

# ROS Cheat Sheet

The PDF of the images can be downloaded

here

## 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.  
**roseed** Open requested ROS file in a text editor.  
**roscp** Copy a file from one place to another.  
**roscdep** 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]]
$ roseed [package] [file]
$ roscp [package] [file] [destination]
$ roscdep 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

#### rosviz

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

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

Examples:  
Record select topics:  

```
$ rosviz record topic1 topic2
```

Replay all messages without waiting:  

```
$ rosviz play -a demo_log.bag
```

Replay several bag files at once:  

```
$ rosviz play demo1.bag demo2.bag
```

### Introspection and Command Tools

#### rosmg/rossrv

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

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

Examples:  
Display the Pose msg:  

```
$ rosmg show Pose
```

List the messages in the nav\_msgs package:  

```
$ rosmg package nav_msgs
```

List the packages using sensor\_msgs/CameraInfo:  

```
$ rosmg packages sensor_msgs/CameraInfo
```

#### rostopic

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

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

Examples:  
Kill all nodes:  

```
$ rostopic kill -a
```

List nodes on a machine:  

```
$ rostopic machine aqy.local
```

Ping all nodes:  

```
$ rostopic 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 "n.data=='foo'" /topic_name
```

Pipe the output of rostopic to rosmg to view the msg type:  

```
$ rostopic type /topic_name | rosmg show
```

#### roscpp

A tool for getting and setting ROS parameters on the parameter server using YAML-encoded files.

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

Examples:  
List all the parameters in a namespace:  

```
$ roscpp list /namespace
```

Setting a list with one as a string, integer, and float:  

```
$ roscpp set /foo ["1", 1, 1.0]
```

Dump only the parameters in a specific namespace to file:  

```
$ roscpp 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 rosviz to view the srv type:  

```
$ rosservice type add_two_ints | rosviz show
```

Display all services of a particular type:  

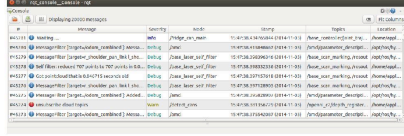
```
$ rosservice find rospy_tutorials/AddTwoInts
```

# ROS Indigo Cheatsheet

## Logging Tools

### rqt\_console

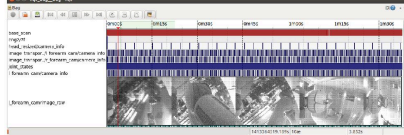
A tool to display and filtering messages published on rosout.



Usage:  
\$ rqt\_console

### rqt\_bag

A tool for visualizing, inspecting, and replaying bag files.



Usage, viewing:  
\$ rqt\_bag bag\_file.bag  
Usage, bagging:  
\$ rqt\_bag \*press the big red record button.\*

### rqt\_logger\_level

Change the logger level of ROS nodes. This will increase or decrease the information they log to the screen and rqt\_console.

Usage:  
viewing \$ rqt\_logger\_level

## Introspection & Command Tools

### rqt\_top

A tool for viewing published topics in real time.

Usage:  
\$ rqt  
Plugin Menu->Topic->Topic Monitor

### rqt\_msg, rqt\_srv, and rqt\_action

A tool for viewing available msgs, srvs, and actions.

Usage:  
\$ rqt  
Plugin Menu->Topic->Message Type Browser  
Plugin Menu->Service->Service Type Browser  
Plugin Menu->Action->Action Type Browser

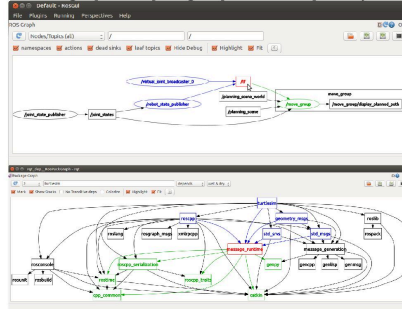
## rqt\_publisher, and rqt\_service\_caller

Tools for publishing messages and calling services.

Usage:  
\$ rqt  
Plugin Menu->Topic->Message Publisher  
Plugin Menu->Service->Service Caller

## rqt\_graph, and rqt\_dep

Tools for displaying graphs of running ROS nodes with connecting topics and package dependencies respectively.



Usage:  
\$ rqt\_graph  
\$ rqt\_dep

## rqt\_top

A tool for ROS specific process monitoring.

Usage:  
\$ rqt  
Plugin Menu->Introspection->Process Monitor

## rqt\_reconfigure

A tool for dynamically reconfiguring ROS parameters.

Usage:  
\$ rqt  
Plugin Menu->Configuration->Dynamic Reconfigure

## Development Environments

### rqt\_shell, and rqt\_py\_console

Two tools for accessing an xterm shell and python console respectively.

Usage:  
\$ rqt  
Plugin Menu->Miscellaneous Tools->Shell  
Plugin Menu->Miscellaneous Tools->Python Console

## Data Visualization Tools

### tf\_echo

A tool that prints the information about a particular transformation between a source\_frame and a target\_frame.

Usage:  
\$ rosrn tf tf\_echo <source\_frame> <target\_frame>

Examples:  
To echo the transform between /map and /odom:  
\$ rosrn tf tf\_echo /map /odom

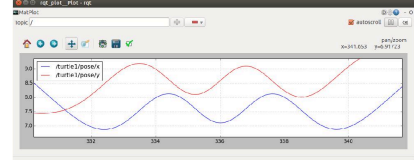
### view\_frames

A tool for visualizing the full tree of coordinate transforms.

Usage:  
\$ rosrn tf2.tools.view\_frames.py  
\$ evince frames.pdf

### rqt\_plot

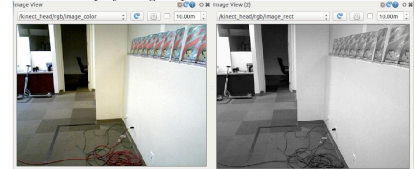
A tool for plotting data from ROS topic fields.



Examples:  
To graph the data in different plots:  
\$ rqt\_plot /topic1/field1 /topic2/field2  
To graph the data all on the same plot:  
\$ rqt\_plot /topic1/field1,/topic2/field2  
To graph multiple fields of a message:  
\$ rqt\_plot /topic1/field1:field2:field3

### rqt\_image\_view

A tool to display image topics.



Usage:  
\$ rqt\_image\_view

## 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
```

Copyright © 2015 Open Source Robotics Foundation  
Copyright © 2010 Willow Garage

From:

<http://robosub-vm.eecs.wsu.edu/wiki/> - **Palouse RoboSub Technical Documentation**

Permanent link:

<http://robosub-vm.eecs.wsu.edu/wiki/cs/ros/cheatsheet/start?rev=1473630027>



Last update: **2016/09/11 14:40**