# TP1 - My Teleop

**Polytech Angers** - Mobile Robotics



Create a package named `tp1`. The next nodes are to be done in this package. Check the First Node document to do that.

# My teleop

We want to create a node that allows to control the `turtlebot` with the keyboard ('o' and 'l' to move forward/backward and 'k' and 'm' to rotate left/right).

Create an executable `my_teleop.py` inside the tp1 package. The following command should start the node:

```
docker@ros2:~/wdir$ ros2 run tp1 my_teleop.py
```

## Expected behavior

Here is the expected output:

```
docker@ros2:~/wdir$ ros2 run tp1 my_teleop.py
[INFO] [1679392751.167146941] [my_teleop]: Starting the teleop node
[INFO] [1679392751.167595062] [my_teleop]: The commands:
        'o' to move forward
        'l' to move backward
        'k' to turn left
        'm' to turn right
        'q' to quit
[INFO] [1679392752.308008329] [my_teleop]: Stopping the node...
```

To do such node you will face two issues:

- To get an input from the keyboard without waiting for the `Enter key` to be pressed;
- To do a loop in a node without blocking everything.

The following provides you tools to solve those issues.

## Get the keyboard input without pressing enter

The following code returns the keyboard input without waiting for "enter" to be pressed (this is a Linux solution that should work well on the docker image)

```
import sys, tty, termios

def getkey():
    """ To get a keyboad key (only one char)
        without waiting for Enter to be pressed
    """
    fd = sys.stdin.fileno()
    old_settings = termios.tcgetattr(fd)
    try:
        tty.setraw(sys.stdin.fileno())
        ch = sys.stdin.read(1)
    finally:
        termios.tcsetattr(fd, termios.TCSADRAIN, old_settings)
    return ch
```

## Run an infinite loop inside a node

To run a loop inside a node, you can use a thread so that the loop will not block the execution of the node (sending and getting ROS2 messages).

```
import threading

def myloop():
    stop = False
    while not stop:
        # do something
        pass

my_thread = threading.Thread(target=myloop)  # the target is the function
to call when running the thread
my_thread.start()  # start the thread
```

For this exercise it seems to be easier if the thread variable is a member of your class node...

## Improvement

Add a new functionality to your node: when pressing the `c` key on the keyboard it should clear the drawn path on the turtle bot interface. **Do this with python code** : do not call the ros2 bash command with something like `os.system("my ros2 command")`... This can help : ros2 documentation: service in python

**Note**: Do not rewrite everything, this can be done by adding three lines of codes in your previous version...