Robot Operating System (ROS) Command-line Tools Cheat Sheet

roscd

`roscd <package>`	go to package dir
`roscd log`	go to log dir

Changes directory to the path of the specified package, stack, or special location.

rospd

`rospd <package>`	go to package dir

Changes directory to the path of the specified package, and ads the path to the directory stack.

rosd

`rosd <id>`	go to id

Changes directory to the package on the directory stack that has the specified `id`.

rosls

```
rosls <package>
```

Changes directory to the patch of the specified package.

rosed

```
rosed <package> <file>
```

Edits specified file in package, regardless of depth.

roscp

```
roscp <package> <file> <destination>
```

Copies the specified file from its package to the destination.

roscore

The core is the central manager. It is responsible for the name service, `rosout` and the parameter server.

`--help, -h`	show help
`--port=, -p`	master port
`-v`	verbose

rosnode

A node is an executable that can communicate with other nodes via ROS.

`<command> --help, -h`	help on command
`ping <node>`	test connectivity
`-all, -a`	ping all nodes
`-c <count>`	number of pings
`list`	list active nodes
`-all, -a`	all information
`-u`	XML-RPC URIs
`info <node1> ...`	info about node(s)
`machine <machine>`	list machine's nodes
`kill <node>`	kills node
`-all, -a`	kill all nodes
`cleanup <node>`	deregisters offline nodes

rostopic

Topics are conduits for messages. Nodes can publish to and subscribe to topics.

`<command> --help, -h`	help on command
`bw <topic>`	bandwidth used
`--window, -w`	window size
`echo <topic>`	real-time messages
`--bag, b <file>`	from `.bag` file
`-p`	plotting-friendly format
`-w <width>`	numeric fixed width
`--filter=<expr>`	filter by py expression
`--nostr`	no string fields
`--noarr`	no arrays
`--clear, -c`	clear screen before next
`--all, -a`	all in `.bag` (with `-bag`)
`-n <count>`	number of messages
`--offset`	times relative to now
`find <msg-type>`	find topics
`hz <topic>`	publishing rate
`--window, -w`	window size
`--filter=<expr>`	filter by py expression
`info <topic>`	info about topic
`list`	list topics
`--rate, r <hz>`	publishing rate
`--once, -1`	publish once
`--file, -f <file>`	yaml args from file
`--latch, -l`	latch first (`-f, -r`)
`--verbose, -v`	full details
`pub <topic> <data>`	publish to topic
`--bag, b <file>`	from `.bag` file
`-p`	only publishers
`-s`	only subscribers
`--host`	group by hostname
`--verbose, -v`	full details
`type <topic>`	get topic's type

rosmsg, rossrv

`rosmsg` shows information about message types. `rossrv` shows information about service types. Use `rostopic` to publish to topics, and `rosservice` to call services.

`<command> --help, -h`	help on command
`show <msg-type>`	about this type
`--raw, -r`	raw text and comments
`--bag=, -b <file>`	show from `.bag` file
`list`	list all messages
`md5`	message hash
`package <package>`	list messages in package
`-s`	on single line
`packages`	packages with messages
`-s`	on single line

rosrun

```
rosrun <package> <executable> [args]
```

Runs the specified executable in the specified package.

`--debug`	runs in debug mode

rosdep

Manages package dependencies.

`<command> --help, -h`	help on command
`check <packages>`	checks dependencies
`install <packages>`	generates and runs script
`db`	create/show database
`init`	inits`/etc/ros/rosdep`
`keys <packages>`	keys packages depend on
`resolve <ros-deps>`	resolves to system dep's
`update`	updates the local db (don't use `sudo`)
`what-needs <ros-deps>`	packages that need it
`where-defined <deps>`	yaml that requires it
`fix-permissions`	you used `sudo update`

rosconsole

Configures logger level of nodes. Level is one of `debug|info|warn|error|fatal`.

`get <node> <log>`	get level for node/logger
`set <node> <log> <lv>`	set level for node/logger
`list <node>`	logger/levels for node

rospack

Manages and provides information about packages. `<pkg>` can be omitted if the current directory contains a manifest.

