Robotic Operating System - Cheat Sheet (ROS Melodic)

File system **Command-Line Tools**

apt-cache search ros-indigo	Search for available packages on Ubuntu	
rospack/rosstack	A tool inspecting packages/stacks	Usage: rospack find [package]
roscd	Changes directories to a package or stack.	Usage: roscd [package[/subdir]]
rosls	package or stack information.	Usage: rosls [package[/subdir]]
roscreate-pkg	Creates a new ROS package.	Usage: roscreate-pkg [package name]
roscreate-stack	Creates a new ROS stack.	Usage: roscreate-stack [path]
rosdep	Installs ROS package system dependencies.	Usage: rosdep install [package]
rosmake	Builds a ROS package.	Usage: rosmake [package]
roswtf	Displays errors and warnings about a running ROS system or launch file.	Usage: roswtf or roswtf [file]
catkin_create.pkg	Creates a new ROS stack	Usage: catkin create pkg [package name] [depend1]..[dependN]
wstool	Manage many repos in workspace.	Usage: wstool [init | set | update]
catkin_make	Builds a ROS catkin workspace.	Usage: catkin make
rqt_dep	Displays package structure and dependencies.	Usage: rqt dep [options]

Workspace Methods

Creating a Workspace

```
mkdir catkin_ws && cd catkin_ws    #Create a Directory catkin_ws and change to this directory
wstool init src                    #Initialize the Workspace Without a rosinstall file
catkin_make                        #Builds a ROS catkin workspace
source devel/setup.bash
```

Workspace in Dependencies Resolve

```
sudo rosdep init                   #install ros dependencies this is done only once
rosdep update                      #update ros dep Do NOT run rosdep update with sudo.
rosdep install
--from-paths src --ignore-src \
--rosdistro=${ROS_DISTRO} -y      #installs all the packages that the packages in your catkinspace
```

Repo addition to workspace

```
roscd; cd../src                                 #//Change directory

wstool set repo_name  \
--git http://github.com/org/repo_name.git \
--version=,melodic-devel
                                                #add or changes one entry from your workspace
                                                wstool set [localname]
                                                [SCM-URI]?[--(detached|svn|hg|git|bzr)]
                                                [--version=VERSION]]

wstool update                                   # wstool update
```

CMakeLists.txt

Skeleton

```
cmake_minimum_required(VERSION 2.9.0)   #Set the minimum required version of cmake (2.8.3+ for Kinetic )
project(package_name)                   #Sets the name of the project, and stores it in the variable PROJECT_NAME.
find_package(catkin REQUIRED)           #Find an external project, and load its settings.
catkin_package()                        #This is required to specify catkin-specific information to the build system
```

Package Dependencies

`find_package(catkin REQUIRED COMPONENTS roscpp)`	To use headers or libraries in a package's exported CMake macros; expresses a built-time dependency

`catkin_package(`
`INCLUDE_DIRS include`
`LIBRARIES ${PROJECT_NAME}`
`CATKIN_DEPENDS roscpp`
	Tell dependent packages what headers or libraries to pull in when you package is declared as a catkin component

`catkin_lint`	Checks package configurations for the catkin build system of ROS. Prompts you with errors, if any.
`build_depend`	Defines whats needed for building the package. The dependency must be defined in find_package too.
`build_export_depend`	Is a secondary build_depend and only relevant when providing libraries. It means that if someone is building with your library, but requires another package, it should specified by build_export_depend. It also needs to be set in catkin_package.
`exec_depend`	To tell which packages are required for execution.
`test_depend`	Defines what package is needed for testing. This package should also be defined in find_package in the if (CATKIN_ENABLE_TESTING) block

Note that any packages listed as CATKIN_DEPENDS dependencies must also be decleared as a <run_depend> in package.xml for package.xml formats 1 and 2. For format 3 we at Sedenius use exec_depend.

Messages,Services

```
#Add the following only after find_package(),but before catkin_package()
find_package(catkin REQUIRED COMPONENTS
             message_generation std_msgs)
add_message_files(FILES MyMessage.msg)            #To handle messages
add_service_files(FILES MyServices.msg)           #To handle Serices
add_action_files(FILES MyAction.msg)              #To handle Actions
generate_messages (DEPENDENCIES std_msgs)
catkin_package(CATKIN_DEPENDS
               message_runtime std_msgs)
```

Executables, Libraries Build

```
Add the following after catkin_package() call.
add_library(${PROJECT_NAME} src/main)                  #add Library using specific source files.
add_executable(${PROJECT_NAME}_node src/main)          #add an executable using specific src files.
target_link_libraries(
      ${PROJECT_NAME}_node ${catkin_LIBRARIES})        #Specify libraries or flags to use when linking.
```

