Navigate launch files with a breeze

It is now easier than ever to channel your pain into creating tools that solve your problems

ROSCon DE 2025

Timotej Gašpar

Stupid question … but can you relate to this?

(I know you can)

What does this included launch file do?

```
base_launch_py = IncludeLaunchDescription(
   PythonLaunchDescriptionSource([
       os.path.join(
           get_package_share_directory('dummy_ros_pkg'),
           'launch',
           'base.launch.py')
   ])
)
```

Steps to open base.launch.py:

- Take a deep breath
- Figure out which package does it belong to
- Look through the file explorer in VScode
- Give up
- Remember that Ctrl+P opens files
- It does not find your file because it’s not in your workspace
- Give up
- Curl into a fetal position
- Try again, this time …

Hi, I am Timo

(who?)

- ROS user and enthusiast for almost 10 years
- Colorful experience with ROS in industry
- With b»robotized (https://www.b-robotized.com) team
  - We do ros2_control and other open source ROS 2 projects
  - We do consulting for our clients
  - … and we developed the b»controlled box (come check it out in the lobby)

The motivation

(Aside from the very picturesque description from the other slide)

- Projects with existing ROS code base
- A very intricate system of launch files
  - Separate for control
  - Separate for robot description
  - Switches whether it runs in simulation or not
  - Which exact robot is it used
  - etc.
- Not inherently wrong!
- Often times one needs to RTFS - Read The F*cking Source

(please also check Kodo-Robotics/launchmap)

“

RTFS is ok ….

… if you can **navigate**
the Fing S **efficiently**

Let’s create a VSCode extension

(I use dark theme, I switch to light only for the screenshot)

- The idea is simple:
  - Click on the launch file name
- VSCode extensions are developed in JavaScript and TypeScript (of which I known none!)
- Let’s take the adequate time to become proficient in it and develop the extension … yes! … yes?

My first project created with a coding agent

(Embrace specification writing!)

- First iteration: pure regular expression
  - Terrible idea

- Next iteration: use the launch Python module
  - Great success

- I wrote detailed instructions of what I want to happen, how and emphasized the importance of testing

Presenting the ROS2 Launch Navigator

(No clever under title here)

Click on imported launch files!

```
base_launch_py = IncludeLaunchDescription(
    PythonLaunchDescriptionSource([
        os.path.join(
            get_package_share_directory('dummy_ros_pkg'),
            'base.launch.py')
    ]),
```

Open base.launch.py (ctrl + click)

Python

XML

```
<launch>
  <!-- Include another launch file -->
  <include file="$(find-pkg-share dummy_ros_pkg)/launch/base.launch.xml"/>
```

Follow link (ctrl + click)

Presenting the ROS2 Launch Navigator

(No clever under title here)

Click on Python nodes!

```
dummy_node = Node(
    package='dum
    executable='dummy_node.py',
    name='dummy_node',
    parameters=[
        {'timer_period': timer_period},
        {'string_to_publish': string_msg}
    ]
```

Open dummy_ros_pkg/dummy_node.py (ctrl + click)

Python

XML

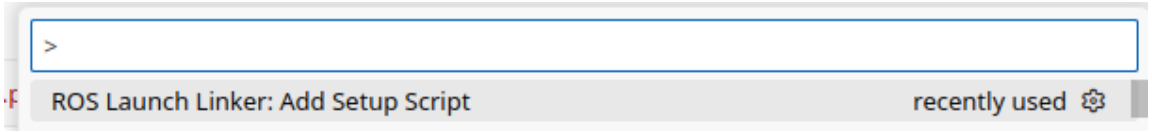
```
<!-- Dummy node configuration --
<node pkg="dummy_ros_pkg" exec="dummy_node.py" name="dummy_node" output="screen
  <!-- Parameters -->
  <param name="timer_period" value="$(var timer_period)"/>
  <param name="string_to_publish" value="$(var string_msg)"/>

  <!-- Topic remapping -->
  <remap from="dummy_topic" to="$(var remap_topic)"/>
```

Follow link (ctrl + click)

Technical side of the extension

(And it’s shortcomings)

- The extension needs the path of the setup.sh files



- It imports the launch file as a Python module and executes all the substitutions
- It then extracts the file name and path
- Matches the file name in the launch file and creates clickable links that open the path

Use the tools to **create tools** that will make your life and the life of others **easier**

Please try it out

(It’s free 😉)

ROS 2 Launch Navigator

https://github.com/b-robotized/ros2-launch-navigator