

Team-Xmbot-Service-Robot ROS 风格指南

0. 前言

0.1 版本

- 1.0版本(2016.5.1)：缪宇颺(myyerrol)创建团队 ROS 开发风格指南。本文档参考了 [ROS 官方风格指南](#)，并根据实际的需求，对内容进行了适当的精简和改进。新队员应该认真学习本指南，掌握 ROS 基本的开发风格以及适用于本项目组的特定规则。如果有细节不统一的地方或者对本文档某处不是很认同，请在组内讨论统一之后，修改本指南。因为文档排版使用的是 Markdown 纯文本标记语言，也请后来者遵循本文档的开发方式，使用 Markdown 来修改、添加内容。

0.2 背景

Team-Xmbot-Service-Robot（晓萌家庭服务机器人团队）作为开源项目，需要团队队员贡献代码，但是如果队员之间所使用的 ROS 风格不一致，便会给团队其他模块负责人造成不小的困扰。我们认为整洁、一致的 ROS 开发风格会使整个项目更加可管理和维护。因此，我们应当使用统一的 ROS 开发风格以使得每个功能包不仅能够在现在发挥作用，而且在将来的若干年之后其依旧能够存在、可以被复用、或者是能够被未来的新队员改进。

1. ROS 命名指南

Tip

所有命名的通用的标准是：

- 命名要足够详细（不要吝啬长度），让开发者能通过名字知道你的意图。

1.1 Packages

Tip

包的命名要**全部小写**、使用统一的项目前缀，并且要足够详细。

- 包的名字必须遵守 C++ 变量命名的规范：**全部小写**、以字母开始、单词之间使用下划线连接。比如：`xm_arm_moveit_config`。
- 每个包的名字前都要有项目的根或子前缀。要求：如果包是作用于整个机器人或多个模块的，那么包的前缀应该为 `xm_`。比如：`xm_robot_hardware`（里面包括了底盘和机械臂两个模块部分）。如果包的作用域是针对于机器人的某一个模块的话，那包的前缀应在原先根前缀 `xm_` 的基础上再添加相应模块的名字来构成。比如：`xm_arm_`、`xm_base_`、`xm_speech_`、`xm_vision_` 等。举个完整的包命名例子：`xm_arm_robot_hardware`（只与机械臂模块相关）。
- BSR 项目请修改根前缀为 `xm_bsr_`，其他规则与上面一样。
- 仿真项目请修改根前缀为 `xm_sim_`，其他规则与上面一样。
- 包的名字应该足够的详细。比如：一个有关机械臂的运动规划包，那么 `xm_arm_motion_planning` 要比 `xm_arm_planning` 好一些。
- 如果包的内容是基于其他第三方包的话，请在名字中进行指定。比如：一个基于 Moveit 的机械臂逆解算插件，就可以命名为 `xm_arm_moveit_ik_plugin`。
- 元包（Metapackages）的命名比较特殊。因为元包包含了若干个子包，所以在给元包命名的时候，取一个能概括所有子包含义的名字是必要的。
- 其他的请参照 [ROS Hector 团队](#) 代码仓库中包的命名规范。

1.2 Messages

Tip

所有 `.msg`、`.srv`、`.action` 文件命名使用组内特定的规则。

- 所有 `.msg`、`.srv`、`.action` 文件命名规则为：首先添加 `xm_` 前缀，之后添加每个文件具体的功能描述，描述部分必须使用驼峰命名法。例如：`xm_GripperCommand.action`。

