4、Control robot movement

4.1 Experimental goal

This course mainly learns how to control the robot's functions such as forward and backward, left and right translation, and left and right rotation.

4.2 Experiment preparation

The functions of the Muto hexapod robot Python library involved in this course are as follows:：

forward(step)： Walk forward, step is the width of the step, the larger the effective value, the wider the width of each step. The value range of step is **[10, 25]**.

back(step)： Go backward, step is the width of the step, the larger the effective value, the wider the width of each step. The value range of step is **[10, 25].**

left(step)： Left translation, step is the step width, the larger the effective value, the larger the width of each step. The value range of step is **[10, 25].**

right(step)： Right translation, step is the step width, the larger the effective value, the larger the width of each step. The value range of step is **[10, 25].**

turnleft(step)： Rotate left, step is the step width, the larger the effective value, the larger the width of each step. The value range of step is **[10, 25]**.

turnright(step)： Rotate right, step is the step width, the larger the effective value, the larger the width of each step. The value range of step is **[10, 25]**.

stop()： Stop movement.

reset(): Reset the robot and the robot will stand in the initial posture.

speed(level): Control the speed of robot movement. The value range of level is **[1-5]**. The larger the value, the faster the movement frequency.

4.3 Experiment procedure

Open the jupyterLab client and find the code path:

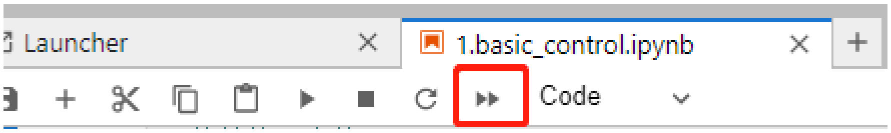
```
muto/Samples/Control/4.movements.ipynb
```

By default g_ENABLE_CHINESE=False, if you need to display Chinese, please set g_ENABLE_CHINESE=True

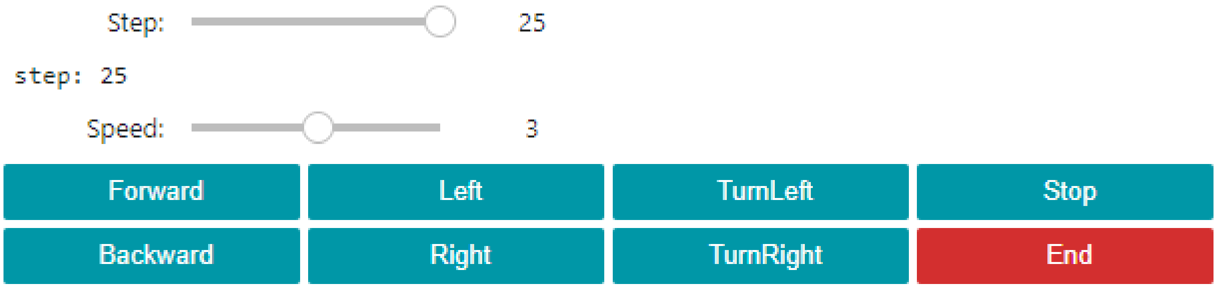
```
# 中文开关，默认为英文 Chinese switch. The default value is English
g_ENABLE_CHINESE = False

Name_widgets = {
    'Stop': ("Stop", "停止"),
    'Forward': ("Forward", "前进"),
    'Backward': ("Backward", "后退"),
    'Left': ("Left", "左平移"),
    'Right': ("Right", "右平移"),
    'TurnLeft': ("TurnLeft", "向左转"),
    'TurnRight': ("TurnRight", "向右转"),
    'End': ("End", "结束")
}
```

Click to run all cells, and then scroll to the bottom to see the generated controls.



Click different buttons to correspond to different functions.



```
step: 25
```

Every time a button is clicked, the corresponding function will be executed. Key event processing:

```
# 按键按下事件处理   Key press event processing
def on_button_clicked(b):
    ALL_Uncheck()
    b.icon = 'check'
    with output:
        print("Button clicked:", b.description)
    if b.description == Name_widgets['Stop'][g_ENABLE_CHINESE]:
        g_bot.stop()
        b.icon = 'uncheck'
    elif b.description == Name_widgets['Forward'][g_ENABLE_CHINESE]:
        g_bot.forward(g_step)
    elif b.description == Name_widgets['Backward'][g_ENABLE_CHINESE]:
        g_bot.back(g_step)
    elif b.description == Name_widgets['Left'][g_ENABLE_CHINESE]:
        g_bot.left(g_step)
    elif b.description == Name_widgets['Right'][g_ENABLE_CHINESE]:
        g_bot.right(g_step)
    elif b.description == Name_widgets['TurnLeft'][g_ENABLE_CHINESE]:
        g_bot.turnleft(g_step)
    elif b.description == Name_widgets['TurnRight'][g_ENABLE_CHINESE]:
        g_bot.turnright(g_step)
```

The variable g_step is controlled by the Step slider. When the value of the slider changes, it will detect which button has been pressed, and the value of g_step will be updated immediately to the robot.

```
def on_slider_step(value):
    global g_step
    g_step = value
    if button_forward.icon == 'check':
        g_bot.forward(g_step)
    elif button_backward.icon == 'check':
        g_bot.back(g_step)
    elif button_move_left.icon == 'check':
        g_bot.left(g_step)
    elif button_move_right.icon == 'check':
        g_bot.right(g_step)
    elif button_turn_left.icon == 'check':
        g_bot.turnleft(g_step)
    elif button_turn_right.icon == 'check':
        g_bot.turnright(g_step)
    print("step:", value)
```

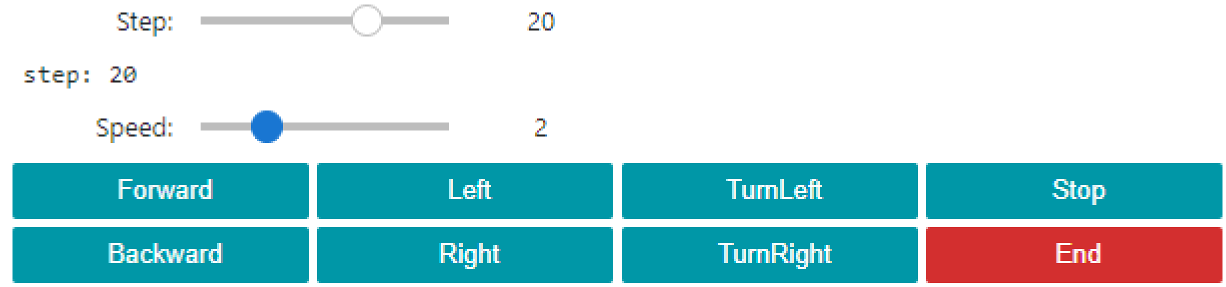
The robot's movement speed is controlled by the Speed slider. When the slider value changes, the robot's movement speed is immediately updated.

```
def on_slider_speed(value):
    global g_bot
    g_bot.speed(value)
    print("speed:", value)
```

4.4 Experiment summary

This time, JupyterLab control is used to control the basic movement of the six-legged robot, including forward and backward, left and right translation, left and right rotation, speed control and other functions. For example, set Step to 20, Speed to 2, and then click the Forward button, the six-legged robot will move forward. At this time, the Forward button will be ticked, and the pressed command will be prompted below the control. At this time, if you change the value of the Step or Speed slider, you can see the robot's step width and speed change.

When you need to stop, please click the Stop button to stop.



Step: 20

```
step: 20
```

Speed: 2

To exit the program, press the End button to exit the program.