

Lesson 12 The Use of Launch File

1.Preface

In the previous lessons, common line is tried to run a new node. However, as the more complex robotic systems are created, it becomes increasingly cumbersome to open multiple terminals and re-enter settings.

Therefore, through creating a launch file, we can start up and configure multiple ROS nodes to be executed as the same time, as well as start more functions. In addition, ROS Master can be started automatically.

2.Basic Grammar of Launch File

2.1 Tag Instruction

Launch file: the configuration and startup of multiple nodes through XML file.

```
<launch>

  <node pkg="turtlesim" name="sim1" type="turtlesim_node"/>

  <node pkg="turtlesim" name="sim2" type="turtlesim_node"/>

</launch>
```

The above is an example of the simplest Launch file. The root element in the Launch file is defined by the <Launch> tag.

The node tag is one of the most commonly used tags in Launch files.

```
<node pkg="package-name" type="executable-name" name="node-name"/>
```

1) <node>: The starting node.

- 2) pkg: The name of package in which the node is located.
- 3) type: The name of the node executable file. If it is written in python, the suffix “.py” needs to be added. If it is written in C++, the name of the executable file can be written directly without the suffix “.cpp”.
- 4) The name of the node when it runs. Each node needs its own unique name. If want to start up two identical files, you can write two different names, for example, start two turtles.

More parameters in <node> can be set in addition to pkg、type、name, as follow:

```
<launch>

  <node

    pkg=""

    type=""

    name=""

    respawn="true"

    required="true"

    launch-prefix="xterm -e"

    output="screen"

    ns="some_namespace"

    args=""

  />

</launch>
```

Parameter	Function
output	By default, information about launching node is stored in the following log file (<code>./ros/log/run_id/node_name-number-stdout.log</code>) by which set the parameters, which can be set here to make the information appear on the screen. For example, output = "screen".
required	Whether all the nodes started by node are shut down
respawn	Whether to automatically restart the node if it shuts down unexpectedly
ns	The node is classified into a different namespace, that is, the prefix given by ns is added in front of the node name

2.2 Parameter Settings

`<param>/:` Set the parameters running in ROS and store them in parameter server.

`<param name="output_frame" value="odom"/>`

1) name: parameter name

2) value: parameter value

`<rosparam>`: load multiple paramters in a parameter file

`<rosparam file="params.yaml" command="load" ns="params" />`

`<arg>`: the local variable in launch file. It is restricted to Launch file.

`<arg name="arg-name" default="arg-value" />`

1) name: parameter name

2) value: parameter value

2.3 Remap nesting

`<remap>`: remap the name of ROS computation graph resource

`<remap from="/turtlebot/cmd_vel"to="/cmd_vel"/">`

1) from: the original name

2) to: the name after mapping

`<include>`: Include other launch files which is similar to header in C programming language.

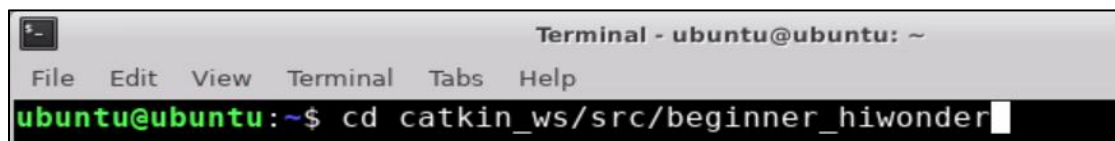
`<include file="$(dirname)/other.launch">`

file: The path to the other Launch files included.

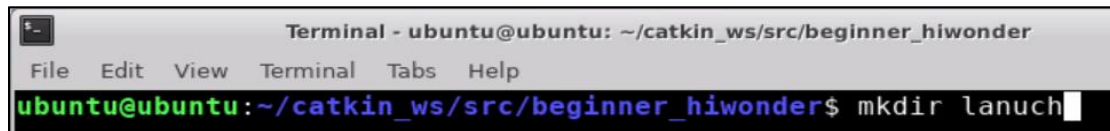
3. Operation Steps

Take the startup of turtle sporting program with a launch file as an exmple, and the steps are as follow:

Step 1: Open the terminal, and then enter “cd catkin_ws/src/beginner_hiwonder” command.

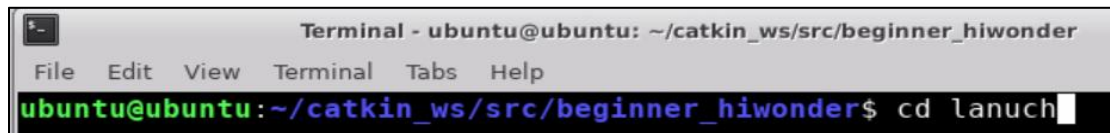


Step 2: enter “mkdir lanuch” command to create a launch folder.



```
Terminal - ubuntu@ubuntu: ~/catkin_ws/src/beginner_hiwonder
File Edit View Terminal Tabs Help
ubuntu@ubuntu:~/catkin_ws/src/beginner_hiwonder$ mkdir lanuch
```

Step 3: Enter “cd lanuch” command to enter the launch folder.

