

ROS Cheat Sheet

This will be converted to an image later.

roscheatsheet_catkin.pdf

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.
roset Open requested ROS file in a text editor.
roscp Copy a file from one place to another.
roscp Installs package system dependencies.
roscp 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`
`$ rostd`
`$ rosls [package[/subdir]]`
`$ roset [package] [file]`
`$ roscp [package] [file] [destination]`
`$ roscp install [package]`
`$ rosvtf or rosvtf [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

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`

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 rosmmsg to view the msg type:
`$ rostopic type /topic_name | rosmmsg show`

rosparam

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

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

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

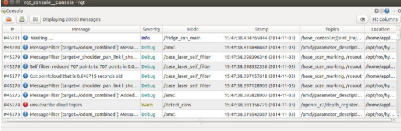
Examples:
 Call a service from the command-line:
`$ rosservice call /add_two_ints 1 2`
 Pipe the output of rosservice to rosviz to view the srv type:
`$ rosservice type add_two_ints | rosviz show`
 Display all services of a particular type:
`$ rosservice find rospy_tutorials/AddTwoInts`

ROS Indigo Cheatsheet

Logging Tools

rqt.console

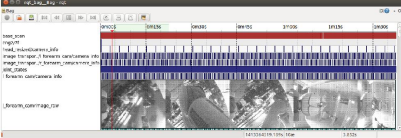
A tool to display and filtering messages published on rosout.



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.topic

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

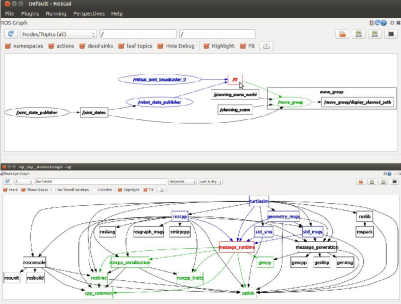
rqt_publisher, and rqt_service_caller

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:
\$ roslaunch tf tf_echo <source_frame> <target_frame>

Examples:
To echo the transform between /map and /odom:
\$ roslaunch tf tf_echo /map /odom

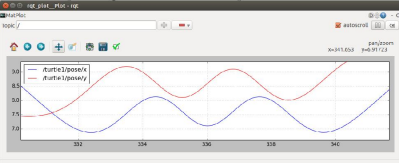
view_frames

A tool for visualizing the full tree of coordinate transforms.

Usage:
\$ roslaunch tf2_tools view_frames.py
\$ evince frames.pdf

rqt_plot

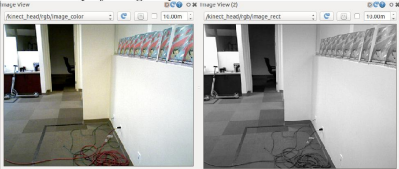
A tool for plotting data from ROS topic fields.



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 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:

<https://robosub.eecs.wsu.edu/wiki/> - Palouse RoboSub Technical Documentation

Permanent link:

<https://robosub.eecs.wsu.edu/wiki/cs/ros/cheatsheet/start?rev=1473629954>



Last update: **2016/09/11 14:39**