following topics:

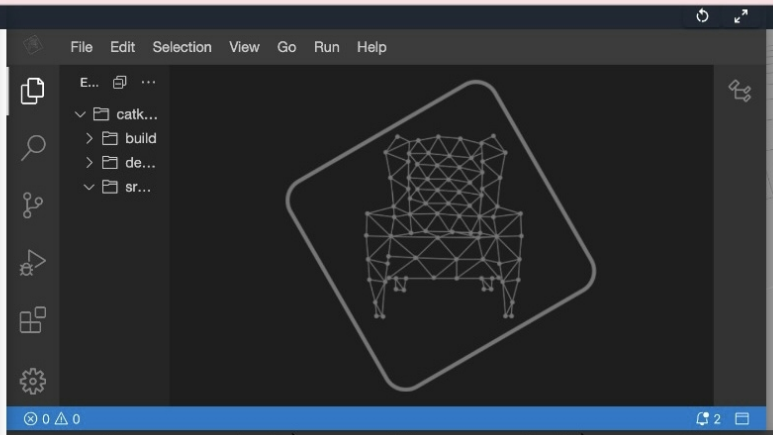
- How to navigate through a Linux filesystem
- How to interact with a Linux filesystem
- How to edit files using the Shell (vi editor)
- Manage access to files (Permissions)
- Create simple Linux programs (Bash Scripts)
- Manage execution of Linux programs (Processes)
- How to connect to the remote computer of a robot (ssh)

1.4 How will you learn all this?

You will learn through hands-on experience from day one! In the Robot Ignite Academy, we strongly believe that the best way to learn is by practicing, practicing, and then... practicing some more!

Since this is a Linux Course, we are going to focus on the Linux utilities, leaving ROS aside. But during the course, you are going to still be able to interact with the simulation environment as well. Specifically, you are going to work with a BB8 robot. Isn't that cool?

BB-8:



```
publishing and latching message. Press ctrl-C to terminate
^Cuser:/home/simulations/public_sim_ws/src/all/ros_basics_examples/linux_demo$ user:/home/simulations/publi
sim_ws/src/all/ros_basics_examples/linux_demo$ ./demo.sh forward
bash: user:/home/simulations/public_sim_ws/src/all/ros_basics_examples/linux_demo$: No such file or directo
ry
user:/home/simulations/public_sim_ws/src/all/ros_basics_examples/linux_demo$ publishing and latching messag
e. Press ctrl-C to terminate^C
user:/home/simulations/public_sim_ws/src/all/ros_basics_examples/linux_demo$ ./demo.sh stop
publishing and latching message. Press ctrl-C to terminate
^Cuser:/home/simulations/public_sim_ws/src/all/ros_basics_examples/linux_demo$ ./demo.sh forward
publishing and latching message. Press ctrl-C to terminate
^Cuser:/home/simulations/public_sim_ws/src/all/ros_basics_examples/linux_demo$ ./demo.sh rotate
publishing and latching message. Press ctrl-C to terminate
```

app.theconstructsim.com/Desktop/

Update

are represented by the character **x**, have been added to all the permission groups.

Excellent! But you may be asking yourself... what if I only want to modify the permissions for 1 of the groups? Or what if I want to remove a permission instead of assigning it? And those are very good questions! Let's go step by step.

The structure of the **chmod** command goes as follows:

In [ ]:

```
chmod  <groups to assign the permissi
```

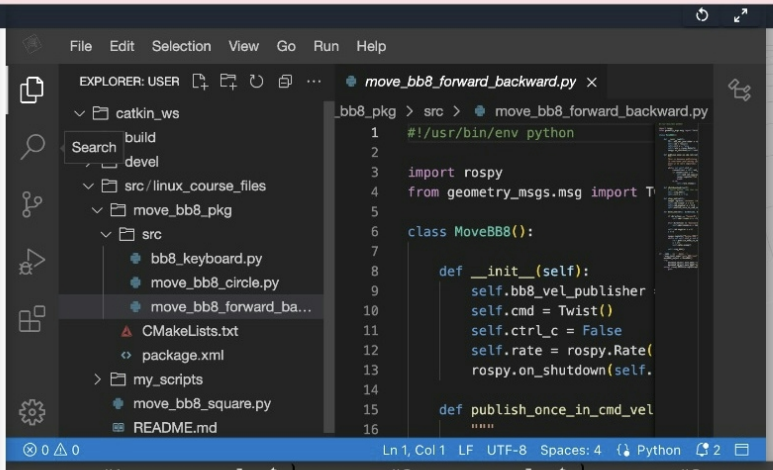
We already know the 2 last parameters. As for the groups, you can specify them using the following flags:

- **u**: Owner
- **g**: Group
- **o**: Others
- **a**: All users. For all users, you can also leave it blank, as we did in the example command you executed before.

