dfki ai Deutsches Forschungszentrum für Künstliche Intelligenz
German Research Center for Artificial Intelligence

A better launch system for ROS2

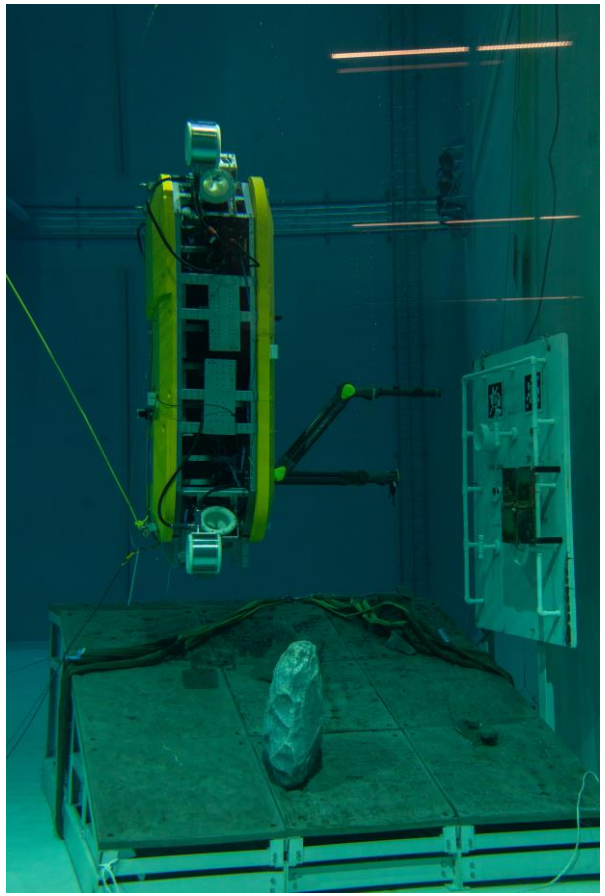
Nikolas Dahn

Clip: Baahubali

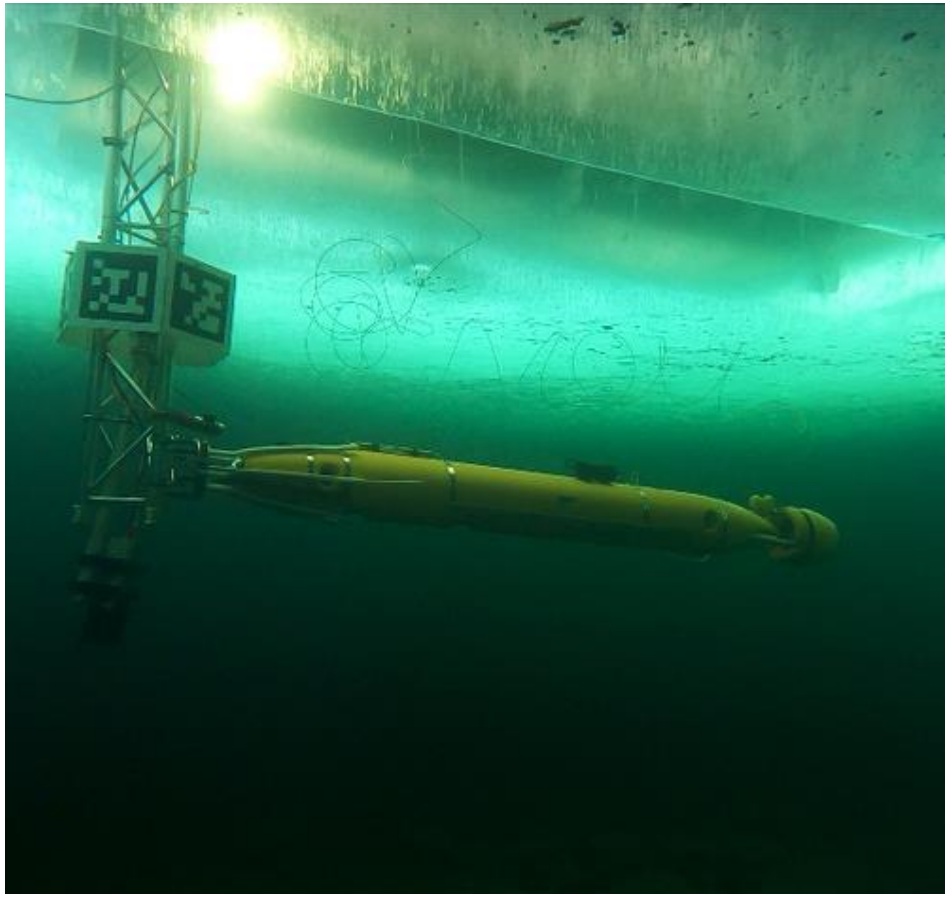
Who is that guy?

