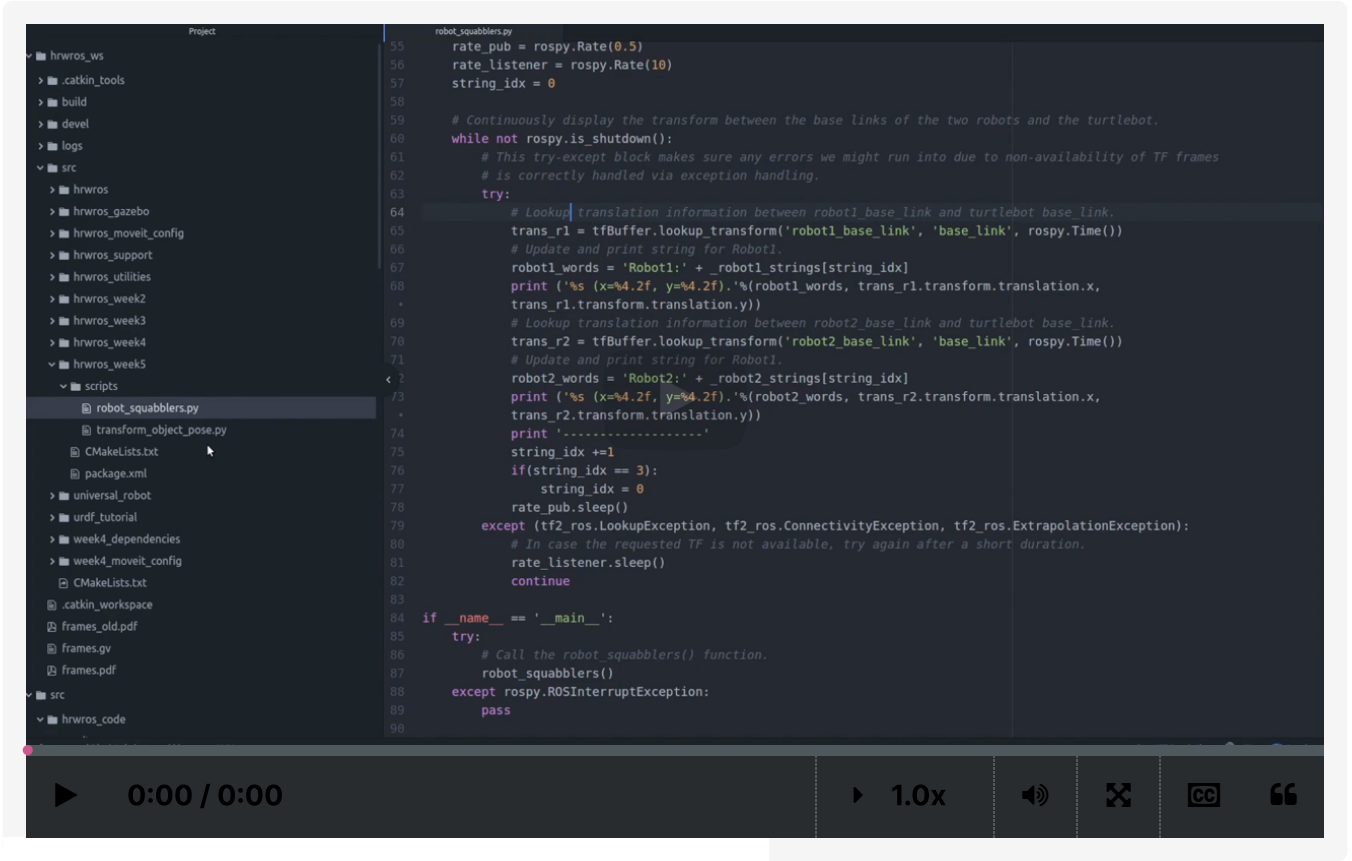
After taking an in-depth look at the squabbling robots last video lecture, we will now dive deeper into the `lookup_transform` API.

5.3.5 tf2_ros API lookup_transform



Subtitles (captions) in other languages than provided can be viewed at YouTube. Select your language in the CC-button of YouTube

Script

This time, we will be looking at the `transform_object_pose.pt` script, you can also find it on the scripts folder of `hrwros_week5` package

- Again, we see a list of necessary imports, which includes rospy, the tf2_ros module, and a bunch of necessary message modules.
- We also create a new tf buffer and listener, like in the previous video.

Note: On the files you will get, the tf Buffer and TransformListener are created after the initialization of the ROS Node (Line 77)

- In the callback function for the logical camera, we can see that the logical camera provides us with a lot of information about the pose of an object in the reference frame of the logical camera itself.
- The API transform can only work with poses that have a timestamp, because the transform might be influenced by a moving robot part. You can't always use the current pose of a link, sometimes you have to look at the past when the pose was generated!
- Thus, the script creates a new header and time stamp information.
- After that, we update the pose information using information from the logical camera.
- Now, we have everything we need for the transform API. We will transform the pose of the object from the camera reference frame to the world reference frame.

Prepare everything needed for the script to work

Make sure the factory environment simulation is running. Don't forget to verify that all robots have shown up!

```
$ roslaunch hrwros_gazebo hrwros_environment.launch
```

- Robot 1 is currently in the FOV of the camera. We don't want this, so let's use MoveIt Commander in a new CCS to move it out of the way.

```
$ rosrun hrwros_week4 hrwros_moveit_commander_cmdline
```

- R1Home is a suitable position for it.

```
> use robot1
> go R1Home
```

Now the arm is out of the way, let's spawn in an object on the conveyor belt!

- In a new, sourced, CCS shell, execute the following command:

```
$ rosservice call /spawn_object_once
```

- Now you can see a white box object on the conveyor belt near robot 1 in Gazebo. Let's find out what the logical camera can see:

```
$ rostopic echo /hrwros/logical_camera
```

- Search for a model of the type "`object`" in the output in the console.
    - It will tell you the position and rotation of the withe box with respect to the camera frame.

Run the transform_object_pose script

- Switch to a new, sourced CCS shell, and run the script:

```
$ rosrun hrwros_week5 transform_object_pose.py
```

- You can now see the output of the script in the terminal.
    - The first pose is the pose of the object in the world reference frame.
    - The second pose is the pose of the object in the camera reference frame.
    - Using the right-hand rule, you can verify the poses are correct!
    - Again, during this course, we will take care of the rotations for you.

Question 1

1 point possible (ungraded)

For the transform API to function correctly, there is a package/python module dependency. Which one is it?

- ◯ tf2_geometry_msgs
- ◯ geometry_msgs
- ◯ gazebo_msgs

Submit