1.3 Lanuch / Config / Xacro

Tip

所有机器人启动、配置、描述文件命名全部小写，并且描述的尽可能**清楚**。

1.4 Nodes

Tip

命名要小写，节点的类型（程序的名字）可以与包名中的某些单词重复，但一定要保证通过名字能准确知道其功能。

节点有类型和名字。节点类型指的是可执行程序的名字，而节点名字指的是启动后生成的，可以与其他节点通信的名字。

- 节点名字与节点类型一样。要**全部小写**、使用下划线作为单词间的连接，并用有意义的单词来命名。
- 通常，节点名字要与节点类型相同。但如果要启动多个同样类型的节点，请更改节点名字，使它们互不相同。
- 通常，节点类型应该尽可能短，因为它们中的一部分已经被包的名字所包含了。比如：假设存在一个包 **laser_scan**，其中包含观察激光数据的程序，那程序的名字应该命名为 **viewer**，而不是 **laser_scan_viewer**。但是本人不太认同以上官方的说法，因为这与 C++ 代码命名指南相冲突（代码本身要命名的足够详细）。这条规则可以成立的条件是要把代码名和包名联系起来看，但如果只看代码名字的话，我想很难从有限的单词中可以看出代码所要实现的功能。因此，**这条规则可以忽略掉。只要代码的名字好理解，与包名某些单词重复也是没有关系的。**比如：

```
$ rosrn xm_arm_teleop xm_arm_teleop_joint_position_keyboard
```

1.5 Topic / Service / Action

Tip

命名小写、具有描述性、且添加适合的前缀来明确名字的作用范围。

topic、service 和 action 的名字是节点服务端、客户端之间通信的桥梁。它们存在于一个分层的命名空间中，客户端便可以提供机制在运行时来重映射它们的名字。因此，它们相较于包的命名更加灵活。

- topic、service 和 action 的名字遵循 C++ 变量命名指南：小写、下划线。
- 命名应该足够的具有描述性。并且最好添加相应的前缀来指明 topic、service、action 的作用范围。比如：假设机械臂关节节点要通过 topic 发布关节状态数据，那命名成 **/joint_states** 要比 **/states** 好。**/xm_arm/joint_states** 又要比 **/joint_states** 好。

- 如果某些程序发布的 topic、service、action 名字没有使用前缀，请在 launch 文件中使用 `<remap>` 来重映射。比如：节点 `robot_state_publisher` 默认订阅的 Topic 是 `/joint_states`，使用重映射可以使其订阅 `/xm_arm/joint_states` 上的数据。

2. ROS 格式指南

Tip

所有格式的通用标准是：

- 在 ROS 包中，除代码外，其他跟 ROS 有关的文件都需要使用官方指定的 2 格缩进。
- 所有文件的末尾都要留出一个空行，不能多也不能少。

2.1 Package.xml

Tip

使用项目组内特定的风格。

package.xml 是每个 ROS 包都必须包含的，可以通过使用 `catkin_create_pkg` 自动生成，其他详细介绍请看 [ROS package](#)。以下是 package.xml 的格式风格。

- `version` 版本标签的含义为：主版本-子版本-修改次数。现在所有包的主版本默认为 1，子版本和修改次数默认为 0。即初始化为 `1.0.0`。之后，主、子版本和修改次数的值会伴随 ROS 包的修改而不断变化。当修改次数达到一定值时，可以将子版本加 1，而修改次数重新归零。以此类推，如果子版本数增到一定程度时，就可以将主版本加 1，而其他两个归零。对于修改到什么程度就可以向子版本或主版本进 1，请各模块负责人自己决定。举个例子：`1.0.0` -> `1.0.20` -> `1.1.0` -> `1.2.10` -> `2.0.0`。
- 内容全部使用 2 格缩进。
- 所有在 package.xml 中被注释的都要删除掉，只留下最后精简过的、有用的信息。
- `<maintainer>` 和 `<author>` 的区别在于，`<maintainer>` 是包的主要或最早开发者，而 `<author>` 则是之后为包提供修改的开发者。显然，出现在 `<maintainer>` 的名字也应该出现在 `<author>` 中。如果有其他开发者修补了 Bug 或进行了增量式的开发，请将名字写在 `<author>` 标签之后。
- 因为 ROS 使用的软件许可证协议是 BSD，所以我们所有的包也要以 BSD 协议发布出去。
- 空格和空行的使用请严格遵循下面完整例子所实现的那样。

- 如果包是元包（Metapackages），请在 package.xml 的末尾处添加以下内容：

```
<export>
  <metapackage />
</export>
```

