"
  ```

color

S et the console foreground and background colors.

More information: https://docs.microsoft.com/windows-server/administration/windows-commands/color.

- S et the console colors to the default values:

  ```
  color
  ```

- List available color values and detailed information:

  ```
  color /?
  ```

- S et the console foreground and background to a specific color:

  ```
  color {{foreg rou nd_code}}{{b ack g rou nd_code}}
  ```

comp

Compare the contents of two files or sets of files.

U se wildcards (*) to compare sets of files.

More information: https://docs.microsoft.com/windows-server/administration/windows-commands/comp.

- Compare files interactively:

```
com p
```

- Compare two specified files:

```
com p {{path/to/file_1 }} {{path/to/file_2 }}
```

- Compare two sets of files:

```
com p {{path/to/directory_1 /* }} {{path/to/directory_2 /* }}
```

- Display differences in decimal format:

```
com p /d {{path/to/file_1 }} {{path/to/file_2 }}
```

- Display differences in AS CII format:

```
com p /a {{path/to/file_1 }} {{path/to/file_2 }}
```

- Display line numbers for differences:

```
com p /l {{path/to/file_1 }} {{path/to/file_2 }}
```

- Compare files case-insensitively:

```
com p /c {{path/to/file_1 }} {{path/to/file_2 }}
```

- Compare only the first 5 lines of each file:

```
com p /n= {{5 }} {{path/to/file_1 }} {{path/to/file_2 }}
```

cuninst

This command is an alias of `choco uninstall`.

- View documentation for the original command:

  `tldr choco uninstall`

del

Delete one or more files.

More information: https://docs.microsoft.com/windows-server/administration/windows-commands/del.

- Delete one or more space-separated files or patterns:

```
del {{file_pattern}}
```

- P rompt for confirmation before deleting each file:

```
del {{file_pattern}} /p
```

- F orce the deletion of read-only files:

```
del {{file_pattern}} /f
```

- R ecursively delete file(s) from all subdirectories:

```
del {{file_pattern}} /s
```

- Do not prompt when deleting files based on a global wildcard:

```
del {{file_pattern}} /q
```

- Display the help and list available attributes:

```
del /?
```

- Delete files based on specified attributes:

```
del {{file_pattern}} /a {{attrib u te}}
```

dir

List directory contents.

More information: https://docs.microsoft.com/windows-server/administration/windows-commands/dir.

- S how the contents of the current directory:

  ```
  dir
  ```

- S how the contents of a given directory:

  ```
  dir {{path/to/directory}}
  ```

- S how the contents of the current directory, including hidden ones:

  ```
  dir /A
  ```

- S how the contents of a given directory, including hidden ones:

  ```
  dir {{path/to/directory}} /A
  ```

doskey

Manage macros, windows commands and command-lines.

More information: https://docs.microsoft.com/windows-server/administration/windows-commands/doskey.

- List available macros:

```
dosk ey /m acros
```

- Create a new macro:

```
dosk ey {{nam e}} =  " {{com m and}}"
```

- Create a new macro for a specific executable:

```
dosk ey /exenam e= {{execu tab le}} {{nam e}} =  " {{com m and}}"
```

- R emove a macro:

```
dosk ey {{nam e}} =
```

- Display all commands that are stored in memory:

```
dosk ey /history
```

- S ave macros to a file for portability:

```
dosk ey /m acros >  {{m acinit}}
```

- Load macros from a file:

```
dosk ey /m acrofile =  {{m acinit}}
```

driverq uery

Display information about installed device drivers.

More information: https://docs.microsoft.com/windows-server/administration/windows-commands/driverq uery

- Display a list of all installed device drivers:

```
driv erq u ery
```

- Display a list of drivers in the specified format:

```
driv erq u ery /fo {{tab le| list| csv }}
```

- Display a list of drivers with a column to indicate if they are signed:

```
driv erq u ery /si
```

- E xclude the header in the output list:

```
driv erq u ery /nh
```

- Display a list of drivers for a remote machine:

```
driv erq u ery /s {{hostnam e}} /u {{u sernam e}} /p {{passw ord}}
```

- Display a list of drivers with verbose information:

```
driv erq u ery /v
```

- Display detailed usage information:

```
driv erq u ery /?
```

eventcreate

Create custom entries in the event log.

Event IDs can be any number between 1 and 1000.

More information: https://docs.microsoft.com/windows-server/administration/windows-commands/eventcreate.

- Create a new event with a given id (1-1000) in the log:

```
eventcreate /t {{success|error|warning|information}} /id {{id}} /d "{{message}}"
```

- Create an event in a specific event log:

```
eventcreate /l {{log_name}} /t {{type}} /id {{id}} /d "{{message}}"
```

- Create an event with a specific source:

```
eventcreate /so {{source_name}} /t {{type}} /id {{id}} /d "{{message}}"
```

- Create an event in a remote machine's event log:

```
eventcreate /s {{hostname}} /u {{username}} /p {{password}} /t {{type}} /id {{id}} /d "{{message}}"
```

exit

Q uit the current CMD instance or the current batch file.

More information: https://docs.microsoft.com/windows-server/administration/windows-commands/exit.

- Q uit the current CMD instance:

  ```
  exit
  ```

- Q uit the current batch script:

  ```
  exit /b
  ```

- Q uit using a specific exit code:

  ```
  exit {{exit_code}}
  ```

expand

U ncompress one or more Windows Cabinet files.

More information: https://docs.microsoft.com/windows-server/administration/windows-commands/expand.

- U ncompress a single-file Cabinet file to the specified directory:

```
expand {{path/to/file. cab }} {{path/to/directory}}
```

- Display the list of files in a source Cabinet file:

```
expand {{path/to/file. cab }} {{path/to/directory}} - d
```

- U ncompress all files from the Cabinet file:

```
expand {{path/to/file. cab }} {{path/to/directory}} - f: *
```

- U ncompress a specific file from a Cabinet file:

