ROS: Modularity of ROS and roslaunch

The objective of this lab is to learn how to use roslaunch and benefit from the modularity of ROS.

Setting up your computer

Open terminal window.

Clone your git repository to your home folder:

```
$ git clone URL_for_<yourname>-rtech_repository
```

Use **`ls`** to confirm that ***<yourname>-rtech*** has been downloaded to your home folder**.**

ROS tutorials about Unified Robot Description Format (URDF)

In this lab you will be combining knowledge from all the previous computer labs to make your R2D2 robot move via keyboard teleoperation.

First, test that turtlesim is still working properly on your computer. Open two terminal windows to type

```
$ rosrun turtlesim turtlesim_node
```

```
$ rosrun turtlesim turtle_teleop_key
```

If everything works, shut down both nodes.

Now, confirm that you can successfully load and visualise your R2D2 xacro-file from the previous lab. Do you remember how to do that?

Next, let's update your R2D2 description by adding a new empty link to your xacro-file. Name this link `base_footprint`.

Create a new fixed joint from `base_footprint` to `base_link` so that `base_footprint` is positioned right under `base_link` but on the same plane as the bottom of wheels.

Now, when you visualise the modified robot description in RViz and set ***Fixed Frame*** in ***Global Options*** to `base_footprint`, your R2D2 should be positioned right on top of the grid plane. By using ***Save Config As*** menu option in RViz, save this RViz configuration file in `my_r2d2/config` folder as `r2d2.rviz`.

Next, let's describe an easy-to-use launch file for `my_r2d2` package. Create a folder called `launch` in `my_r2d2`, use your favourite text editor to create a file named `r2d2.launch`, and save this file inside the `my_r2d2/launch/` folder.

Insert the following content to your launch-file:

```
<launch>
  <param name="robot_description" command="$(find xacro)/xacro.py '$(find
my_r2d2)/urdf/r2d2.urdf.xacro'"/>
  <node name="robot_state_publisher" pkg="robot_state_publisher" type="state_publisher"/>
  <node name="joint_state_publisher" pkg="joint_state_publisher" type="joint_state_publisher"/>
  <node name="rviz" pkg="rviz" type="rviz" args="-d $(find my_r2d2)/config/r2d2.rviz"/>
</launch>
```

Shut down any ROS nodes and roscore you may be running. Now type:

```
$ roslaunch my_r2d2 r2d2.launch
```

What do you see?

Clone the `r2d2_navigator` package (https://github.com/ut-ims-robotics/r2d2_navigator) to your catkin workspace. Compile your workspace.

Run the node called `fake_r2d2_controller` from `r2d2_navigator` package and use `rosnode` to determine, which topics `fake_r2d2_controller` subscribes to.

As we know, ROS is very modular, so let's try to control our R2D2 by using the `turtle_teleop_key` node from `turtlesim` package. Run `turtle_teleop_key` and determine, which topics it is publishing to.

By using launch-files, find a method for achieving a situation where the `fake_r2d2_controller` subscribes to messages from `turtle_teleop_key`. (*Hint: <remap>*)

Your end result should be that by using no more than 2 terminal windows, you are able to open your R2D2 model in RViz (Fixed Frame: odom) and move the R2D2 around using `turtle_teleop_key`.

Show this result to the instructor.

CLEAN UP YOUR WORKSPACE

NB! Before you leave the lab, make sure you have pushed all the files in your catkin workspace to your git cloud service.

In terminal, **cd** to **<yourname>-rtech**

Type

```
git config user.email "youremail@example.com"
```

Type

```
git status
```

You should now see all the new and modified files in red.

Prepare the relevant files for the commit.

```
git add file_name_in_red1 file_name_in_red2
```

When you now type

```
git status
```

you should see all the added files in green. You are now ready to commit changes. Type

```
git commit –m "Insert a brief explanation"
```

Your changes have now been committed but not yet uploaded to the cloud. To upload your files, type

```
git push
```

In your web browser, **verify that all the files** have been uploaded to the **<yourname>-rtech** repository.

Double check!! Did you also get files from **`r2d2_navigator`** package to your git cloud storage?

Delete the ***<yourname>-rtech*** folder and any other files you created from the lab's computer.