`-q`	quiets error reports
`<command> --help, -h`	help on command
`cflags-only-I <pkg>`	list exports from manifest
`--deps-only`	exclude package itself
`cflags-only-other <pkg>`	exports without `-I`
`--deps-only`	exclude package itself
`deps <pkg>`	dependencies
`deps-indent <pkg>`	entire dependency chain
`deps-manifests <pkg>`	manifests
`deps-msgsrv <pkg>`	message-gen marker files
`depends-on <pkg>`	pkgs that depend on this
`depends-on1 <pkg>`	directly depend on this
`deps-why <pkg>`	dependencies from
`--target=, -t <pkg>`	package to target
`deps1 <pkg>`	packages this depends on
`exports <pkg>`	list exports
`--deps-only`	exclude package
`--lang=<lang>`	specific language
`--attrib=<attr>`	attribute
`find <pkg>`	absolute path to package
`langs`	language-specific libraries
`libs-only-L <pkg>`	list exports with `-L`
`--deps-only`	exclude package itself
`libs-only-l <pkg>`	list exports with `-l`
`--deps-only`	exclude package itself
`libs-only-other <pkg>`	exports without `-L`, `-l`
`--deps-only`	exclude package itself
`list`	all packages and paths
`list-duplicates`	list duplicated packages
`list-names`	list packages, names only
`plugins <pkg>`	packages that depend on
`--attrib=<attr>`	this with attribute
`--top=<pkg>`	also depending on this
`profile`	crawl all and report time
`--length=<length>`	number to display
`--zombie-only`	those missing manifests
`rosdep <pkg>`	`rosdeps` from manifests
`rosdep0 <pkg>`	from *this* manifest
`vcs <pkg>`	`versioncontrol` from manifests
`vcs0 <pkg>`	from *this* manifest

catkin_test_results

```
catkin_test_results <results-dir>
```

Outputs the summary of test results.

`--all`	show good, bad, ugly
`--verbose`	show all erred/failed

rosmake

```
rosmake <package>
```

Builds a package and its dependencies.

`--test-only`	only run tests
`-t`	build and test packages
`--mark-installed, -i`	mark with ROS_NOBUILD
`--unmark-installed, -u`	unmark as built
`-v`	display erred builds
`-V`	display all builds
`--robust, -r, -k`	don't stop on errors
`--build-everything`	build all and don't stop
`--specified-only, -s`	only build package x
`--buildtest=<test>`	package to buildtest
`--buildtest1=<test>`	package to buildtest1
`--output=<dir>`	output to directory
`--pre-clean`	`make clean` first
`--bootstrap`	do bootstrap packages
`--disable-logging`	disable logging
`--target=<target>`	make with this target
`--pjobs=<count>`	override ROS_PARALLEL_JOBS
`--threads=<count>`	build n packages async
`--profile`	show time profile at end
`--skip-blacklist`	skip packages in blacklist
`--require-platform`	only if platform supported
`--require-platform-recursive`	only if dependencies supported
`--status-rate=<rate>`	status bar update freq.

rosservice

Manage and query a service. Services can be called with yaml syntax.

`call <svc> <args>`	calls service with args
`--wait`	first wait for service
`find <svc-type>`	show all services of type
`info <svc>`	help on command
`list`	list all available services
`-n`	include node name
`node <svc>`	get node with service
`type <svc>`	show services type
`uri <svc>`	show service's URI
`args <svc>`	get available arguments

rosunit

```
rosunit <file> <test-args>
```

Runs a unit test.

`--time-limit=<time>`	time limit for test
`--name=<test>`	name of unit test
`--package=<package>`	optional name of package

roslaunch

```
roslaunch <package> <launch-file> [args]
roslaunch <launch-file1> ... [args]
roslaunch - [args]
```

Launches multiple nodes using XML launch files, locally and via ssh. Can set parameters on server.

`-p <port>`	roscore's port
`--wait`	wait for roscore
`--local`	only local nodes
`--screen`	force output to screen
`-v`	verbose
`--dump-params`	show launch file params
`--nodes`	names in launch file
`--args <node>`	get args used for node
`--find <node>`	get launch file for node
`--files <file>`	files used to process file
`-p <port>`	roscore's port

rosparam

Parameters are stored on the parameter server (managed by `roscore`) as YAML-encoded files. All commands offer a `-v` (verbose output) option.