以下是个完整的例子：

```
<?xml version="1.0"?>
<package>
  <name>xm_arm_robot_hardware</name>
  <version>0.0.1</version>
  <description>The xm_arm_robot_hardware package implements hardware i
nterface by using ros_control.</description>

  <maintainer email="myyerrol@126.com">myyerrol</maintainer>

  <license>BSD</license>

  <author email="myyerrol@126.com">myyerrol</author>

  <buildtool_depend>catkin</buildtool_depend>

  <build_depend>control_toolbox</build_depend>
  <build_depend>controller_manager</build_depend>
  <build_depend>hardware_interface</build_depend>
  <build_depend>realtime_tools</build_depend>
  <build_depend>roscpp</build_depend>
  <build_depend>sensor_msgs</build_depend>
  <build_depend>std_msgs</build_depend>

  <run_depend>control_toolbox</run_depend>
  <run_depend>controller_manager</run_depend>
  <run_depend>hardware_interface</run_depend>
  <run_depend>realtime_tools</run_depend>
  <run_depend>roscpp</run_depend>
  <run_depend>sensor_msgs</run_depend>
  <run_depend>std_msgs</run_depend>

</package>
```

2.2 CMakeLists.txt

Tip

使用项目组内特定的风格。

CMakeLists.txt 是 CMake 编译系统的输入文件，用来描述如何构建代码。更多详细的介绍请看 [ROS CMakeLists.txt](#)。以下简单地介绍编写或修改 CMakeLists.txt 所使用的格式风格。

- 内容全部使用 2 格缩进。
- 每个命令之间要空 1 行。
- 所有在 CMakeLists.txt 中被注释的都要删除掉，只留下最后精简过的、有用的信息。
- 每个命令中的元素最好每行只写一个，而且要对齐。这样看起来更清楚。
- 针对于元包（MetaPackages），请在 CMakeLists 中写入以下内容：

```
cmake_minimum_required(VERSION 2.8.3)
project(<PACKAGE_NAME>)
find_package(catkin REQUIRED)
catkin_metapackage()
```

以下是个完整的例子：

```
cmake_minimum_required(VERSION 2.8.3)
project(xm_arm_robot_hardware)

find_package(catkin REQUIRED COMPONENTS
  control_toolbox
  controller_manager
  hardware_interface
  realtime_tools
  roscpp
  sensor_msgs
  std_msgs
)

catkin_package()

include_directories(
  include
  ${catkin_INCLUDE_DIRS}
)

add_executable(
  xm_arm_robot_hardware
  include/xm_arm_robot_hardware/xm_arm_robot_hardware.h
  src/xm_arm_robot_hardware.cpp
  src/main.cpp
)

target_link_libraries(
  xm_arm_robot_hardware
  ${catkin_LIBRARIES}
)
```

2.3 Launch

Tip

使用项目组内特定的风格。

launch 文件作为 roslaunch 命令的输入，可以启动多个 ROS 节点并在参数服务器上设置参数。更多有关详细内容请关注 [ROS launch](#)。以下简介 launch 文件的格式风格。

- 所有内容 2 格缩进。
- 每个标签之间要空 1 行。如果是相同标签，则之间可以不空行。
- 每个标签的开头，最好能有一行注释来解释标签所要实现的功能。
- 每个标签的结束内容要与 `</>` 之间空一格。
- 每个标签内部的 `=` 前后是没有空格的。
- 针对根标签内部可以嵌入子标签的情况，请子标签缩进 2 个空格。
- 如果标签一行太长，则请适当的换行。换行之后请与上一行的元素对齐。
- `<node>` 的基本声明顺序为：`name`，`pkg`，`type`。

```
<node name="listener" pkg="rospy_tutorials" type="listener.py" />
```

- `<arg>` 的基本声明顺序为：`name`，`value`。

```
<arg name="foo" default="1" />
```

- `<param>` 的基本声明顺序为：`name`，`type`，`value`。

```
<param name="publish_frequency" type="double" value="10.0" />
```

- `<rosparam>` 的基本声明顺序为：`command`，`file`。

```
<rosparam command="load" file="$(find rosparam)/example.yaml" />
```

以下是完整的例子：


```

<launch>

  <!-- Load joint controller configurations from YAML file to parameter server -->
  <rosparam file="$(find xm_arm_gazebo_controller_config)/config/xm_arm_gazebo_joint_states.yaml" command="load" />

  <node name="joint_state_controller_spawner" pkg="controller_manager" type="spawner"
    respawn="false" output="screen" ns="/xm_arm"
    args="joint_state_controller" />

  <!-- Convert joint states to TF transforms for rviz, etc -->
  <node name="robot_state_publisher" pkg="robot_state_publisher" type="robot_state_publisher"
    respawn="false" output="screen">
    <remap from="/joint_states" to="/xm_arm/joint_states" />
  </node>

</launch>

