# Baxter入门整理：

## 环境配置

|  |  |
| --- | --- |
| 1.Install Ubuntu 14.04 | 1.注意：   * 添加清华镜像源 |
| 2.Suitable python version | 1.安装注意：   * 在2.7之外再安装3.5，用virtual env 进行版本控制，但是indigo的默认是python2.7.不妨先使用2.7? * Python3.6 conda安装 |
| 3.Indigo | 1.安装问题记录：   * 添加秘钥时遇到server out of time. 解决方法参考：https://askubuntu.com/questions/574569/apt-get-stuck-at-0-connecting-to-us-archive-ubuntu-com/575130#575130 |
| 4.Ros for baxter | 1.安装问题：   * Install sdk 的时候遇到：**error: option --install-layout not recognized**     catkin\_make install -DCMAKE\_BUILD\_TYPE=Release -DSETUPTOOLS\_DEB\_LAYOUT=OFF  hello |
| 5.PyQt4 | conda install pyqt=4 |
| 6.Hello Baxter  参考http://sdk.rethinkrobotics.com/wiki/Baxter\_Research\_Robot\_Software\_Developers\_Kit\_(SDK)即可 |  |

## ROS入门

|  |  |
| --- | --- |
| 1.Ros系统概述：  ROS 是一个适用于机器人的开源的元操作系统。它提供了操作系统应有的服务，包括硬件抽象，底层设备控制，常用函数的实现，进程间消息传递，以及包管理。它也提供用于获取、编译、编写、和跨计算机运行代码所需的工具和库函数。在某些方面ROS相当于一种“机器人框架。 | 5.注意： gzable和vrep |
| 2.Ros 概念：   * File system:   + Packages   + Meta packages   + Package manifest(package.xml)   + Repositories   + Msg types: my\_package/msg/MyMessageType.msg   + Srv types: my\_package/srv/MyServiceType.srv * Compute Graph:   + Node   + Master   + Parameter Server(Part of Master)   + Topics   Messages are routed via a transport system with publish / subscribe semantics. A node sends out a message by publishing it to a given topic. The topic is a name that is used to identify the content of the message. A node that is interested in a certain kind of data will subscribe to the appropriate topic. There may be multiple concurrent publishers and subscribers for a single topic, and a single node may publish and/or subscribe to multiple topics. In general, publishers and subscribers are not aware of each others' existence. The idea is to decouple the production of information from its consumption. **Logically, one can think of a topic as a strongly typed message bus.** Each bus has a name, and anyone can connect to the bus to send or receive messages as long as they are the right type.   * The publish / subscribe model is a very flexible communication paradigm, but its many-to-many, one-way transport is not appropriate for request / reply interactions, which are often required in a distributed system. Request / reply is done via services, which are defined by a pair of message structures: one for the request and one for the reply. A providing node offers a service under a name and a client uses the service by sending the request message and awaiting the reply. ROS client libraries generally present this interaction to the programmer as if it were a remote procedure call.      * Name space:   + Name 分为4种：base/relative/global/private:   Node  / node I  /wg/node2  /wg/node3  Relative (default)  bar -> /wg/bax  foo/bax -> /wg/foo/bax  GIObal  'Wax -> 'Wax  'Wax -> 'Wax  / foo/bax -> / foo/bax  Private  -> / nodel/bax  -> /wg/node2/bax  foo/bax -> /wg/node3/foo/bax | 3.区别：   * Topics与Service的区别在于Topics是一个单向信道，类似一个数据总线；而Service则是一个query/response的结构，Service使node对命令做出响应。 * Master不是交换信道，其只是作为类似DNS服务器那样的提供地址的中间介质。   Service invocation  Node  Topic  Node  ROS basic_concepts dia  Publication  Subscri ption |
| 3.Ros网络服务技术:  Ros内部使用TCP与UDP两种网络协议进行数据传输，一个系统内的网络协议并不唯一。Node之间彼此之间相连构成信道。Master只起到辅助链接，给定位置，作为一个协商信道。一个典型的过程如下所示：     * 1. Subscriber starts. It reads its command-line remapping arguments to resolve which topic name it will use. (Remapping Arguments)   2. Publisher starts. It reads its command-line remapping arguments to resolve which topic name it will use. (Remapping Arguments)   3. Subscriber registers with the Master. (XMLRPC)   4. Publisher registers with the Master. (XMLRPC)   5. Master informs Subscriber of new Publisher. (XMLRPC)   6. Subscriber contacts Publisher to request a topic connection and negotiate the transport protocol. (XMLRPC)   7. Publisher sends Subscriber the settings for the selected transport protocol. (XMLRPC)   8. Subscriber connects to Publisher using the selected transport protocol. (TCPROS, etc...)   ODG version  hokuyo  XML/RPC: foo:1234  TCP data: fm.•2345  4.  V—XMURpc  Tr.P)  TCP server. f002345  connect(fcn2345)  LaserScan data messages  TCP | 3.区分：  node是通讯对象的单元以及个体，这个个体可以是subscriber也可以是publisher.他们之间的关系相当于是人和厨师的关系。 |
| * 1. Creat ROS package      + Catkin 文件系统的组成： package.xml CMakeList.txt      + Cmake 结构，cmakelist是层次化结构，每层package都会有cmakelist.      + Catkin的结构package create工具：   $ catkin\_create\_pkg beginner\_tutorials std\_msgs rospy roscpp  # catkin\_create\_pkg <package\_name> [depend1] [depend2] [depend3]  典型的分布式构造结构   * + Dependency observer:     - rospack depends1 beginner\_tutorials 可以观察一阶依赖   + 理解Package.xml     - <?xml version="1.0"?> <package format="2">  <name>beginner\_tutorials</name>  <version>0.1.0</version>  <description>The beginner\_tutorials package</description>   <maintainer email="you@yourdomain.tld">Your Name</maintainer>  <license>BSD</license>  <url type="website">http://wiki.ros.org/beginner\_tutorials</url>  <author email="you@yourdomain.tld">Jane Doe</author>   <buildtool\_depend>catkin</buildtool\_depend>   <build\_depend>roscpp</build\_depend>  <build\_depend>rospy</build\_depend>  <build\_depend>std\_msgs</build\_depend>   <exec\_depend>roscpp</exec\_depend>  <exec\_depend>rospy</exec\_depend>  <exec\_depend>std\_msgs</exec\_depend>  </package>     - Package.xml由name|description|license|denpendency等结构构造而成。   + catkin\_make的两种语法：     - 整体make: catkin\_make| catkin\_install     - 对局部make: catkin\_make --source my\_src |catkin\_make install --source my\_src   + 安装完配置环境的过程：   + To add the workspace to your ROS environment you need to source the generated setup file:   + $ . ~/catkin\_ws/devel/setup.bash |  |