```
expand {{path/to/file. cab }} {{path/to/directory}} - f: {{file}}
```

- Ignore the directory structure when uncompressing, and add them to a single directory:

```
expand {{path/to/file. cab }} {{path/to/directory}} - i
```

explorer

The Windows File Explorer.

More information: https://ss64.com/nt/explorer.html

- Open Windows Explorer:

  ```
  explorer
  ```

- Open Windows Explorer in the current directory:

  ```
  explorer .
  ```

- Open Windows Explorer in a specific directory:

  ```
  explorer {{path/to/directory}}
  ```

fc

Compare the differences between two files or sets of files.

Use wildcards (*) to compare sets of files.

More information: https://docs.microsoft.com/windows-server/administration/windows-commands/fc.

- Compare 2 specified files:

```
fc {{path/to/file_1}} {{path/to/file_2}}
```

- Perform a case-insensitive comparison:

```
fc /c {{path/to/file_1}} {{path/to/file_2}}
```

- Compare files as Unicode text:

```
fc /u {{path/to/file_1}} {{path/to/file_2}}
```

- Compare files as ASCII text:

```
fc /l {{path/to/file_1}} {{path/to/file_2}}
```

- Compare files as binary:

```
fc /b {{path/to/file_1}} {{path/to/file_2}}
```

- Disable tab-to-space expansion:

```
fc /t {{path/to/file_1}} {{path/to/file_2}}
```

- Compress whitespace (tabs and spaces) for comparisons:

```
fc /w {{path/to/file_1}} {{path/to/file_2}}
```

find

F ind a specified string in one or more files.

More information: https://docs.microsoft.com/windows-server/administration/windows-commands/find.

- F ind lines that contain a specified string:

```
find {{string }} {{path/to/file_or_directory}}
```

- Display lines that do not contain the specified string:

```
find {{string }} {{path/to/file_or_directory}} /v
```

- Display the count of lines that contain the specified string:

```
find {{string }} {{path/to/file_or_directory}} /c
```

- Display line numbers with the list of lines:

```
find {{string }} {{path/to/file_or_directory}} /n
```

findstr

Find specified text within one or more files.

More information: https://docs.microsoft.com/windows-server/administration/windows-commands/findstr.

- Find space-separated string(s) in all files:

```
findstr "{{query}}" *
```

- Find space-separated string(s) in a piped command's output:

```
{{dir}} | findstr "{{query}}"
```

- Find space-separated string(s) in all files recur[s]ively:

```
findstr /s "{{query}}" *
```

- Find strings using a case-insensitive search:

```
findstr /i "{{query}}" *"
```

- Find strings in all files using regular expressions:

```
findstr /r "{{expression}}" *
```

- Find a literal string (containing spaces) in all text files:

```
findstr /c:"{{query}}" *.txt
```

- Display the line number before each matching line:

```
findstr /n "{{query}}" *
```

- Display only the filenames that contain a match:

```
findstr /m "{{query}}" *
```

finger

Return information about one or more users on a specified system.

The remote system must be running the Finger service.

More information: https://docs.microsoft.com/windows-server/administration/windows-commands/finger.

- Display information about a specific user:

  ```
  finger {{user}}@{{host}}
  ```

- Display information about all users on the specified host:

  ```
  finger @{{host}}
  ```

- Display information in a longer format:

  ```
  finger {{user}}@{{host}} -l
  ```

- Display help information:

  ```
  finger /?
  ```

fondue

A command-line installer for optional Windows features.

More information: https://docs.microsoft.com/windows-server/administration/windows-commands/fondue.

- Enable a specific Windows feature:

  ```
  fondue /enable-feature:{{feature}}
  ```

- Hide all output messages to the user:

  ```
  fondue /enable-feature:{{feature}} /hide-ux:all
  ```

- Specify a caller process name for error reporting:

  ```
  fondue /enable-feature:{{feature}} /caller-name:{{name}}
  ```

forfiles

Select one or more files to execute a specified command on.

More information: https://docs.microsoft.com/windows-server/administration/windows-commands/forfiles.

- Search for files in the current directory:

```
forfiles
```

- Search for files in a specific directory:

```
forfiles /p {{path/to/directory}}
```

- Run the specified command for each file:

```
forfiles /c " {{command}}"
```

- Search for files using a specific glob mask:

```
forfiles /m {{glob_pattern}}
```

- Search for files recursively:

```
forfiles /s
```

- Search for files older than 5 days:

```
forfiles /d {{+5}}
```

ftp

Interactively transfer files between a local and remote FTP server.

More information: https://docs.microsoft.com/windows-server/administration/windows-commands/ftp.

- Connect to a remote FTP server interactively:

```
ftp {{host}}
```

- Log in as an anonymous user:

```
ftp -A {{host}}
```

- Disable automatic login upon initial connection:

```
ftp -n {{host}}
```

- Run a file containing a list of FTP commands:

```
ftp -s:{{path/to/file}} {{host}}
```

- Download multiple files (glob expression):

```
mget {{*.png}}
```

- Upload multiple files (glob expression):

```
mput {{*.zip}}
```

- Delete multiple files on the remote server:

```
mdelete {{*.txt}}
```

- Display detailed help:

```
ftp --help
```

ftype

Display or modify file types used for file extension association.

More information: https://docs.microsoft.com/windows-server/administration/windows-commands/ftype.

- Display a list of all file types:

  ```
  ftype
  ```

- Display the associated program for a specific file type:

  ```
  ftype {{file_type}}
  ```

- S et the associated program for a specific file type:

  ```
  ftype {{file_type}}= " {{path/to/execu tab le_com m and}}"
  ```

G et-Content

G et the content of the item at the specified location.

More information: https://docs.microsoft.com/powershell/module/microsoft.powershell.management/get-content.

- Display the content of a file:

```
G et- C ontent - P ath {{path/to/file}}
```

- Display the first few lines of a file:

```
G et- C ontent - P ath {{path/to/file}} - T otalC ou nt {{cou nt}}
```

- Display the content of the file and keep reading from it until `C trl + C` is pressed:

```
G et- C ontent - P ath {{path/to/file}} - W ait
```

getmac

Display the MAC addresses of a system.

More information: https://docs.microsoft.com/windows-server/administration/windows-commands/getmac.

- Display the MAC addresses for the current system:

```
g etm ac
```

- Display the details in a specific format:

```
g etm ac /fo {{tab le| list| csv }}
```

- E xclude the header in the output list:

```
g etm ac /nh
```

- Display the MAC addresses for a remote machine:

```
g etm ac /s {{hostnam e}} /u  {{u sernam e}} /p {{passw ord}}
```

- Display the MAC addresses with verbose information:

```
g etm ac /v
```

- Display detailed usage information:

```
g etm ac /?
```

gpupdate

A tool to check and apply Windows Group Policy settings.

More information: https://docs.microsoft.com/windows-server/administration/windows-commands/gpupdate.

- Check and apply updated Group Policy settings:

  ```
  gpupdate
  ```

- Specify the target Group Policy settings to check for update:

  ```
  gpupdate /target:{{computer|user}}
  ```

- Force all Group Policy settings to be reapplied:

  ```
  gpupdate /force
  ```

- Display detailed usage information:

  ```
  gpupdate /?
  ```

ipconfig

Display and manage the network configuration of Windows.

More information: https://docs.microsoft.com/windows-server/administration/windows-commands/ipconfig.

- Show a list of network adapters:

```
ipconfig
```

- Show a detailed list of network adapters:

```
ipconfig /all
```

- Renew the IP addresses for a network adapter:

```
ipconfig /renew {{adapter}}
```

- Free up the IP addresses for a network adapter:

```
ipconfig /release {{adapter}}
```

- Remove all data from the DNS cache:

```
ipconfig /flushdns
```

iscc

Compiler for Inno S etup installers.

It compiles an Inno S etup scripts into an Windows installer executable.

More information: https://j rsoftware.org/isinfo.php

- Compile an Inno S etup script:

  ```
  iscc {{path/to/file. iss}}
  ```

- Q uietly compile an Inno S etup installer:

  ```
  iscc /Q {{path/to/file. iss}}
  ```

- Compile a signed Inno S etup installer:

  ```
  iscc /S = {{nam e}}= {{com m and}} {{path/to/file. iss}}
  ```

logoff

Terminate a login session.

More information: https://docs.microsoft.com/windows-server/administration/windows-commands/logoff.

- Terminate the current session:

```
log off
```

- Terminate a session by its name or id:

```
log off {{session_name| session_id}}
```

- Terminate a session on a specific server connected through RDP:

```
log off {{session_name| session_id}} /server: {{servername}}
```

mkdir

Creates a directory.

More information: https://docs.microsoft.com/windows-server/administration/windows-commands/mkdir.

- Create a directory:

```
m k dir {{directory_nam e}}
```

- R ecursively create a nested directory tree:

```
m k dir {{path/to/su b _directory_nam e}}
```

mklink

Create symbolic links.

More information: https://docs.microsoft.com/windows-server/administration/windows-commands/mklink.

- Create a symbolic link to a file:

```
m k link  {{path/to/link }} {{path/to/sou rce_file}}
```

- Create a symbolic link to a directory:

```
m k link  /d {{path/to/link }} {{path/to/sou rce_directory}}
```

- Create a hard link to a file:

```
m k link  /h {{path/to/link }} {{path/to/sou rce_file}}
```

- Create a directory j unction:

```
m k link  /j  {{path/to/link }} {{path/to/sou rce_file}}
```

more

Display paginated output from stdin or a file.

More information: https://docs.microsoft.com/windows-server/administration/windows-commands/more.

- Display paginated output from stdin:

```
{{echo test}} |  m ore
```

- Display paginated output from one or more files:

```
m ore {{path/to/file}}
```

- Convert tabs to the specified number of spaces:

```
m ore {{path/to/file}} /t{{spaces}}
```

- Clear the screen before displaying the page:

```
m ore {{path/to/file}} /c
```

- Display the output starting at line 5 :

```
m ore {{path/to/file}} + {{5 }}
```

- E nable extended interactive mode (see help for usage):

```
m ore {{path/to/file}} /e
```

- Display full usage information:

```
m ore /?
```

mount

Mount Network File System (NFS) network shares.

More information: https://docs.microsoft.com/windows-server/administration/windows-commands/mount.

- Mount a share to the "Z" drive letter:

```
mount \\{{computer_name}}\{{share_name}} {{Z:}}
```

- Mount a share to the next available drive letter:

```
mount \\{{computer_name}}\{{share_name}} *
```

- Mount a share with a read timeout in seconds (defaults to 0.8, can be 0.9 or 1 to 60):

```
mount -o timeout={{seconds}} \\{{computer_name}}\{{share_name}} {{Z:}}
```

- Mount a share and retry up to 10 times if it fails:

```
mount -o retry={{retries}} \\{{computer_name}}\{{share_name}} {{Z:}}
```

- Mount a share with forced case sensitivity:

```
mount -o casesensitive \\{{computer_name}}\{{share_name}} {{Z:}}
```

- Mount a share as an anonymous user:

```
mount -o anon \\{{computer_name}}\{{share_name}} {{Z:}}
```

- Mount a share using a specific mount type:

```
mount -o mtype={{soft|hard}} \\{{computer_name}}\{{share_name}} {{Z:}}
```

msg

Send a message to a specific user or session.

More information: https://docs.microsoft.com/windows-server/administration/windows-commands/msg.

- Send a message to a specified user or session:

```
msg {{username|session_name|session_id}} {{message}}
```

- Send a message from stdin:

```
echo "{{message}}" | msg {{username|session_name|session_id}}
```

- Send a message to a specific server:

```
msg /server:{{server_name}} {{username|session_name|
session_id}}
```

- Send a message to all users of the current machine:

```
msg *
```

- Set a delay in seconds for a message:

```
msg /time:{{seconds}}
```

nfsstat

Display or reset the number of calls made to the NFS server.

More information: https://docs.microsoft.com/windows-server/administration/windows-commands/nfsstat.

- Display the recorded number of calls made to the NFS server:

```
nfsstat
```

- Reset the recorded number of calls made to the NFS server:

```
nfsstat -z
```

octo

Command-line tools for O ctopus Deploy.

More information: https://octopus.com/docs/octopus-rest-api/octo.exe-command-line.

- Create a package:

  ```
  octo pack  - - id= {{pack ag e_nam e}}
  ```

- P ush a package to a repository on the O ctopus server:

  ```
  octo pu sh - - pack ag e= {{pack ag e_nam e}}
  ```

- Create a release:

  ```
  octo create- release - - proj ect= {{proj ect_nam e}} - -
  pack ag ev ersion= {{v ersion}}
  ```

- Deploy a release:

  ```
  octo deploy- release - - proj ect= {{proj ect_nam e}} - -
  pack ag ev ersion= {{v ersion}} - - deployto= {{env ironm ent_nam e}} - -
  tenant= {{deploym ent_targ et}}
  ```

path

Display or set the search path for executable files.

More information: https://docs.microsoft.com/windows-server/administration/windows-commands/path.

- Display the current path:

```
path
```

- S et the path to one or more semicolon-separated directories:

```
path {{path/to/directory( s) }}
```

- Append a new directory to the original path:

```
path {{path/to/directory}}; % path%
```

- S et command prompt to only search the current directory for executables:

```
path ;
```

pathping

A trace route tool combining features of **`ping`** and **`tracert`**.

More information: https://docs.microsoft.com/windows-server/administration/windows-commands/pathping.

- Ping and trace the route to a host:

  ```
  pathping {{hostname}}
  ```

- Do not perform reverse lookup of ip-address to hostname:

  ```
  pathping {{hostname}} -n
  ```

- Specify the maximum number of hops to search for the target (the default is 30):

  ```
  pathping {{hostname}} -h {{max_hops}}
  ```

- Specify the milliseconds to wait between pings (the default is 240):

  ```
  pathping {{hostname}} -p {{time}}
  ```

- Specify the number of queries per hop (the default is 100):

  ```
  pathping {{hostname}} -q {{queries}}
  ```

- Force IPV4 usage:

  ```
  pathping {{hostname}} -4
  ```

- Force IPV6 usage:

  ```
  pathping {{hostname}} -6
  ```

- Display detailed usage information:

  ```
  pathping /?
  ```

popd

Changes the current directory to the directory stored by the `pu s h d` command.

More information: https://docs.microsoft.com/windows-server/administration/windows-commands/popd.

- Switch to directory at the top of the stack:

```
popd
```

powershell

Command-line shell and scripting language designed especially for system administration.

More information: https://docs.microsoft.com/windows-server/administration/windows-commands/powershell.

- Start a Windows PowerShell session in a Command Prompt window:

  ```
  powershell
  ```

- Load a specific PowerShell console file:

  ```
  powershell -PSConsoleFile {{path/to/file}}
  ```

- Start a session with a specified version of PowerShell:

  ```
  powershell -Version {{version}}
  ```

- Prevent the shell from exit after running startup commands:

  ```
  powershell -NoExit
  ```

- Describe the format of data sent to PowerShell:

  ```
  powershell -InputFormat {{Text|XML}}
  ```

- Determine how output from PowerShell is formatted:

  ```
  powershell -OutputFormat {{Text|XML}}
  ```

- Display help:

  ```
  powershell -Help
  ```

print

Print a text file to a printer.

More information: https://docs.microsoft.com/windows-server/administration/windows-commands/print.

- Print a text file to the default printer:

```
print {{path/to/file}}
```

- Print a text file to a specific printer:

```
print /d: {{printer}} {{path/to/file}}
```

psping

A ping tool that includes TCP ping, latency and bandwidth measurement.

More information: https://docs.microsoft.com/sysinternals/downloads/psping.

- Ping a host using ICMP:

```
psping {{hostname}}
```

- Ping a host over a TCP port:

```
psping {{hostname}}:{{port}}
```

- Specify the number of pings and perform it quietly:

```
psping {{hostname}} -n {{pings}} -q
```

- Ping the target over TCP 50 times and produce a histogram of the results:

```
psping {{hostname}}:{{port}} -q -n {{50}} -h
```

- Display usage information:

```
psping /?
```

pushd

Place a directory on a stack so it can be accessed later.

See also **popd** to switch back to original directory.

More information: https://docs.microsoft.com/windows-server/administration/windows-commands/pushd.

- Switch to directory and push it on the stack:

```
pushd {{directory}}
```

pwlauncher

A command-line tool for managing the Windows T o G o startup options.

More information: https://docs.microsoft.com/windows-server/administration/windows-commands/pwlauncher.

- Display the current Windows T o G o status:

  ```
  pw lau ncher
  ```

- E nable or disable the Windows T o G o startup options:

  ```
  pw lau ncher /{{enab le| disab le}}
  ```

rdpsign

A tool for signing Remote Desktop Protocol (RDP) files.

More information: https://docs.microsoft.com/windows-server/administration/windows-commands/rdpsign.

- Sign an RDP file:

```
rdpsign {{path/to/file.rdp}}
```

- Sign an RDP file using a specific sha256 hash:

```
rdpsign {{path/to/file.rdp}} /sha256 {{hash}}
```

- Enable quiet output:

```
rdpsign {{path/to/file.rdp}} /q
```

- Display verbose warnings, messages and statuses:

```
rdpsign {{path/to/file.rdp}} /v
```

- Test the signing by displaying the output to stdout without updating the file:

```
rdpsign {{path/to/file.rdp}} /l
```

reg add

Add new keys and their values to the registry.

More information: https://docs.microsoft.com/windows-server/administration/windows-commands/reg-add.

- Add a new registry key:

```
reg add {{key_name}}
```

- Add a new value under a specific key:

```
reg add {{key_name}} /v {{value}}
```

- Add a new value with specific data:

```
reg add {{key_name}} /d {{data}}
```

- Add a new value to a key with a specific data type:

```
reg add {{key_name}} /t {{type}}
```

- Forcefully overwrite the existing registry value without a prompt:

```
reg add {{key_name}} /f
```

reg compare

Compare keys and their values in the registry.

More information: https://docs.microsoft.com/windows-server/administration/windows-commands/reg-compare.

- Compare all values under a specific key with a second key:

```
reg compare {{first_key_name}} {{second_key_name}}
```

- Compare a specific value under two keys:

```
reg compare {{first_key_name}} {{second_key_name}} /v {{value}}
```

- Compare all sub keys and values for two keys:

```
reg compare {{first_key_name}} {{second_key_name}} /s
```

- Only output the matches between the specified keys:

```
reg compare {{first_key_name}} {{second_key_name}} /os
```

- Output the differences and matches between the specified keys:

```
reg compare {{first_key_name}} {{second_key_name}} /oa
```

reg copy

Copy keys and their values in the registry.

More information: https://docs.microsoft.com/windows-server/administration/windows-commands/reg-copy.

- Copy a registry key to a new registry location:

```
reg copy {{old_key_name}} {{new_key_name}}
```

- Copy a registry key recursively to a new registry location:

```
reg copy {{old_key_name}} {{new_key_name}} /s
```

- Forcefully copy a registry key without a prompt:

```
reg copy {{old_key_name}} {{new_key_name}} /f
```

reg delete

Delete keys or their values from the registry.

More information: https://docs.microsoft.com/windows-server/administration/windows-commands/reg-delete.

- Delete a specific registry key:

```
reg delete {{key_name}}
```

- Delete a value under a specific key:

```
reg delete {{key_name}} /v {{value}}
```

- Delete all values recursively under the specified key:

```
reg delete {{key_name}} /va
```

- Forcefully delete all values recursively under a key without a prompt:

```
reg delete {{key_name}} /f /va
```

reg export

Export the specified sub keys and values into a file.

More information: https://docs.microsoft.com/windows-server/administration/windows-commands/reg-export.

- Export all sub keys and values of a specific key:

```
reg export {{key_name}} {{path/to/file.reg}}
```

- Force overwriting of an existing file without prompt:

```
reg export {{key_name}} {{path/to/file.reg}} /y
```

reg flags

Display or set flags on registry keys.

More information: https://docs.microsoft.com/windows-server/administration/windows-commands/reg-flags

- Display current flags for a specific key:

```
reg flags {{key_name}} query
```

- Display help and available flag types:

```
reg flags /?
```

- Set specified space-separated flags, and unset unmentioned flags, for a specific key:

```
reg flags {{key_name}} set {{flags}}
```

- Set specified flags for a specific key and its sub keys:

```
reg flags {{key_name}} set {{flags}} /s
```

reg import

Import all available keys, subkeys, and values from a file.

More information: https://docs.microsoft.com/windows-server/administration/windows-commands/reg-import.

- Import all keys, subkeys and values from a file:

```
reg  im port {{path/to/file. reg }}
```

reg load

Load saved sub keys into a different sub key in the registry.

T his is intended for troubleshooting and temporary keys.

More information: https://docs.microsoft.com/windows-server/administration/windows-commands/reg-load.

- Load a backup file into the specified key:

```
reg  load {{k ey_nam e}} {{path/to/file}}
```

reg q uery

Display the values of keys and sub keys in the registry.

More information: https://docs.microsoft.com/windows-server/administration/windows-commands/reg-q uery

- Display all values of a key:

```
reg q u ery {{k ey_nam e}}
```

- Display a specific value of a key:

```
reg q u ery {{k ey_nam e}} /v {{v alu e}}
```

- Display all values of a key and its sub keys:

```
reg q u ery {{k ey_nam e}} /s
```

- S earch for keys and values matching a specific pattern:

```
reg q u ery {{k ey_nam e}} /f " {{q u ery_pattern}}"
```

- Display a value of a key matching a specified data type:

```
reg q u ery {{k ey_nam e}} /t {{type}}
```

reg restore

Restore a key and its values from a backup file.

See `reg save` for more information.

More information: https://docs.microsoft.com/windows-server/administration/windows-commands/reg-restore.

- Overwrite a specified key with data from a backup file:

```
reg restore {{key_name}} {{path/to/file}}
```

reg save

Save a registry key, its sub keys and values to a file.

More information: https://docs.microsoft.com/windows-server/administration/windows-commands/reg-save.

- Save a registry key, its sub keys and values to a specific file:

```
reg save {{key_name}} {{path/to/file}}
```

- Forcefully overwrite an existing file without a prompt:

```
reg save {{key_name}} {{path/to/file}} /y
```

reg unload

Remove data from the registry that was loaded using the `reg load` command.

More information: https://docs.microsoft.com/windows-server/administration/windows-commands/reg-unload.

- Remove data from the registry for a specified key:

```
reg unload {{key_name}}
```

reg

A command-line interface for managing keys and their values in the Windows registry.

See `reg query`, `reg add` and other pages for additional information.

More information: https://docs.microsoft.com/windows-server/administration/windows-commands/reg.

- Execute registry commands:

  ```
  reg {{command}}
  ```

- Display general information and list all available commands:

  ```
  reg /?
  ```

- Call help on a specific command:

  ```
  reg {{command}} /?
  ```

repair-bde

Attempt to repair or decrypt a damaged B itLocker-encrypted volume.

More information: https://docs.microsoft.com/windows-server/administration/windows-commands/repair-bde.

- Attempt to repair a specified volume:

```
repair- b de {{C : }}
```

- Attempt to repair a specified volume and output to another volume:

```
repair- b de {{C : }} {{D : }}
```

- Attempt to repair a specified volume using the provided recovery key file:

```
repair- b de {{C : }} - R ecov eryK ey {{path/to/file. b ek }}
```

- Attempt to repair a specified volume using the provided numerical recovery password:

```
repair- b de {{C : }} - R ecov eryP assw ord {{passw ord}}
```

- Attempt to repair a specified volume using the provided password:

```
repair- b de {{C : }} - P assw ord {{passw ord}}
```

- Attempt to repair a specified volume using the provided key package:

```
repair- b de {{C : }} - K eyP ack ag e {{path/to/directory}}
```

- Log all output to a specific file:

```
repair- b de {{C : }} - L og F ile {{path/to/file}}
```

- Display all available options:

```
repair- b de /?
```

replace

Replace files.

See also: **`robocopy`**, **`move`**, **`copy`**, and **`del`**.

More information: https://docs.microsoft.com/windows-server/administration/windows-commands/replace.

- Replace the destination file with the one from the source directory:

  ```
  replace {{path/to/file_or_directory}} {{path/to/destination}}
  ```

- Add files to the destination directory instead of replacing existing files:

  ```
  replace {{path/to/file_or_directory}} {{path/to/destination}} /a
  ```

- Interactively copy multiple files, with a prompt before replacing or adding a destination file:

  ```
  replace {{path/to/file_or_directory}} {{path/to/destination}} /p
  ```

- Replace even read only files:

  ```
  replace {{path/to/file_or_directory}} {{path/to/destination}} /r
  ```

- Wait for you to insert a disk before it replaces files (originally to allow inserting a floppy disk):

  ```
  replace {{path/to/file_or_directory}} {{path/to/destination}} /w
  ```

- Replace all files in subdirectories of the destination:

  ```
  replace {{path/to/file_or_directory}} {{path/to/destination}} /s
  ```

- Replace only files in the destination directory which are older than the files in the source directory:

  ```
  replace {{path/to/file_or_directory}} {{path/to/destination}} /u
  ```

- Display detailed usage information:

  ```
  replace /?
  ```

rmdir

Remove a directory and its contents.

More information: https://docs.microsoft.com/windows-server/administration/windows-commands/rmdir.

- Remove an empty directory:

```
rmdir {{path/to/directory}}
```

- Remove a directory and its contents recursively:

```
rmdir {{path/to/directory}} /s
```

- Remove a directory and its contents recursively without prompting:

```
rmdir {{path/to/directory}} /s /q
```

robocopy

Robust File and Folder Copy.

By default files will only be copied if the source and destination have different time stamps or different file sizes.

More information: https://docs.microsoft.com/windows-server/administration/windows-commands/robocopy.

- Copy all `.jpg` and `.bmp` files from one directory to another:

```
robocopy {{path/to/source}} {{path/to/destination}} {{*.jpg}} {{*.bmp}}
```

- Copy all files and subdirectories, including empty ones:

```
robocopy {{path/to/source}} {{path/to/destination}} /E
```

- Mirror/Sync a directory, deleting anything not in source and include all attributes and permissions:

```
robocopy {{path/to/source}} {{path/to/destination}} /MIR /COPYALL
```

- Copy all files and subdirectories, excluding source files that are older than destination files:

```
robocopy {{path/to/source}} {{path/to/destination}} /E /XO
```

- List all files 50 MBytes or larger in size instead of copying them:

```
robocopy {{path/to/source}} {{path/to/destination}} /MIN:52428800 /L
```

- Allow resuming if network connection is lost and limit retries to 5 and wait time to 15 sec:

```
robocopy {{path/to/source}} {{path/to/destination}} /Z /R:5 /W:15
```

- Display detailed usage information:

```
robocopy /?
```

rpcinfo

List programs via RPC on remote computers.

More information: https://docs.microsoft.com/windows-server/administration/windows-commands/rpcinfo.

- List all programs registered on the local computer:

```
rpcinfo
```

- List all programs registered on a remote computer:

```
rpcinfo /p {{computer_name}}
```

- Call a specific program on a remote computer using TCP:

```
rpcinfo /t {{computer_name}} {{program_name}}
```

- Call a specific program on a remote computer using UDP:

```
rpcinfo /u {{computer_name}} {{program_name}}
```

scoop bucket

Manage buckets: Git repositories containing files which describe how scoop installs applications.

If Scoop doesn't know where the bucket is located its repository location must be specified.

More information: https://github.com/lukesampson/scoop/wiki/Buckets

- List all buckets currently in use:

  ```
  scoop bucket list
  ```

- List all known buckets:

  ```
  scoop bucket known
  ```

- Add a known bucket by its name:

  ```
  scoop bucket add {{name}}
  ```

- Add an unknown bucket by its name and Git repository URL:

  ```
  scoop bucket add {{name}} {{https://example.com/repository.git}}
  ```

- Remove a bucket by its name:

  ```
  scoop bucket rm {{name}}
  ```

scoop

A command-line installer for Windows.

More information: https://scoop.sh.

- Install a package:

  ```
  scoop install {{package}}
  ```

- Remove a package:

  ```
  scoop uninstall {{package}}
  ```

- Update all installed packages:

  ```
  scoop update *
  ```

- List installed packages:

  ```
  scoop list
  ```

- Display information about a package:

  ```
  scoop info {{package}}
  ```

- Search for a package:

  ```
  scoop search {{package}}
  ```

- Remove old versions of all packages and clear the download cache:

  ```
  scoop cleanup -k *
  ```

set

Display or set environment variables for the current instance of CMD.

More information: https://docs.microsoft.com/windows-server/administration/windows-commands/set.

- List all current environment variables:

```
set
```

- S et an environment variable to a specific value:

```
set {{nam e}}= {{v alu e}}
```

- List environment variables starting with the specified string:

```
set {{nam e}}
```

- P rompt the user for a value for the specified variable:

```
set /p {{nam e}}= {{prom pt_string }}
```

sfc

S cans the integrity of Windows system files.

More information: https://docs.microsoft.com/windows-server/administration/windows-commands/sfc.

- Display information about the usage of the command:

  ```
  sfc
  ```

- S can all system files and, if possible, repair any problems:

  ```
  sfc /scannow
  ```

- S can all system files without attempting to repair any:

  ```
  sfc /v erifyonly
  ```

- S can a specific file and, if possible, repair any problems:

  ```
  sfc /scanfile= {{path/to/file}}
  ```

- S can a specific file without attempting to repair it:

  ```
  sfc /v erifyfile= {{path/to/file}}
  ```

- When repairing offl ine, specify the boot directory:

  ```
  sfc /offb ootdir= {{path/to/directory}}
  ```

- When repairing offl ine, specify the Windows directory:

  ```
  sfc /offw indir= {{path/to/directory}}
  ```

showmount

Display information about NFS filesystems on Windows Server.

More information: https://docs.microsoft.com/windows-server/administration/windows-commands/showmount.

- Display all exported filesystems:

```
showmount -e
```

- Display all NFS clients and their mounted directories:

```
showmount -a
```

- Display all NFS mounted directories:

```
showmount -d
```

- Display all exported filesystems for a remote server:

```
showmount -e {{server_address}}
```

shutdown

A tool for shutting down, restarting or logging off a machine.

More information: https://docs.microsoft.com/windows-server/administration/windows-commands/shutdown.

- S hutdown the current machine:

  ```
  shu tdow n /s
  ```

- S hutdown the current machine force-closing all apps:

  ```
  shu tdow n /s /f
  ```

- R estart the current machine immediately:

  ```
  shu tdow n /r /t 0
  ```

- H ibernate the current machine:

  ```
  shu tdow n /h
  ```

- Log off the current machine:

  ```
  shu tdow n /l
  ```

- S pecify a timeout in seconds to wait before shutting down:

  ```
  shu tdow n /s /t {{seconds}}
  ```

- Abort a shutdown seq uence whose timeout is yet to expire:

  ```
  shu tdow n /a
  ```

- S hutdown a remote machine:

  ```
  shu tdow n /m  {{\ \ hostnam e}}
  ```

subst

Associates a path with a virtual drive letter.

More information: https://docs.microsoft.com/windows-server/administration/windows-commands/subst.

- List active associations:

```
su b st
```

- Add an association:

```
su b st {{Z : }} {{C : \ P ython2 . 7 }}
```

- R emove an association:

```
su b st {{Z : }} /d
```

systeminfo

Display operating system configuration for a local or remote machine.

More information: https://docs.microsoft.com/windows-server/administration/windows-commands/systeminfo.

- Display system configuration for the local machine:

  ```
  system info
  ```

- Display system configuration in a specified output format:

  ```
  system info /fo {{tab le| list| csv }}
  ```

- Display system configuration for a remote machine:

  ```
  system info /s {{rem ote_nam e}} /u  {{u sernam e}} /p {{passw ord}}
  ```

- Display detailed usage information:

  ```
  system info /?
  ```

takeown

Take ownership of a file or directory.

More information: https://docs.microsoft.com/windows-server/administration/windows-commands/takeown.

- Take ownership of the specified file:

```
takeown /f {{path/to/file}}
```

- Take ownership of the specified directory:

```
takeown /d {{path/to/directory}}
```

- Take ownership of the specified directory and all subdirectories:

```
takeown /r /d {{path/to/directory}}
```

- Change ownership to the Administrator group instead of the current user:

```
takeown /a /f {{path/to/file}}
```

taskkill

Terminate a process by its process id or name.

More information: https://docs.microsoft.com/windows-server/administration/windows-commands/taskkill.

- Terminate a process by its id:

```
taskkill /pid {{process_id}}
```

- Terminate a process by its name:

```
taskkill /im {{process_name}}
```

- Forcefully terminate a specified process:

```
taskkill /pid {{process_id}} /f
```

- Terminate a process and its child processes:

```
taskkill /im {{process_name}} /t
```

- Terminate a process on a remote machine:

```
taskkill /pid {{process_id}} /s {{remote_name}}
```

- Display information about the usage of the command:

```
taskkill /?
```

tasklist

Display a list of currently running processes on a local or remote machine.

More information: https://docs.microsoft.com/windows-server/administration/windows-commands/tasklist.

- Display currently running processes:

  ```
  task list
  ```

- Display running processes in a specified output format:

  ```
  task list /fo {{table|list|csv}}
  ```

- Display running processes using the specified `.exe` or `.dll` file name:

  ```
  task list /m {{module_pattern}}
  ```

- Display processes running on a remote machine:

  ```
  task list /s {{remote_name}} /u {{username}} /p {{password}}
  ```

- Display services using each process:

  ```
  task list /svc
  ```

title

Set the title of the command prompt window.

More information: https://docs.microsoft.com/windows-server/administration/windows-commands/title.

- Set the title of the current command prompt window:

```
title {{new_title}}
```

tree

Display a graphical tree of the directory structure for a path.

More information: https://docs.microsoft.com/windows-server/administration/windows-commands/tree.

- Display the tree for the current directory:

  ```
  tree
  ```

- Display the tree for a specific directory:

  ```
  tree {{path/to/directory}}
  ```

- Display the tree for a directory including files:

  ```
  tree {{path/to/directory}} /f
  ```

- Display the tree using AS CII characters instead of extended characters:

  ```
  tree {{path/to/directory}} /a
  ```

tskill

Ends a process running in a session on a Remote Desktop Session Host.

More information: https://docs.microsoft.com/windows-server/administration/windows-commands/tskill.

- Terminate a process by its process identifier:

  ```
  tskill {{process_id}}
  ```

- Terminate a process by its name:

  ```
  tskill {{process_name}}
  ```

type

Display the contents of a file.

More information: https://docs.microsoft.com/windows-server/administration/windows-commands/type.

- Display the contents of a specific file:

```
type {{path/to/file}}
```

tz_util

A tool for displaying or configuring the system time zone.

More information: https://docs.microsoft.com/windows-server/administration/windows-commands/tz_util

- Get the current time zone:

  ```
  tzutil /g
  ```

- Display a list of available time zones:

  ```
  tzutil /l
  ```

- Set the system time zone to the specific value:

  ```
  tzutil /s {{timezone_id}}
  ```

ver

Display the current Windows or MS-DOS version number.

More information: https://docs.microsoft.com/windows-server/administration/windows-commands/ver.

- Display the current version number:

```
ver
```

virtualboxvm

The VirtualBox virtual machine management CLI.

More information: https://www.virtualbox.org.

- Start a virtual machine:

```
virtualboxvm --startvm {{name|uuid}}
```

- Start a virtual machine in fullscreen mode:

```
virtualboxvm --startvm {{name|uuid}} --fullscreen
```

- Mount the specified DVD image file:

```
virtualboxvm --startvm {{name|uuid}} --dvd {{path/to/image_file}}
```

- Display a command-line window with debug information:

```
virtualboxvm --startvm {{name|uuid}} --debug-command-li
```

- Start a virtual machine in a paused state:

```
virtualboxvm --startvm {{name|uuid}} --start-paused
```

vol

Display information about volumes.

More information: https://docs.microsoft.com/windows-server/administration/windows-commands/vol.

- Display the label and serial number for the current drive:

```
v ol
```

- Display the label and serial number for a specific volume:

```
v ol {{D : }}
```

where

Display the location of files that match the search pattern.

Defaults to current work directory and paths in the PATH environment variable.

More information: https://docs.microsoft.com/windows-server/administration/windows-commands/where.

- Display the location of file pattern:

```
where {{file_pattern}}
```

- Display the location of file pattern including file size and date:

```
where /T {{file_pattern}}
```

- Recursively search for file pattern at specified path:

```
where /R {{path/to/directory}} {{file_pattern}}
```

- Display only the error code for the location of file pattern:

```
where /Q {{file_pattern}}
```

whoami

Display details about the current user.

More information: https://docs.microsoft.com/windows-server/administration/windows-commands/whoami.

- Display the username of the current user:

```
w hoam i
```

- Display the groups that the current user is a member of:

```
w hoam i /g rou ps
```

- Display the privileges of the current user:

```
w hoam i /priv
```

- Display the user principal name (U P N) of the current user:

```
w hoam i /u pn
```

- Display the logon id of the current user:

```
w hoam i /log onid
```

winget

Windows Package Manager CLI.

More information: https://docs.microsoft.com/windows/package-manager/winget.

- Install a package:

  ```
  winget install {{package}}
  ```

- Display information about a package:

  ```
  winget show {{package}}
  ```

- Search for a package:

  ```
  winget search {{package}}
  ```

wmic

Interactive shell for detailed information about running processes.

More information: https://docs.microsoft.com/windows-server/administration/windows-commands/wmic.

- Fundamental grammar:

```
wmic {{alias}} {{where_clause}} {{verb_clause}}
```

- Show brief details about the currently running processes:

```
wmic process list brief
```

- Show full details about the currently running processes:

```
wmic process list full
```

- Access specific fields such as process name, process ID and parent process ID:

```
wmic process get {{name,processid,parentprocessid}}
```

- Display information about a specific process:

```
wmic process where {{name="example.exe"}} list full
```

- Display specific fields for a specific process:

```
wmic process where processid={{pid}} get {{name,commandline}}
```

- Kill a process:

```
wmic process {{pid}} delete
```

wsl

Manage the Windows Subsystem for Linux from the command-line.

More information: https://docs.microsoft.com/windows/wsl/reference.

- Start a Linux shell (in the default distribution):

```
wsl {{shell_command}}
```

- Run a Linux command without using a shell:

```
wsl --exec {{command}} {{command_arguments}}
```

- Specify a particular distribution:

```
wsl --distribution {{distribution}} {{shell_command}}
```

- List available distributions:

```
wsl --list
```

- Export a distribution to a `.tar` file:

```
wsl --export {{distribution}} {{path/to/distro_fs.tar}}
```

- Import a distribution from a `.tar` file:

```
wsl --import {{distribution}} {{path/to/install_location}}
{{path/to/distro_fs.tar}}
```

- Change the version of the specified distribution:

```
wsl --set-version {{distribution}} {{version}}
```

- Shut down Windows Subsystem for Linux:

```
wsl --shutdown
```

xcopy

Copy files and directory trees.

More information: https://docs.microsoft.com/windows-server/administration/windows-commands/xcopy.

- Copy the file(s) to the specified destination:

```
xcopy {{path/to/file_or_directory}} {{path/to/destination}}
```

- List files that will be copied before copying:

```
xcopy {{path/to/file_or_directory}} {{path/to/destination}} /p
```

- Copy the directory structure only, excluding files:

```
xcopy {{path/to/file_or_directory}} {{path/to/destination}} /t
```

- Include empty directories when copying:

```
xcopy {{path/to/file_or_directory}} {{path/to/destination}} /e
```

- Keep the source ACL in the destination:

```
xcopy {{path/to/file_or_directory}} {{path/to/destination}} /o
```

- Allow resuming when network connection is lost:

```
xcopy {{path/to/file_or_directory}} {{path/to/destination}} /z
```

- Disable the prompt when the file exists in the destination:

```
xcopy {{path/to/file_or_directory}} {{path/to/destination}} /y
```

- Display detailed usage information:

```
xcopy /?
```