

ROS Indigo Cheatsheet

Filesystem Management Tools

rospack	A tool for inspecting packages .
rospack profile	Fixes path and pluginlib problems.
roscd	Change directory to a package.
rospd/rostd	Pushd equivalent for ROS .
rosls	Lists package or stack information.
rosed	Open requested ROS file in a text editor.
roscp	Copy a file from one place to another.
rosdep	Installs package system dependencies.
roswtf	Displays a errors and warnings about a running ROS system or launch file.
catkin_create_pkg	Creates a new ROS stack.
wstool	Manage many repos in workspace.
catkin_make	Builds a ROS catkin workspace.
rqt_dep	Displays package structure and dependencies.

Usage:

```
$ rospack find [package]
$ roscd [package[/subdir]]
$ rospd [package[/subdir] | +N | -N]
$ rosd
$ rosls [package[/subdir]]
$ rosed [package] [file]
$ roscp [package] [file] [destination]
$ rosdep install [package]
$ roswtf or roswtf [file]
$ catkin_create_pkg [package_name] [depend1]..[dependN]
$ wstool [init | set | update]
$ catkin_make
$ rqt_dep [options]
```

Start-up and Process Launch Tools

roscore

The basis [nodes](#) and programs for ROS-based systems. A roscore must be running for ROS nodes to communicate.

Usage:

```
$ roscore
```

roslaunch

Runs a ROS package's executable with minimal typing.

Usage:

```
$ roslaunch package_name executable_name
```

Example (runs [turtlesim](#)):

```
$ roslaunch turtlesim turtlesim_node
```

roslaunch

Starts a roscore (if needed), [local nodes](#), [remote nodes](#) via SSH, and sets parameter server [parameters](#).

Examples:

```
Launch a file in a package:
$ roslaunch package_name file_name.launch
Launch on a different port:
$ roslaunch -p 1234 package_name file_name.launch
Launch on the local nodes:
$ roslaunch --local package_name file_name.launch
```

Logging Tools

rosvbag

A set of tools for recording and playing back of ROS topics.

Commands:

rosvbag record	Record a bag file with specified topics.
rosvbag play	Play content of one or more bag files.
rosvbag compress	Compress one or more bag files.
rosvbag decompress	Decompress one or more bag files.
rosvbag filter	Filter the contents of the bag.

Examples:

```
Record select topics:
$ rosvbag record topic1 topic2
Replay all messages without waiting:
$ rosvbag play -a demo.log.bag
Replay several bag files at once:
$ rosvbag play demo1.bag demo2.bag
```

Introspection and Command Tools

rosmmsg/rossrv

Displays Message/Service (msg/srv) data structure definitions.

Commands:

rosmmsg show	Display the fields in the msg/srv.
rosmmsg list	Display names of all msg/srv.
rosmmsg md5	Display the msg/srv md5 sum.
rosmmsg package	List all the msg/srv in a package.
rosmmsg packages	List all packages containing the msg/srv.

Examples:

```
Display the Pose msg:
$ rosmmsg show Pose
List the messages in the nav_msgs package:
$ rosmmsg package nav_msgs
List the packages using sensor_msgs/CameraInfo:
$ rosmmsg packages sensor_msgs/CameraInfo
```

roslaunch

Displays debugging information about ROS nodes, including publications, subscriptions and connections.

Commands:

roslaunch ping	Test connectivity to node.
roslaunch list	List active nodes.
roslaunch info	Print information about a node.
roslaunch machine	List nodes running on a machine.
roslaunch kill	Kill a running node.

Examples:

```
Kill all nodes:
$ roslaunch kill -a
List nodes on a machine:
$ roslaunch machine aqy.local
Ping all nodes:
$ roslaunch ping --all
```

rostopic

A tool for displaying information about ROS [topics](#), including publishers, subscribers, publishing rate, and messages.

Commands:

rostopic bw	Display bandwidth used by topic.
rostopic echo	Print messages to screen.
rostopic find	Find topics by type.
rostopic hz	Display publishing rate of topic.
rostopic info	Print information about an active topic.
rostopic list	List all published topics.
rostopic pub	Publish data to topic.
rostopic type	Print topic type.

Examples:

```
Publish hello at 10 Hz:
$ rostopic pub -r 10 /topic_name std_msgs/String hello
Clear the screen after each message is published:
$ rostopic echo -c /topic_name
Display messages that match a given Python expression:
$ rostopic echo --filter "m.data=='foo'" /topic_name
Pipe the output of rostopic to rosmmsg to view the msg type:
$ rostopic type /topic_name | rosmmsg show
```

roslaunch

A tool for getting and setting ROS [parameters](#) on the parameter server using YAML-encoded files.

Commands:

roslaunch set	Set a parameter.
roslaunch get	Get a parameter.
roslaunch load	Load parameters from a file.
roslaunch dump	Dump parameters to a file.
roslaunch delete	Delete a parameter.
roslaunch list	List parameter names.

Examples:

```
List all the parameters in a namespace:
$ roslaunch list /namespace
Setting a list with one as a string, integer, and float:
$ roslaunch set /foo "[1, 1, 1.0]"
Dump only the parameters in a specific namespace to file:
$ roslaunch dump dump.yaml /namespace
```

rosservice

A tool for listing and querying ROS services.

Commands:

rosservice list	Print information about active services.
rosservice node	Print name of node providing a service.
rosservice call	Call the service with the given args.
rosservice args	List the arguments of a service.
rosservice type	Print the service type.
rosservice uri	Print the service ROSRPC uri.
rosservice find	Find services by service type.

Examples:

```
Call a service from the command-line:
$ rosservice call /add_two_ints 1 2
Pipe the output of rosservice to rossrv to view the srv type:
$ rosservice type add_two_ints | rossrv show
Display all services of a particular type:
$ rosservice find rospy_tutorials/AddTwoInts
```


ROS Indigo Catkin Workspaces

Create a catkin workspace

Setup and use a new catkin workspace from scratch.

Example:

```
$ source /opt/ros/hydro/setup.bash
$ mkdir -p ~/catkin_ws/src
$ cd ~/catkin_ws/src
$ catkin_init_workspace
```

Checkout an existing ROS package

Get a local copy of the code for an existing package and keep it up to date using [wstool](#).

Examples:

```
$ cd ~/catkin_ws/src
$ wstool init
$ wstool set tutorials --git git://github.com/ros/ros_tutorials.git
$ wstool update
```

Create a new catkin ROS package

Create a new ROS catkin package in an existing workspace with [catkin create package](#). After using this you will need to edit the [CMakeLists.txt](#) to detail how you want your package built and add information to your [package.xml](#).

Usage:

```
$ catkin_create_pkg <package_name> [depend1] [depend2]
```

Example:

```
$ cd ~/catkin_ws/src
$ catkin_create_pkg tutorials std_msgs rospy roscpp
```

Build all packages in a workspace

Use [catkin make](#) to build all the packages in the workspace and then source the setup.bash to add the workspace to the [ROS_PACKAGE_PATH](#).

Examples:

```
$ cd ~/catkin_ws
$ ~/catkin_make
$ source devel/setup.bash
```