- Nikolas Dahn
- Researcher at DFKI
  - German Research Center for Artificial Intelligence
  - Robotics Innovation Center
- I work with robots
  - And I work with ROS2!

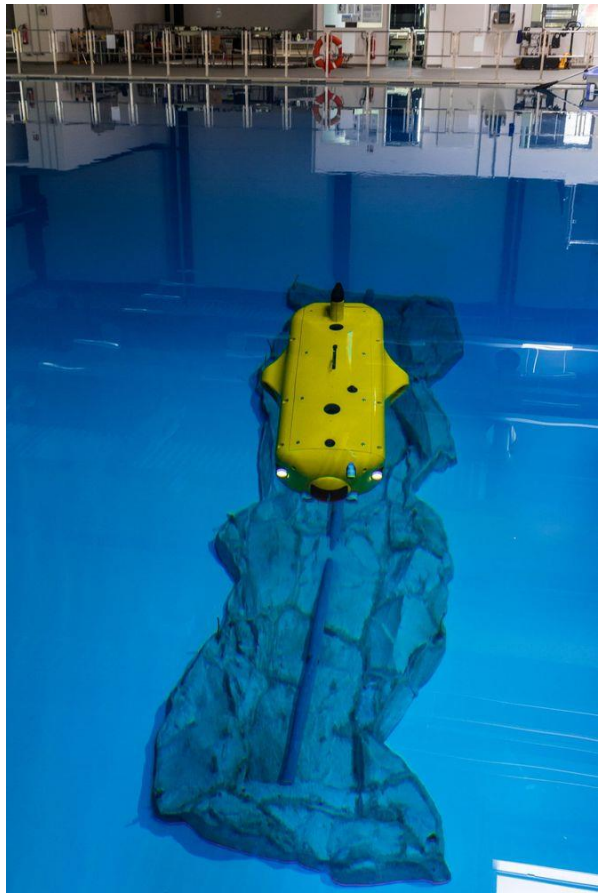
(Some of) the robots we work with



Cuttlefish



DeepLeng



Flatfish

The Launchfiles

- So this is a ROS2 launch file
  - 8 imports
  - 2 different packages
  - 5 different modules
  - Must pass ints as strings
  - Unpythonic
  - Weird constructs
  - Wordy
  - …

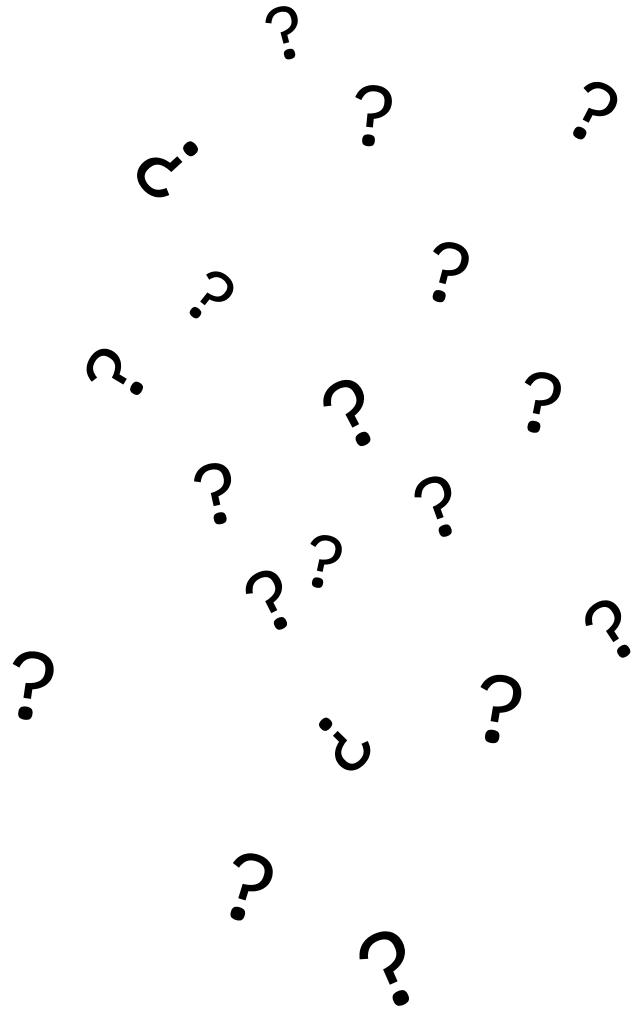
```python
from launch_ros.actions import Node

from launch import LaunchDescription
from launch.actions import DeclareLaunchArgument, ExecuteProcess, TimerAction
from launch.conditions import IfCondition
from launch.substitutions import LaunchConfiguration, PythonExpression


def generate_launch_description():
    new_background_r = LaunchConfiguration('new_background_r')

    new_background_r_launch_arg = DeclareLaunchArgument(
        'new_background_r',
        default_value='200'
    )

    turtlesim_node = Node(
        package='turtlesim',
        executable='turtlesim_node',
        name='sim'
    )

    change_background_r_conditioned = ExecuteProcess(
        condition=IfCondition(
            PythonExpression([
                new_background_r,
                ' == 200',
                ' and ',
                use_provided_red
            ])
        ),
        cmd=[[
            'ros2 param set ',
            '/sim background_r ',
            new_background_r
```

