

launch 파일 작성

01

roslaunch :
여러 노드 한번에
실행

01 roslaunch : 여러 노드 한번에 실행

- Node를 실행시키는 2가지 방법

(실행가능 노드이름 or 실행 파일)

- 첫번째 : `roslaunch <package_name> <executable> [<parameter>]` 을 통한 실행

- `roslaunch`은 단 하나의 Node를 실행(여러 개의 노드를 실행하려면 여러 터미널에서 각각 실행)

- `roslaunch` 실행전에 `roscore`를 실행하여 `rosmaster`를 구동

- 예를 들면, (`ydliar_ros_driver_node`를 실행하지 않았으므로 구동은 되지 않음)

터미널 1 : \$ `roscore`

터미널 2 : \$ `roslaunch limo_base limo_base_node _port_name:= "ttyTHS1"` (← 다양한 파라미터가 있음, `limo_base`. Launch 파일 참조)

터미널 3 : \$ `roslaunch mission_racing obstacle_detect.py`

터미널 4 : \$ `roslaunch mission_racing mission_control.py`

- 필요한 노드들을 터미널마다 열어서 실행하는 것은 불편함

- 파라미터까지 입력하는 것은 더욱 불편함

(→ 두번째 방법 `roslaunch`를 활용)

```
<?xml version="1.0"?>
<launch>
  <!-- ttyTHS1 for NVIDIA nano serial port -->
  <!-- ttyUSB0 for USB serial port -->
  <arg name="port_name" default="ttyTHS1" />
  <arg name="odom_frame" default="odom" />
  <arg name="base_frame" default="base_link" />
  <arg name="use_mcnanu" default="false" />
  <arg name="pub_odom_tf" default="true" />

  <node name="limo_base_node" pkg="limo_base" type="limo_base_node" output="screen" >
    <param name="port_name" value="$(arg port_name)" />
    <param name="odom_frame" value="$(arg odom_frame)" />
    <param name="base_frame" value="$(arg base_frame)" />
    <param name="use_mcnanu" value="$(arg use_mcnanu)" />
    <param name="pub_odom_tf" value="$(arg pub_odom_tf)" />
  </node>
</launch>
```

01 roslaunch : 여러 노드 한번에 실행

- Node를 실행시키는 2가지 방법

- 두번째 : `roslaunch <package_name> <launch_file_name.launch>`

- 한 번에 여러 개의 Node를 동시에 실행

- roscore를 실행하지 않아도 자동으로 rosmaster를 구동해줌

- Launch 파일

- launch 폴더에서 관리

- xml 문법 사용

- launch 파일 안에 다른 launch 파일을 실행하도록 작성 가능

- 예제 : limo_base 노드, xdlidar_ros_driver_node 노드, obstacle_detec 노드, mission_control 노드를
동시 실행하기 위한 launch 파일 작성

- mission_racing 패키지에 launch 폴더를 만들고 관리

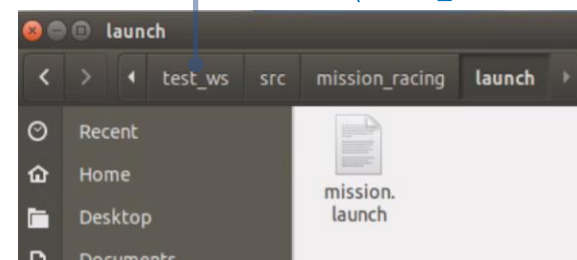
- `$ cd ~/catkin_ws/src/mission_racing`

- `$ mkdir launch` (← launch 디렉토리 생성)

- `$ cd launch`

- `$ touch mission.launch` (← 빈 파일 생성)

제 workspace 이름은 test_ws
라는 이름이지만, 여러분들은
catkin_ws 이름으로 진행하시면
됩니다. (catkin_ws로 나타나면 됨)



01 roslaunch : 여러 노드 한번에 실행

- Node를 실행시키는 2가지 방법

- mission.launch 예제 파일 설명

The screenshot shows a text editor window titled "mission.launch (~/.test_ws/src/mission_racing/launch) - gedit". The editor displays the following XML content:

```
1 <?xml version="1.0"?>
2 <launch>
3
4 <!-- comments -->
5 <include file="$(find limo_base)/launch/limo_base.launch"/>
6 <include file="$(find ydlidar_ros_driver)/launch/X2.launch"/>
7
8 <node pkg="mission_racing" type="obstacle_detect.py" name="obstacle_detect" output="screen"/>
9 <node pkg="mission_racing" type="mission_control.py" name="mission_control" output="screen"/>
10
11 </launch>
```

Annotations and explanations:

- Line 2:** `<launch>` ← launch 파일임을 알려주는 tag
- Line 4:** `<!-- comments -->` ← <!-- 주석작성 --> : 주석 역할
- Line 5:** `<include file="$(find limo_base)/launch/limo_base.launch"/>`
: 다른 roslaunch 파일을 현 파일에서 불러올 때는 include tag에 작성
- Line 6:** `<include file="$(find ydlidar_ros_driver)/launch/X2.launch"/>`
: 다른 package에 있는 launch 파일을 불러올 때 자동으로 경로를 찾아 줌
- Line 8:** `<node pkg="mission_racing" type="obstacle_detect.py" name="obstacle_detect" output="screen"/>`
- Line 9:** `<node pkg="mission_racing" type="mission_control.py" name="mission_control" output="screen"/>`
- Line 11:** `</launch>` ← launch tag의 종료를 알림

General node tag explanation:

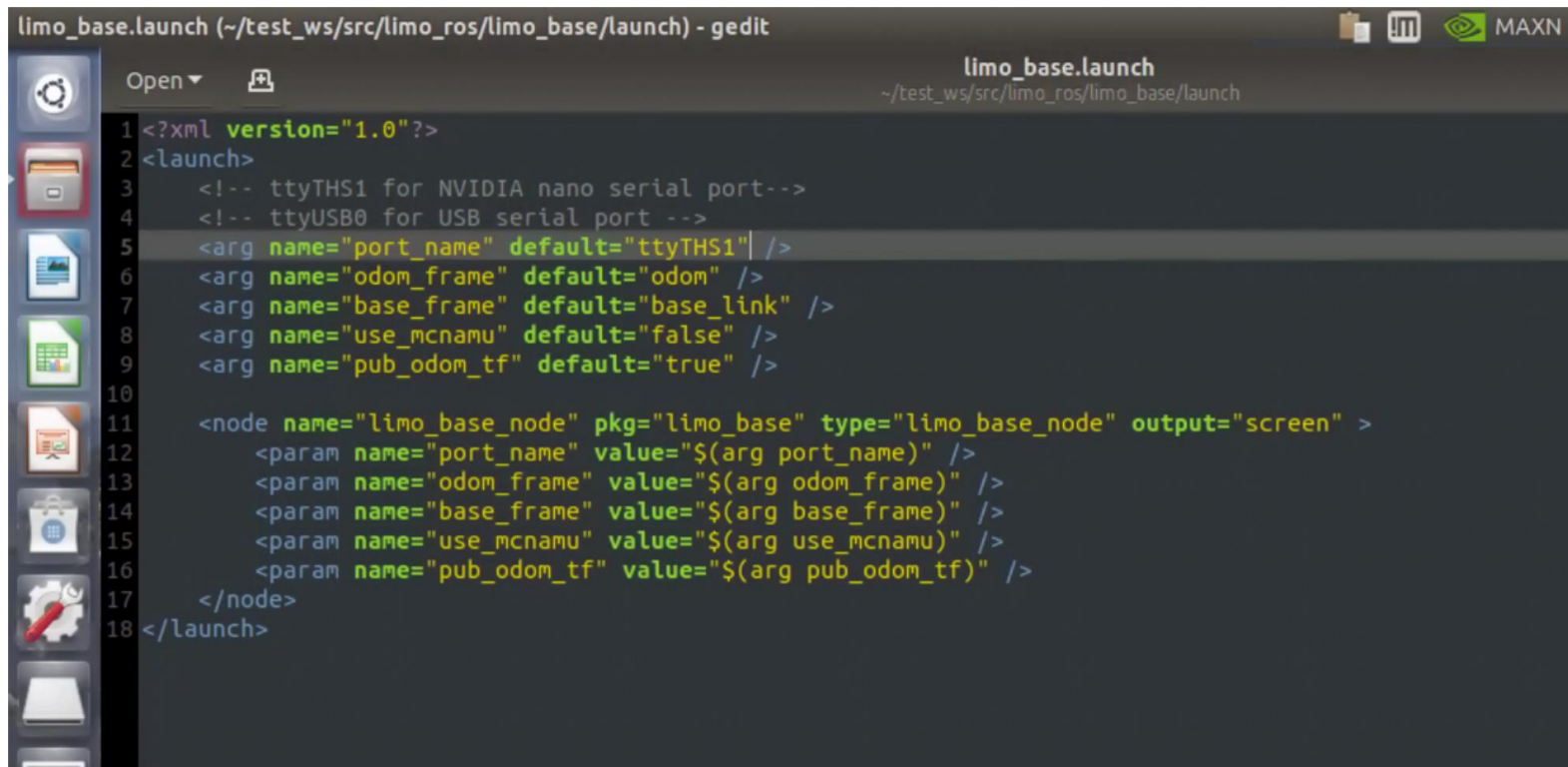
`<node pkg="package_name" type="node_name or 실행파일" name="노드의 실행이름" output="screen"/>`
: 실행해야 할 노드는 node tag에 작성
: output = 기본은 log로 node의 출력을 \$ROS_HOME/log에 작성, "screen" 설정 시 터미널에 출력(생략해도 됨)

`$(find package_name)` : 다른 package에 있는 launch 파일을 불러올 때 자동으로 경로를 찾아 줌

* 그 외 노드에 파라미터를 설정하거나, 인자를 전달하는 등의 명령어가 있으니 <http://wiki.ros.org/roslaunch/XML> 참고

01 roslaunch : 여러 노드 한번에 실행

- Node를 실행시키는 2가지 방법
 - . Limo_base.launch 예제 참고
 - . 다양한 파라미터를 arg 인자 전달을 통해서 설정하면서 노드를 실행



```
1 <?xml version="1.0"?>
2 <launch>
3   <!-- ttyTHS1 for NVIDIA nano serial port-->
4   <!-- ttyUSB0 for USB serial port -->
5   <arg name="port_name" default="ttyTHS1" />
6   <arg name="odom_frame" default="odom" />
7   <arg name="base_frame" default="base_link" />
8   <arg name="use_mcnamu" default="false" />
9   <arg name="pub_odom_tf" default="true" />
10
11   <node name="limo_base_node" pkg="limo_base" type="limo_base_node" output="screen" >
12     <param name="port_name" value="$(arg port_name)" />
13     <param name="odom_frame" value="$(arg odom_frame)" />
14     <param name="base_frame" value="$(arg base_frame)" />
15     <param name="use_mcnamu" value="$(arg use_mcnamu)" />
16     <param name="pub_odom_tf" value="$(arg pub_odom_tf)" />
17   </node>
18 </launch>
```