

# 5、ROS2功能包

## 1、功能包简介

每个机器人可能有很多功能，比如移动控制、视觉感知、自主导航等，如果我们把这些功能的源码都放到一起当然也是可以的，但是当我们想把其中某些功能分享给别人时，就会发现代码都混合到了一起，很难拆分出来。

功能包就是这个原理，我们把不同功能的代码划分到不同的功能包中，尽量降低他们之间的耦合关系，当需要在ROS社区中分享给别人的时候，只需要说明这个功能包该如何使用，别人很快就可以用起来了。

所以功能包的机制，是提高ROS中软件复用率的重要方法之一。

## 2、创建功能包

如何在ROS2中创建一个功能包呢？我们可以使用这个指令：

```
ros2 pkg create <package_name> --build-type <build-type> --dependencies  
<dependencies> --node-name <node-name>
```

ros2命令中：

- **pkg**：表示功能包相关的功能；
- **create**：表示创建功能包；
- **package\_name**：必须项：新建功能包的名字；
- **build-type**：必须项：表示新创建的功能包是C++还是Python的，如果使用C++或者C，那这里就跟ament\_cmake，如果使用Python，就跟ament\_python；
- **dependencies**：可选项:表示功能包的依赖项，C++功能包需包含rclcpp，Python功能包需包含rclpy ,还有其它需要的依赖；
- **node-name**：可选项:可执行程序的名称，会自动生成对应的源文件并生成配置文件；

比如在终端中分别创建C++和Python版本的功能包：

- 切换工作空间的src目录下
- 这里的workspace替换成自己实际的文件夹路径

```
cd workspace/src
```

- 创建C++功能包示例

```
ros2 pkg create pkg_helloworld_cpp --build-type ament_cmake --dependencies  
rclcpp --node-name helloworld
```

- 创建Python功能包示例

```
ros2 pkg create pkg_helloworld_py --build-type ament_python --dependencies rclpy
--node-name helloworld
```

### 3、编译功能包

在创建好的功能包中，我们可以继续完成代码的编写，之后需要编译和配置环境变量，才能正常运行：

- 切换工作空间的目录下
- 编译所有功能包

```
colcon build
```

- 编译指定功能包

```
colcon build --packages-select pkg1 pkg2
```

### 4、带功能包的完整工作空间结构

ROS2工作空间的目录结构如下：

```
workspace --- 自定义的工作空间。
|--- build: 存储中间文件的目录，该目录下会为每一个功能包创建一个单独子目录。
|--- install: 安装目录，该目录下会为每一个功能包创建一个单独子目录。
|--- log: 日志目录，用于存储日志文件。
|--- src: 用于存储功能包源码的目录。
|-- C++功能包
|   |-- package.xml: 包信息，比如:包名、版本、作者、依赖项。
|   |-- CMakeLists.txt: 配置编译规则，比如源文件、依赖项、目标文件。
|   |-- src: C++源文件目录。
|   |-- include: 头文件目录。
|   |-- msg: 消息接口文件目录。
|   |-- srv: 服务接口文件目录。
|   |-- action: 动作接口文件目录。
|-- Python功能包
|   |-- package.xml: 包信息，比如:包名、版本、作者、依赖项。
|   |-- setup.py: 与C++功能包的CMakeLists.txt类似。
|   |-- setup.cfg: 功能包基本配置 文件。
|   |-- resource: 资源目录。
|   |-- test: 存储测试相关文件。
|   |-- 功能包同名目录: Python源文件目录。
```

另外，无论是Python功能包还是C++功能包，都可以自定义一些配置文件相关的目录。

```
|-- C++或Python功能包
|-- launch: 存储launch文件。
|-- rviz: 存储rviz2配置相关文件。
|-- urdf: 存储机器人建模文件。
|-- params: 存储参数文件。
|-- world: 存储仿真环境相关文件。
|-- map: 存储导航所需地图文件。
|-- .....
```

上述这些目录也可以定义为其名称，或者根据需要创建其他一些目录。