Try out ROS2 they said.
It will be fun they said.

Alternatives?

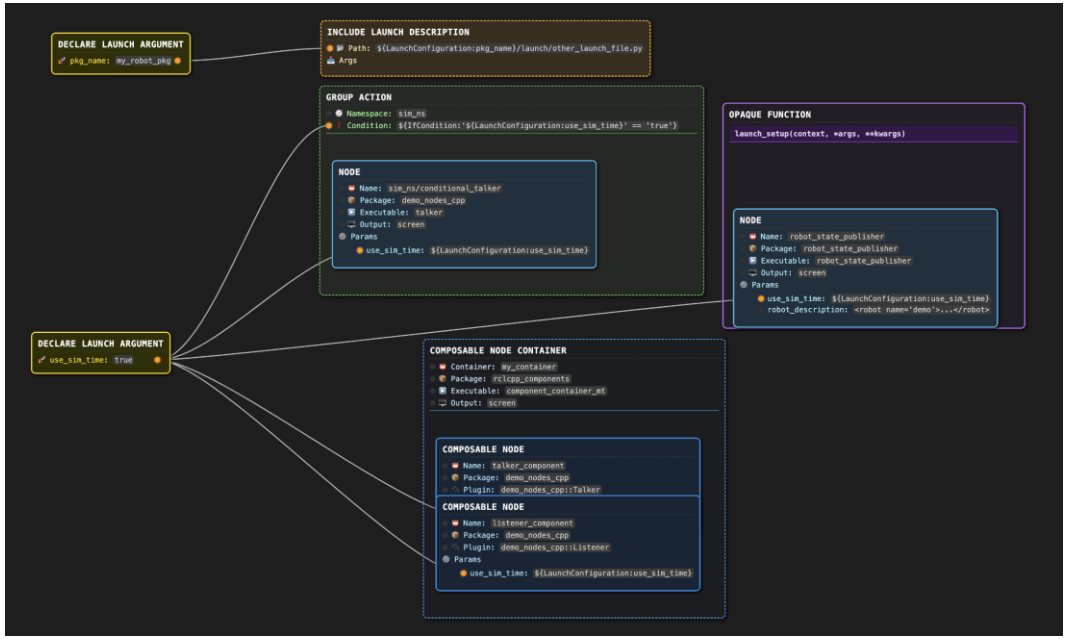
- Some packages to simplify launch files
  - oKermorgant/simple_launch (88 stars)
  - Tacha-S/launch_generator (28 stars)
  - PickNikRobotics/generate_parameter_library (306 stars)

- All of these wrap launch_ros and co., so you still have
  - Non-deterministic execution
  - No direct interaction with nodes
  - No direct access to launch arguments
  - Complex imports across dozens of packages
  - Obscure inner workings
  - Hole-riddled documentation (if at all)



The madness explained (?)

- From the ROS2 launch design document:
  - *„capture the intentions of the user describing the system to be launched“*
  - *„be visualized and statically analyzed without actually launching the described system“*
  - *„visualize and modify the launch description in a WYSIWYG editor “*



LaunchMap, a WYSIWYG editor for launch files.
Use case or symptom?

- But then...
  - Why abuse python as a declarative language?
  - Why not use the inspect module to analyze python launch files?
  - Why is programmatically analyzing and modifying the LaunchDescription so difficult?