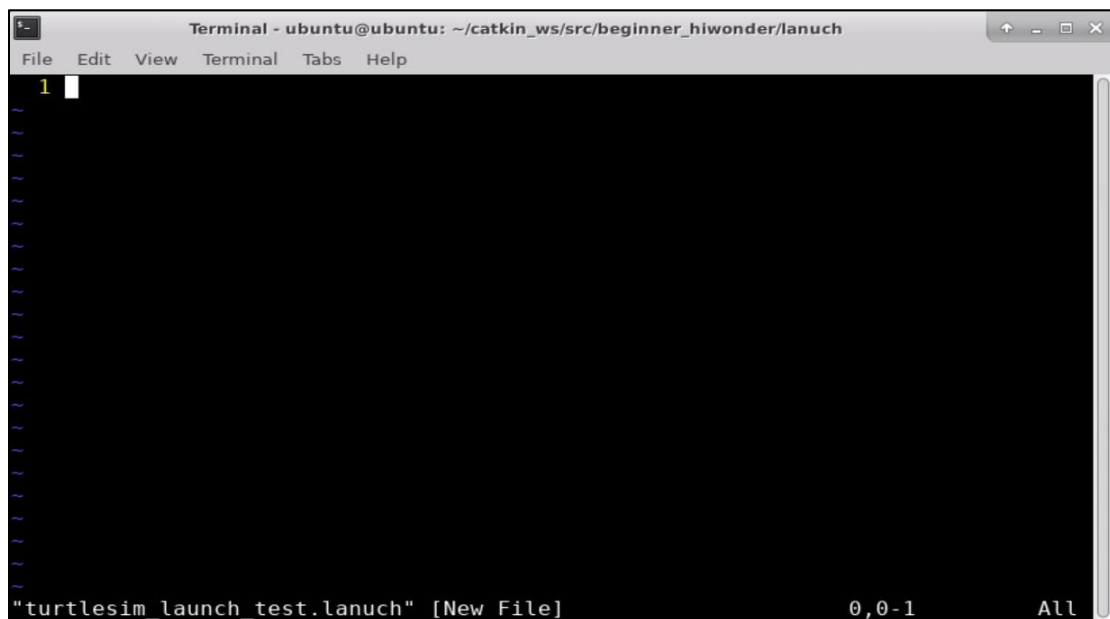


```
Terminal - ubuntu@ubuntu: ~/catkin_ws/src/beginner_hiwonder
File Edit View Terminal Tabs Help
ubuntu@ubuntu:~/catkin_ws/src/beginner_hiwonder$ cd lanuch
```

Step 4: Enter “vi turtlesim_launch_test.lanuch” command to open the file created in step 1 through vi editor.



```
Terminal - ubuntu@ubuntu: ~/catkin_ws/src/beginner_hiwonder/lanuch
File Edit View Terminal Tabs Help
ubuntu@ubuntu:~/catkin_ws/src/beginner_hiwonder/lanuch$ vi turtlesim_launch_test.lanuch
```



```
Terminal - ubuntu@ubuntu: ~/catkin_ws/src/beginner_hiwonder/lanuch
File Edit View Terminal Tabs Help
1
"turtlesim_launch_test.lanuch" [New File] 0,0-1 All
```

