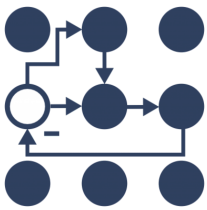
Contents

ROS GitHub Organizations

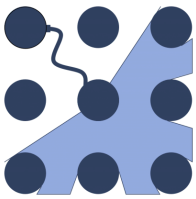
There are several GitHub organizations which host many of the core ROS packages. Below you will find a list of the most common ones with a quick description of their focus:

ROS Android (https://github.com/ros-android)
Android apps based on ROSJava. Also see ROS Android PR2 (https://github.com/ros-android-pr2) and Link to Play Store (https://play.google.com/store/apps/developer?id=Willow+Garage&hl=en)



ROS Controls (https://github.com/ros-controls)
Controllers and interfaces for hardware

ROS Core (https://github.com/ros)
Core ROS libraries and tools such as standard messages, the build system catkin (/catkin), the ROS clients roscpp (/roscpp) and rospy (/rospy), actionlib (/actionlib), pluginlib (/pluginlib), nodelet (/nodelet), etc



ROS Drivers (https://github.com/ros-drivers)
Drivers for sensors and other hardware

ROS Geographic Information (https://github.com/ros-geographic-info)
Packages for managing geographic information within ROS

ROS GitBuildPackage (GBP) (https://github.com/ros-gbp)
Git Build Package style repositories for the release pipeline.

ROS Infrastructure (https://github.com/ros-infrastructure)
Foundational tools for ROS ecosystem such as package release (bloom (/bloom)), build farm scripts, documentation generators. Most of these are backend, non-user facing repositories.

ROS Java (https://github.com/organizations/rosjava)
The Java ROS client library and tools

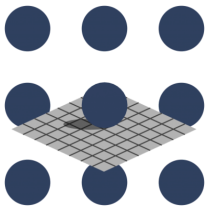
ROS Manipulation (https://github.com/ros-manipulation)
Packages relating to manipulation and the manipulation pipeline.

ROS Interactive Manipulation (https://github.com/ros-interactive-manipulation)
Packages relating to interactive manipulation and the manipulation pipeline.

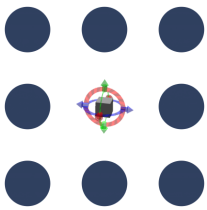
ROS Perception (https://github.com/ros-perception)
Tools for perception and vision

ROS Planning (https://github.com/ros-planning)
Navigation and path planning related packages including navigation (/navigation) and moveit (/moveit)

ROS Robots (https://github.com/ros-robots)
Packages to support robots which do not have their own organization. See also PR2 (https://github.com/PR2) and Turtlebot (https://github.com/turtlebot)



ROS Simulation (https://github.com/ros-simulation)
Simulators for use in ROS



ROS Visualization (https://github.com/ros-visualization)
Graphical tools to visualize data including rviz (/rviz) and rqt (/rqt).

A historical list of institutions that have contributed to ROS is available on the repositories (/Repositories) page.

It is not required that if you are doing something related to one of these categories that your package be in one of the organization. And every choice of where to put a packages is always a judgement call for most packages could be multiply categorized. The goal of using these organizations is to facilitate maintenance so grouping software by developer communities is often the best way to decide where to host a package.

1. Guidelines for administering organizations

These are general guidelines for anyone administering a repository with ROS code.

- All package maintainers should have administrative access to the repositories in which they maintain packages to enable them to process tickets and do their role.
- All package maintainers should be publicly visible as members of the organization.
- The main point of contact for the organization should be listed as a public member.

2. Guidelines for administering ros-* organizations on GitHub

In addition to the general guidelines for administering an organization on GitHub listed above. If the organization is using the ROS name in the ros-* format:

- The ROS Platform Manager should also be in the Owners group to facilitate handoff's between maintainers should there be a gap.

Wiki: RecommendedRepositoryUsage/CommonGitHubOrganizations (last edited 2014-03-04 18:35:40 by TullyFoote (/TullyFoote))