Can we do better?

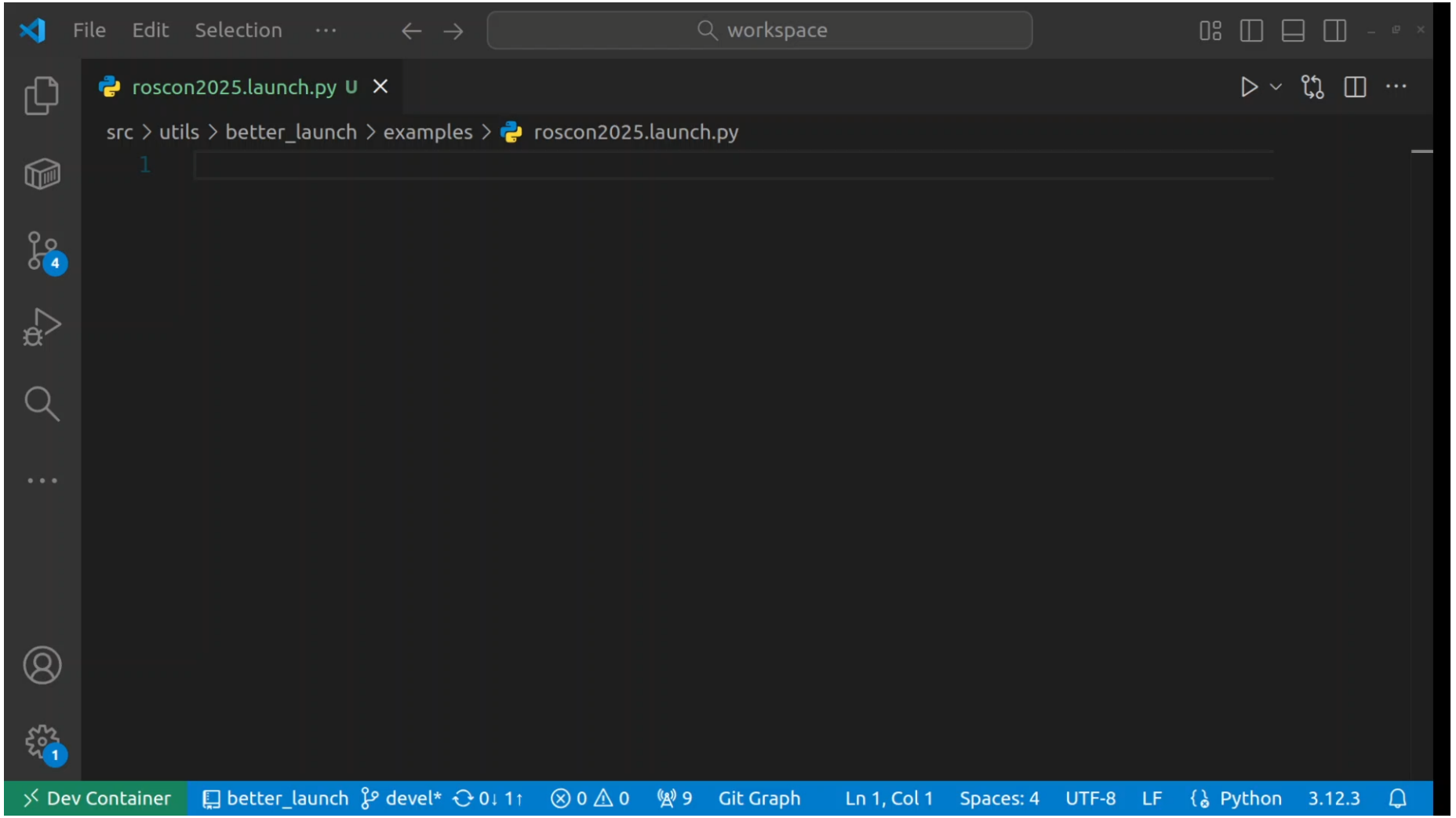
- Enter ***better_launch***
  - Handles the entire launch process
  - NO dependencies on ROS2 launch
  - Handles the entire launch process

- When used in launch files it allows you to
  - *Cover most use cases with one import line*
  - *Use natural types without conversions*
  - *Use natural python syntax*
  - *Deterministic execution*
  - *Directly interact with your nodes*

```python
from better_launch import BetterLaunch, launch_this

@launch_this
def my_start(
    new_background_r: int = 200,
):
    bl = BetterLaunch()

    with bl.group("my_namespace"):
        turtle_node = bl.node(
            package="turtlesim",
            executable="turtlesim_node",
            name="sim",
            params={"background_r": 120}
        )

    if new_background_r == 200:
        time.sleep(2.0)
        turtle_node.set_params({"background_r": 200})

    print(turtle_node.get_published_topics())
```

Better launch better_launch!