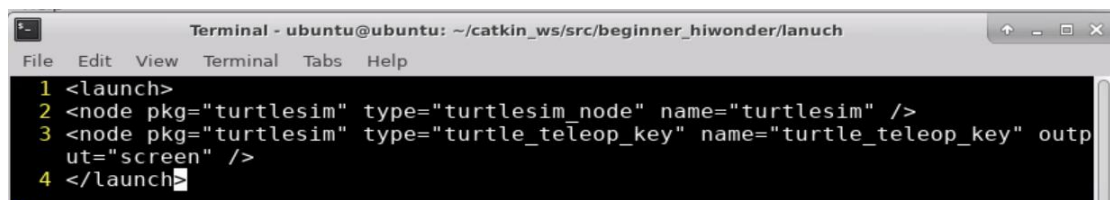
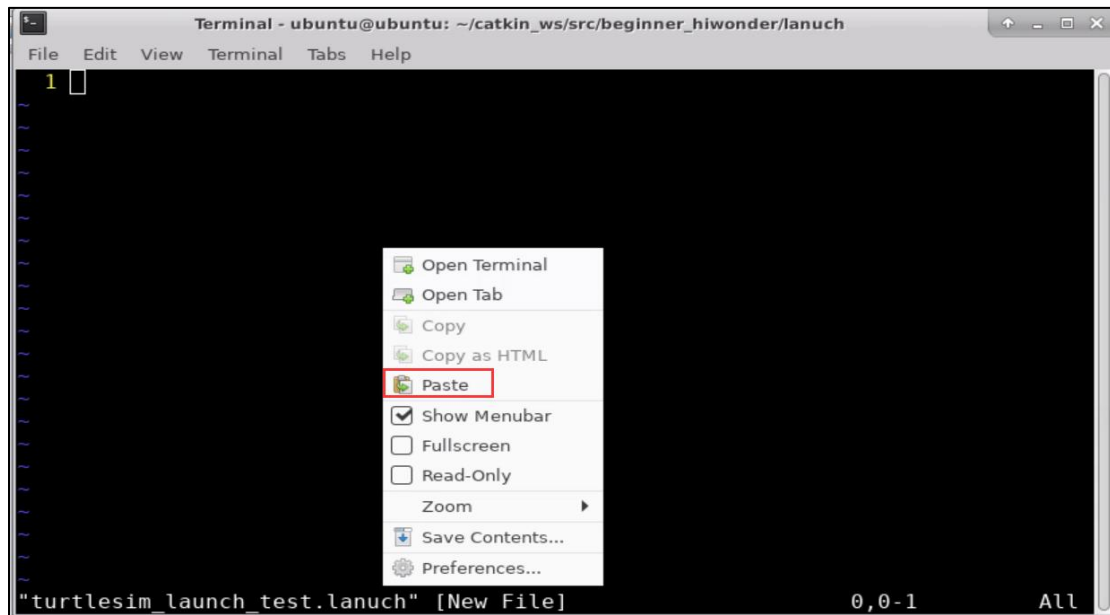
Step 5: Copy the following content into the folder.

```
<launch>

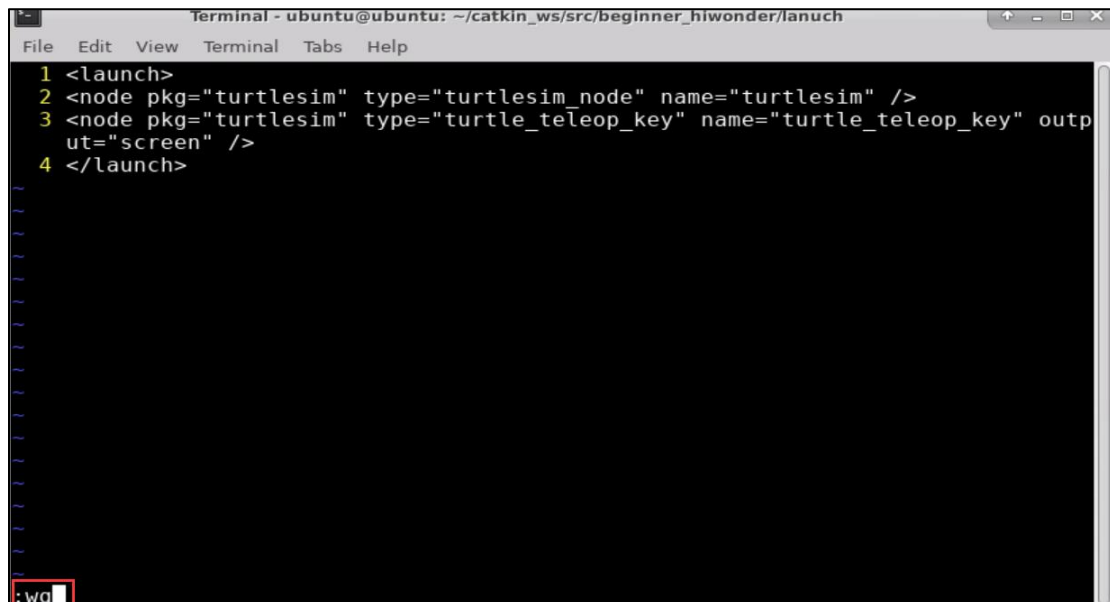
<node pkg="turtlesim" type="turtlesim_node" name="turtlesim" />

<node pkg="turtlesim" type="turtle_teleop_key" name="turtle_teleop_key" output="screen" />
```

</launch>



Step 6: Press “Esc” and enter “:wq” to save and exit.



Step 7: Enter “roslaunch beginner_hiwonder turtlesim_launch_test.lanuch” command to start TurtleSim.

```
Terminal - ubuntu@ubuntu: ~  
File Edit View Terminal Tabs Help  
ubuntu@ubuntu:~$ roslaunch beginner_hiwonder turtlesim_launch_test.lanuch
```

