You're reading the documentation for an older, but still supported, version of ROS 2. For information on the latest version, please have a look at Iron.

Using substitutions

Goal: Learn about substitutions in ROS 2 launch files.

Tutorial level: Intermediate

Time: 15 minutes

Table of Contents

Background

Launch files are used to start nodes, services and execute processes. This set of actions may have arguments, which affect their behavior. Substitutions can be used in arguments to provide more flexibility when describing reusable launch files. Substitutions are variables that are only evaluated during execution of the launch description and can be used to acquire specific information like a launch configuration, an environment variable, or to evaluate an arbitrary Python expression.

This tutorial shows usage examples of substitutions in ROS 2 launch files.

Prerequisites

This tutorial uses the turtlesim package. This tutorial also assumes you are familiar with creating packages.

As always, don't forget to source ROS 2 in every new terminal you open.

Using substitutions

1 Create and setup the package

First, create a new package with the name `launch_tutorial`:

Python package C++ package

Create a new package of build_type `ament_python`:

```
ros2 pkg create --build-type ament_python --license Apache-2.0 launch_tutorial
```

Inside of that package, create a directory called `launch`:

Linux macOS Windows

```
mkdir launch_tutorial/launch
```

Finally, make sure to install the launch files:

Python package C++ package

Add in following changes to the `setup.py` of the package:

```
import os
from glob import glob
from setuptools import find_packages, setup

package_name = 'launch_tutorial'

setup(
    # Other parameters ...
    data_files=[
        # ... Other data files
        # Include all launch files.
        (os.path.join('share', package_name, 'launch'), glob(os.path.join('launch', '*launch.
[pxy][yma]*')))
    ]
)
```

2 Parent launch file

Let's create a launch file that will call and pass arguments to another launch file. This launch file can either be in Python, or in YAML.

To do this, create following file in the `launch` folder of the `launch_tutorial` package.

Python   YAML

Copy and paste the complete code into the `launch/example_main.launch.py` file:

```python
from launch_ros.substitutions import FindPackageShare

from launch import LaunchDescription
from launch.actions import IncludeLaunchDescription
from launch.launch_description_sources import PythonLaunchDescriptionSource
from launch.substitutions import PathJoinSubstitution, TextSubstitution


def generate_launch_description():
    colors = {
        'background_r': '200'
    }

    return LaunchDescription([
        IncludeLaunchDescription(
            PythonLaunchDescriptionSource([
                PathJoinSubstitution([
                    FindPackageShare('launch_tutorial'),
                    'launch',
                    'example_substitutions.launch.py'
                ])
            ]),
            launch_arguments={
                'turtlesim_ns': 'turtlesim2',
                'use_provided_red': 'True',
                'new_background_r': TextSubstitution(text=str(colors['background_r']))
            }.items()
        )
    ])
```

The `FindPackageShare` substitution is used to find the path to the `launch_tutorial` package. The `PathJoinSubstitution` substitution is then used to join the path to that package path with the `example_substitutions.launch.py` file name.

```python
PathJoinSubstitution([
    FindPackageShare('launch_tutorial'),
    'launch',
    'example_substitutions.launch.py'
])
```

The `launch_arguments` dictionary with `turtlesim_ns` and `use_provided_red` arguments is passed to the `IncludeLaunchDescription` action. The `TextSubstitution` substitution is used to define the `new_background_r` argument with the value of the `background_r` key in the `colors` dictionary.

```
launch_arguments={
    'turtlesim_ns': 'turtlesim2',
    'use_provided_red': 'True',
    'new_background_r': TextSubstitution(text=str(colors['background_r']))
}.items()
```

3 Substitutions example launch file

Now create the substitution launch file in the same folder:

Python YAML

Create the file `launch/example_substitutions.launch.py` and insert the following code:

```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():
    turtlesim_ns = LaunchConfiguration('turtlesim_ns')
    use_provided_red = LaunchConfiguration('use_provided_red')
    new_background_r = LaunchConfiguration('new_background_r')

    turtlesim_ns_launch_arg = DeclareLaunchArgument(
        'turtlesim_ns',
        default_value='turtlesim1'
    )
    use_provided_red_launch_arg = DeclareLaunchArgument(
        'use_provided_red',
        default_value='False'
    )
    new_background_r_launch_arg = DeclareLaunchArgument(
        'new_background_r',
        default_value='200'
    )

    turtlesim_node = Node(
        package='turtlesim',
        namespace=turtlesim_ns,
        executable='turtlesim_node',
        name='sim'
    )
    spawn_turtle = ExecuteProcess(
        cmd=[[
            'ros2 service call ',
            turtlesim_ns,
            '/spawn ',
            'turtlesim/srv/Spawn ',
            '"{x: 2, y: 2, theta: 0.2}"'
        ]],
        shell=True
    )
    change_background_r = ExecuteProcess(
        cmd=[[
            'ros2 param set ',
            turtlesim_ns,
            '/sim background_r ',
            '120'
        ]],
        shell=True
    )
    change_background_r_conditioned = ExecuteProcess(
        condition=IfCondition(
            PythonExpression([
                new_background_r,
                ' == 200',
                ' and ',
                use_provided_red
            ])
        ),
        cmd=[[
            'ros2 param set ',
            turtlesim_ns,
            '/sim background_r ',
            new_background_r
        ]],
        shell=True
    )

    return LaunchDescription([
        turtlesim_ns_launch_arg,
        use_provided_red_launch_arg,
        new_background_r_launch_arg,
        turtlesim_node,
        spawn_turtle,
        change_background_r,
        TimerAction(
            period=2.0,
            actions=[change_background_r_conditioned],
        )
    ])
```