CMakeLists.txt

Installation

```
install (TARGETS ${PROJECT_NAME}
      DESTINATION ${CATKIN_PACKAGE_LIB_DESTINATION})
install (TARGETS ${PROJECT_NAME}_node
      DESTINATION ${CATKIN_PACKAGE_BIN_DESTINATION})
install (PROGRAMS scripts/yourscripts
      DESTINATION ${CATKIN_PACKAGE_BIN_DESITNATION})
install (DIRECTORY launch
      DESTINATION ${CATKIN_PACKAGE_SHARE_DESTINATION})
```

Logging Tools

`rqt_console`	A tool to display and filtering messages published on rosout.	$rqt_console
`rqt_bag`	A tool for visualizing, inspecting, and replaying bag files.	$ rqt_bag bag file.bag
`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	$rqt_bag *press the big red record button

`rosbag`	A set of tools for recording and playing back of ROS topics
	
	rosbag record — Record a bag file with specified topics.
	rosbag play — Play content of one or more bag files.
	rosbag compress — Compress one or more bag files.
	rosbag decompress — Decompress one or more bag files.
	rosbag filter — Filter the contents of the bag.
		Record select topics:
		$ rosbag record topic1 topic2
		Replay all messages without waiting:
		$ rosbag play -a demo log.bag
		Replay several bag files at once:
		$ rosbag play demo1.bag demo2.bag

`tf_echo`	A tool that prints the information about a particular transformation between a source frame and a target frame.	$ rosrun tf tf echo <source frame> <target frame>
		To echo the transform between /map and /odom
		$ rosrun tf tf echo /map /odom

Sedenius Engineering provides custom Advance ROS tools for advance works flows. Visit www.sedenius.com/produkte

Introspection & Command Tools

`rqt_topic`	A tool for viewing published topics in real time.	$rqt Plugin Menu->Topic->Topic Monitor
`rqt_msg,rqt_srv` and `rqt_action`	A tool for viewing available msgs, srvs, and actions.	$rqt Plugin Menu->Topic->Message Type Browser Plugin Menu->Topic->Services Type Browser Plugin Menu->Topic->Action Type Browser
`rqt_publisher` `rqt_servicecaller`	Tools for publishing messages and calling services.	$rqt Plugin Menu->Topic->Message Publisher Plugin Menu->Topic->Services Caller
`rqt-reconfigure`	Tools for publishing messages and calling services.	$rqt Plugin Menu->Configuration->Dynamic Reconfigure
`rqt-graph` and `rqt-dep`	Tools for displaying graphs of running ROS nodes with connecting topics and package dependancies respectively	$ rqt graph $ rqt dep
`rqt-top`	A tool for ROS specific process monitoring.	$ rqt Plugin Menu->Introspection->Process Monitor

`rosnode`	Displays debugging information about ROS nodes, including publications, subscriptions and connections.
	Commands:
	rosnode ping — Test connectivity to node.
	rosnode list — List active nodes.
	rosnode info — Print information about a node.
	rosnode machine — List nodes running on a machine.
	rosnode kill — Kill a running node.
		Kill all nodes:
		$ rosnode kill -a
		List nodes on a machine:
		$ rosnode machine aqy.local
		Ping all nodes:
		$ rosnode ping --all

Introspection & Command Tools

`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
		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

`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

`rosmsg/rossrv`	Displays Message/Service (msg/srv) data structure definitions.
	Commands:
	rosmsg show — Display the fields in the msg/srv.
	rosmsg list — Display names of all msg/srv.
	rosmsg md5 — Display the msg/srv md5 sum.
	rosmsg package — List all the msg/srv in a package.
	rosmsg packages — List all packages containing the msg/srv
		Examples:
		Display the Pose msg:
		$ rosmsg show Pose
		List the messages in the nav msgs package:
		$ rosmsg package nav msgs
		List the packages using sensor msgs/CameraInfo:
		$ rosmsg packages sensor msgs/CameraInfo

Running System

Nodes,Topics,Messages

```
rosnode  list
rostopic list
rostopic echo cmd_vel
rostopic hz cmd_vel
rostopic info cmd_vel
rosmsg   show geometry_msgs/Twist
```

Remote Connection

Master's ROS enviroment	`ROS_IP or HOSTNAME`	Set to this machine's network address
	`ROS_MASTER_URI`	Set to URI containing that IP or hostname
Your enviroment	`ROS_IP or HOSTNAME`	Set to this machine's network address
	`ROS_MASTER_URI`	Set to URI from master

To debug,
    Check ping from each side to the other ,
    Run roswtf on each side.

SEDENIUS TECHNOLOGIES

ROS