```

2.4 License

Tip

使用基于 BSD 的项目组软件开发许可证。

因为，ROS 开发使用的是 BSD 软件开发许可证协议。所以我们项目组在发布代码和软件包的时候，也要使用 BSD 许可证协议。以下简介格式指南。

- 每个 ROS 软件包的目录下都要放置一个名为 LICENSE 的许可证文件。
- 每个 C++ 代码的头文件和源文件都要在开始放置许可证协议。
- 每个 Python 代码的头部都要放置许可证协议。

以下是相应的例子：

Software License Agreement (BSD License)

Copyright (c) 2016, Team-Xmbot-Service-Robot
All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

- * Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
- * Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
- * Neither the name of the Team-Xmbot-Service-Robot nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

```
/******
*   Software License Agreement (BSD License)
*
*   Copyright (c) 2016, Team-Xmbot-Service-Robot
*   All rights reserved.
*
*   Redistribution and use in source and binary forms, with or without
*   modification, are permitted provided that the following conditions
*   are met:
*
*   * Redistributions of source code must retain the above copyright
*     notice, this list of conditions and the following disclaimer.
*   * Redistributions in binary form must reproduce the above
*     copyright notice, this list of conditions and the following
*     disclaimer in the documentation and/or other materials provided
*     with the distribution.
*   * Neither the name of the Team-Xmbot-Service-Robot nor the names
*     of its contributors may be used to endorse or promote products
*     derived from this software without specific prior written
*     permission.
*
*   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTOR
S
*   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
*   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS
*   FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE
*   COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIREC
T,
*   INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDIN
G,
*   BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES;
*   LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER
*   CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT
*   LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN
*   ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE
*   POSSIBILITY OF SUCH DAMAGE.
*****/
```

```

"""
*****
*   Software License Agreement (BSD License)
*
*   Copyright (c) 2016, Team-Xmbot-Service-Robot
*   All rights reserved.
*
*   Redistribution and use in source and binary forms, with or without
*   modification, are permitted provided that the following conditions
*   are met:
*
*   * Redistributions of source code must retain the above copyright
*     notice, this list of conditions and the following disclaimer.
*   * Redistributions in binary form must reproduce the above
*     copyright notice, this list of conditions and the following
*     disclaimer in the documentation and/or other materials provided
*     with the distribution.
*   * Neither the name of the Team-Xmbot-Service-Robot nor the names
*     of its contributors may be used to endorse or promote products
*     derived from this software without specific prior written
*     permission.
*
*   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTOR
S
*   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
*   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS
*   FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE
*   COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIREC
T,
*   INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDIN
G,
*   BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES;
*   LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER
*   CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT
*   LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN
*   ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE
*   POSSIBILITY OF SUCH DAMAGE.
*****
"""

```

3. ROS 管理指南

3.1 目录位置

Tip

删除没用的空目录。其他文件的放置位置要合理。

以下路径都是相对于 ROS 包 `<PACKAGE_NAME>` 根目录的。

- C++ 代码头文件要放到 `include/<PACKAGE_NAME>/` 目录下。
- C++ 源文件放到 `src/` 目录下。
- Python 文件要放到 `scripts/` 目录下。
- Python 生成的模块要放到 `src/<PACKAGE_NAME>/` 目录下。
- launch 文件要放到 `launch/` 目录下。
- rviz 可视化配置文件要放到 `rviz/` 目录下。
- urdf 机器人描述文件要放到 `urdf/` 目录下。
- 机器人零件模型要放到 `meshes/` 目录下。
- gazebo 仿真环境配置文件要放到 `world/` 目录下。
- message 文件要放到 `msg/` 目录下。
- service 文件要放到 `srv/` 目录下。
- action 文件要放到 `action/` 目录下。
- 其他配置文件要放到 `config/` 目录下。

3.2 节点通信类型

Tip

根据模块之间通信需求，合理使用。

3.3 元包

Tip

可以将功能相近的若干个包组织在一个元包内。