One way or another, this will be live

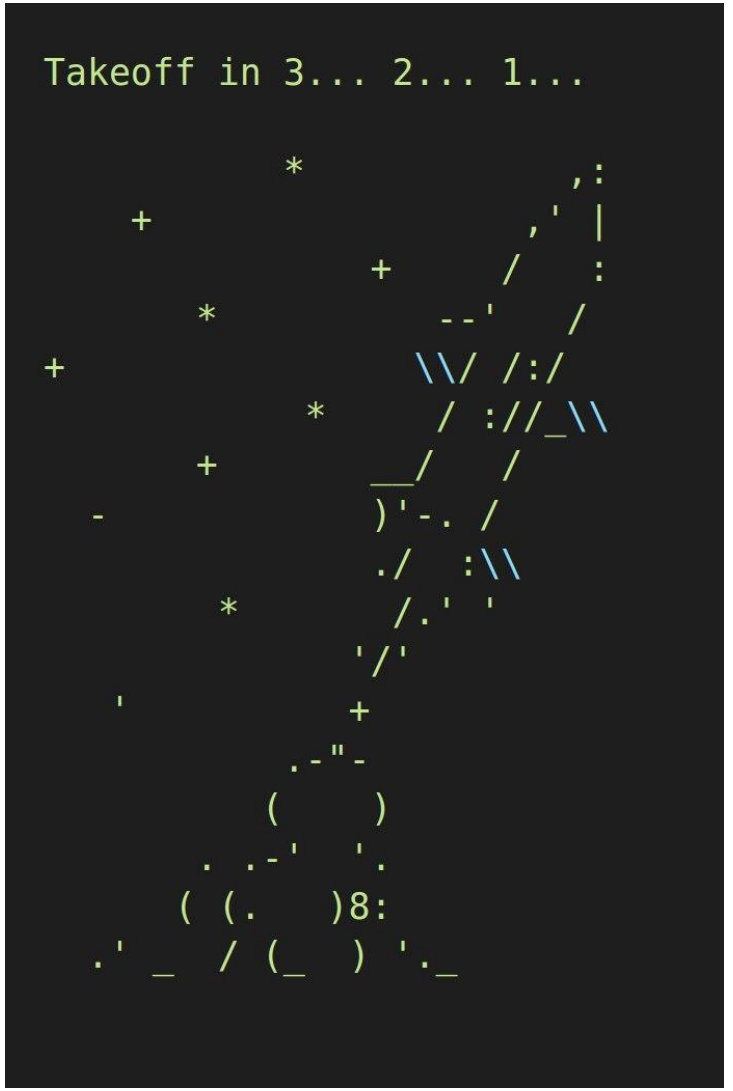


Why you should use `better_launch`

- Write short, intuitive and predictable launch files
  - Include regular ROS2 launch files
  - Be included from regular ROS2 launch files
  - Alternative CLI tool with additional features

- Convenience functions for gazebo, robot descriptions, etc.
- Faster than ros2 launch, similar memory footprint
- No zombie processes

- A **TUI** to interact with nodes!
  - Terminal output formatted for humans



The cherries on top

- Services to stop and restart nodes
- Multiple nodes orchestration
- Declarative TOML launchfiles format
- Documentation will always be a priority

dfki ai Deutsches Forschungszentrum für Künstliche Intelligenz
German Research Center for Artificial Intelligence

Thank you

Happy for your questions!

Find better_launch at
https://github.com/dfki-ric/better_launch

Why better_launch?

- Simple, short and intuitive launch files
- Deterministic execution
- Can include ROS2 launchfiles
- Can be included ***by*** ROS2 launchfiles
- Faster than ros2 launch
- Auto completion and help text for launch args
- Lifecycle management
- Clean shutdown, no zombie processes
- Quickly find parameter files and load them
- Convenience methods for gazebo, rosbags, controllers
- A TUI for managing nodes and node output
- Colorized formatted output made for humans
- Services for stopping and restarting individual nodes
- Similar memory footprint to ros2 launch
- Complete and thorough documentation
- Includes a variety of example launchfiles
- New declarative TOML launchfile format
- …