Let's write down some examples in order to better understand this. So, for instance, if we wanted to give write permissions only to the **group**, how would the command be? Can you guess? If not, you can find the solution right below:

In [ ]:

```
chmod g+w move_bb8_square.py
```



#1 #2 #3 #4

```
user:~$ cd /home/user/catkin_ws/src/linux_course_files/move_bb8_pkg/my_scripts/
bash: cd: /home/user/catkin_ws/src/linux_course_files/move_bb8_pkg/my_scripts/: No such file or dir
ectory
user:~$ ls -la
total 100
drwxr-xr-x 12 user user 4096 Sep 29 15:21 .
drwxr-xr-x  1 root root 4096 Sep 29 14:59 ..
-rw-r--r--  1 user user 5609 Sep 29 15:21 .__bashrc
-rw-r--r--  1 user user 3620 Sep 29 15:21 .bash_aliases
-rw-------  1 user user 3338 Sep 29 15:23 .bash_history
-rwxrwxrwx  1 user user 5609 Sep 29 08:56 .bashrc
-rw-r--r--  1 user user  225 Sep 29 14:43 .bashrc_bridge
-rw-r--r--  1 user user  116 Sep 29 14:43 .bashrc_ros1
-rw-r--r--  1 user user  240 Sep 29 14:43 .bashrc_ros2
drwxr-xr-x  3 user user 4096 Sep 29 11:21 .cache
```

app.theconstructsim.com/Desktop/

Update

/home/user/catkin_ws/src/linux_course_files/move_bb8_pkg/src

In the previous example, we went directly to a path. This is because we already knew where we wanted to go, so we could use a shortcut, moving directly to the folder we wanted. But this is not always the case. Many times, you will have to move folder by folder, checking what each folder contains. Sometimes, you will even need to go back to the previous folder. All of this can be achieved using the **cd** command.

For instance, imagine that we now need to move back to the **move_bb8_pkg** folder. How would you do that? Any idea? Maybe you are thinking about using a command like this one:

In [ ]:

```
cd /home/user/catkin_ws/src/linux_cou
```

Well, that is actually correct. At least, it will do the job. But let me tell you that there's a much easier way to achieve the same. Let's try the following:

▶ Execute in Shell #1

In [ ]:

```
cd ../
```

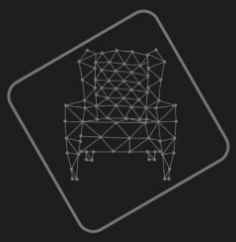
```
user:~/catkin_ws/src/linux_course_files/move_bb8_pkg/src$ cd ../
user:~/catkin_ws/src/linux_course_files/move_bb8_pkg$
```

And now, let's move back to the **src** folder.

File Edit Selection View Go Run Help

EXPLORER: U...

- catkin_ws
  - build
  - devel
  - src / linux_course_files
    - move_bb8_pkg
      - src
      - CMakeLists.txt
      - package.xml
    - my_scripts
      - move_bb8_square.py
      - README.md



#1 #2 #3 #4

```
user:~$ cd /home/user/catkin_ws/src/linux_course_files/move_bb8_pkg/src/
user:~/catkin_ws/src/linux_course_files/move_bb8_pkg/src$ python bb8_keyboard.py

Reading from the keyboard  and Publishing to Twist!
---------------------------
Moving around:
   u    i    o
   j    k    l
   m    ,    .
For Holonomic mode (strafing), hold down the shift key:
---------------------------
   U    I    O
   J    K    L
   M    <    >
t : up (+z)
```

2 - Linux Essentials

Linux for Robotics

100%

app.theconstructsim.com/Desktop/

- **Foreground processes** : These are initialized and controlled through a terminal session. In other words, there has to be a user connected to the system to start such processes; they haven't started automatically as part of the system functions/services.
- **Background processes**: These are processes not connected to a terminal. This means that they don't expect any user input.

4.1.1 Visualize processes

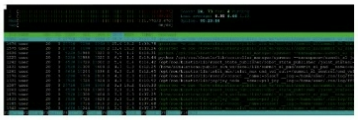
There exist several different commands that will allow you to visualize the running processes on the system. In this course, though, we are going to focus on 2 of them: **htop** and **ps**.

▶ Execute in Shell #1

In [ ]:

```
htop
```

You will see something similar to this:



The **htop** command is an improved version of **top**

File Edit Selection View Go Run Help

move_bb8_forward_backward.py

```
#!/usr/bin/env python

import rospy
from geometry_msgs.msg import Twist

class MoveBB8():

    def __init__(self):
        self.bb8_vel_publisher = rospy.Publisher(
        self.cmd = Twist()
        self.ctrl_c = False
        self.rate = rospy.Rate(1)
        rospy.on_shutdown(self.shutdownhook)

    def publish_once_in_cmd_vel(s
        """
```

Select Indentation

Ln 1, Col 1 LF UTF-8 Spaces: 4 Python

```
  PID USER      PRI  NI  VIRT   RES   SHR S CPU% MEM%   TIME+  Command
  433 user       20   0 1161M 41208 19524 S  0.0  0.3  0:01.21 node /usr/local/share/shell/demo/app.js
 4687 user       20   0 22836  3168  2744 R  0.0  0.0  0:00.01 htop
 4687 user       20   0 22836  3168  2744 R  0.0  0.0  0:00.04 htop
  433 user       20   0 1161M 41692 19524 S  0.0  0.3  0:01.23 node /usr/local/share/shell/demo/app.js
  432 user       20   0 1161M 41692 19524 S  0.0  0.3  0:01.21 node /usr/local/share/shell/demo/app.js
  924 user       20   0 28040  2692  2568 S  0.0  0.0  0:00.01 node /usr/local/share/shell/demo/app.js
F1Help  F2Setup F3Search F4Filter F5Tree  F6SortBy F7Nice -F8Nice +F9Kill  F10Quit
```

4 - Advanced Utilities II

Linux for Robotics

100%