The `turtlesim_ns`, `use_provided_red`, and `new_background_r` launch configurations are defined. They are used to store values of launch arguments in the above variables and to pass them to required actions. These `LaunchConfiguration` substitutions allow us to acquire the value of the launch argument in any part of the launch description.

`DeclareLaunchArgument` is used to define the launch argument that can be passed from the above launch file or from the console.

```
turtlesim_ns = LaunchConfiguration('turtlesim_ns')
use_provided_red = LaunchConfiguration('use_provided_red')
new_background_r = LaunchConfiguration('new_background_r')

turtlesim_ns_launch_arg = DeclareLaunchArgument(
    'turtlesim_ns',
    default_value='turtlesim1'
)
use_provided_red_launch_arg = DeclareLaunchArgument(
    'use_provided_red',
    default_value='False'
)
new_background_r_launch_arg = DeclareLaunchArgument(
    'new_background_r',
    default_value='200'
)
```

The `turtlesim_node` node with the `namespace` set to `turtlesim_ns` `LaunchConfiguration` substitution is defined.

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

Afterwards, the `ExecuteProcess` action called `spawn_turtle` is defined with the corresponding `cmd` argument. This command makes a call to the spawn service of the turtlesim node.

Additionally, the `LaunchConfiguration` substitution is used to get the value of the `turtlesim_ns` launch argument to construct a command string.

```
spawn_turtle = ExecuteProcess(
    cmd=[[
        'ros2 service call ',
        turtlesim_ns,
        '/spawn ',
        'turtlesim/srv/Spawn ',
        '"{x: 2, y: 2, theta: 0.2}"'
    ]],
    shell=True
)
```

The same approach is used for the `change_background_r` and `change_background_r_conditioned` actions that change the turtlesim background's red color parameter. The difference is that the `change_background_r_conditioned` action is only executed if the provided `new_background_r` argument equals `200` and the `use_provided_red` launch argument is set to `True`. The evaluation inside the `IfCondition` is done using the `PythonExpression` substitution.

```
change_background_r = ExecuteProcess(
    cmd=[[
        'ros2 param set ',
        turtlesim_ns,
        '/sim background_r ',
        '120'
    ]],
    shell=True
)
change_background_r_conditioned = ExecuteProcess(
    condition=IfCondition(
        PythonExpression([
            new_background_r,
            ' == 200',
            ' and ',
            use_provided_red
        ])
    ),
    cmd=[[
        'ros2 param set ',
        turtlesim_ns,
        '/sim background_r ',
        new_background_r
    ]],
    shell=True
)
```

4 Build the package

Go to the root of the workspace, and build the package:

```
colcon build
```

Also remember to source the workspace after building.

Launching example

Now you can launch using the `ros2 launch` command.

Python   YAML

```
ros2 launch launch_tutorial example_main.launch.py
```

This will do the following:

1. Start a turtlesim node with a blue background
2. Spawn the second turtle
3. Change the color to purple
4. Change the color to pink after two seconds if the provided `background_r` argument is `200` and `use_provided_red` argument is `True`

Modifying launch arguments

Python YAML

If you want to change the provided launch arguments, you can either update them in `launch_arguments` dictionary in the `example_main.launch.py` or launch the `example_substitutions.launch.py` with preferred arguments. To see arguments that may be given to the launch file, run the following command:

```
ros2 launch launch_tutorial example_substitutions.launch.py --show-args
```

This will show the arguments that may be given to the launch file and their default values.

```
Arguments (pass arguments as '<name>:=<value>'):

    'turtlesim_ns':
        no description given
        (default: 'turtlesim1')

    'use_provided_red':
        no description given
        (default: 'False')

    'new_background_r':
        no description given
        (default: '200')
```

Now you can pass the desired arguments to the launch file as follows:

Python YAML

```
ros2 launch launch_tutorial example_substitutions.launch.py turtlesim_ns:='turtlesim3'
use_provided_red:='True' new_background_r:=200
```

Documentation

The launch documentation provides detailed information about available substitutions.

Summary

In this tutorial, you learned about using substitutions in launch files. You learned about their possibilities and capabilities to create reusable launch files.

You can now learn more about using event handlers in launch files which are used to define a complex set of rules which can be used to dynamically modify the launch file.