`<command> --help, -h`	help on command
`set <param> <value>`	sets a parameter
`--textfile, -t <fn>`	set to text file contents
`--binfile, -b <fn>`	set to binary file contents
`get <param>`	gets a parameter
`-p`	pretty output (not YAML)
`load <file> <ns>`	loads from YAML file, optionally into namespace
`dump <file> <ns>`	dumps all to file, optionally filtered by NS
`delete <param>`	sets a parameter
`list <namespace>`	list parameters, optionally in namespace

rosclean

Cleans log files.

`check`	check disk usage
`purge`	remove logs

roswtf

Performs a sanity check on the current installation.

`--all`	run against all packages
`--no-plugins`	disables plugins
`--offline`	only runs offline tests

rosbag

Bags are recordings of messages to topics that can be played back, like macros.

`<command> --help, -h`	help on command
`record <topics>`	records topics to bag
`--all, -a`	record call topics to bag
`--regex=, -e <exp>`	record matching topics
`--exclude, -x <exp>`	exclude matching topics
`--quiet`	suppress output
`--output-prefix, -o <pre>`	prepend bag name
`--output-name, -O <name>`	record to NAME.bag
`--split`	split bag when
`--size=, -b <MB>`	size reached (MB)
`--duration= <t>`	period lapsed
`--buffsize=<MB>`	internal buffer size
`--limit=, -l <n>`	limit to n messages
`--bz2, -j`	use BZ2 compression
`info <bag1> ...`	get bag info
`--yaml, -y`	output in YAML
`--key=, -k <key>`	only field (with `-y`)
`play <bag1> ...`	playback bag
`--quiet, -q`	suppress output
`--immediate, -i`	play all without waiting
`--pause`	start paused
`--queue=<size>`	outgoing queue size
`--clock`	publish clock time
`--hz=<hz>`	publish clock frequency
`--delay=, -d <sec>`	sleep after advertise
`--rate=, -r <factor>`	multiply publish rate
`--start=, -s <sec>`	start into bag by offset
`--duration=, -u <sec>`	play only this much
`--loop, -l`	loop playback
`--keep-alive, -k`	keep alive at end of bag
`check`	can system play bag?
`--genrules=, -g <f>`	generate migration rule
`--append, -a`	append to migration rule
`--noplugins, -n`	don't load plugins rules
`fix <in> <out> <rules>`	fix a bag, rules optional
`--noplugins, -n`	don't load plugins rules
`filter <in> <out> <filter>`	filter by py expr.
`--print=<expr>`	evaluate/print expression
`compress <bag1> ...`	compress bag
`--output-dir=<dir>`	output to dir
`--force, -f`	overwrite if exists
`--quiet`	be less verbose
`decompress <bag1> ...`	decompress bag
`--output-dir=<dir>`	output to dir
`--force, -f`	overwrite if exists
`--quiet`	be less verbose
`reindex <bag1> ...`	repair broken bag
`--output-dir=<dir>`	output to dir
`--force, -f`	overwrite if exists
`--quiet`	be less verbose

catkin_init_workspace

Initializes a catkin workspace by creating top-level `CMakeLists.txt`.

catkin_make

`catkin_make`	builds all
`catkin_make <package>`	builds package

Builds all projects in the workspace. Must be called from top-level in workspace. Additional commands can be appended e.g. `catkin_make install`.

catkin_find†

```
catkin_find <package> <path>
```

Searches catkin workspaces for project-specific files/folders.

catkin_package_version

```
catkin_package_version <path>
```

Shows or bumps the version number in `package.xml` files.

`--bump {major|minor|patch}`	part to bump

catkin_prepare_release

Bumps the version number, commits the modified package.xml, and creates a tag in the repository.

`--no-push`	does not push to repo
`--tag-prefix, -t <pre>`	prefix for release tag
`--bump {major|minor|patch}`	part to bump
`--non-interactive, -y`	answer yes to all
`--no-color`	disables colour output

† Some options have been excluded.

Red Cell Innovation Inc. **L'Innovation de Globules Rouges** Inc.

Version 1.0 – July 29, 2015
https://github.com/TwoRedCells/ros-cheat-sheets

All information sourced from http://wiki.ros.org
Please send corrections to the author.