2、Speed regulation

2.1 Experimental purpose

This course is mainly to learn how to control the width and frequency of dogzilla's steps, so as to control the movement speed.

2.2 Experimental preparation

The functions of dogzilla Python library involved in this course include:

move(direction, step)：Translate back and forth.

direction：Input the string, input the range ['x ',' x ',' y ',' y '],' x 'or' x 'to move the robot dog forward or backward, and' y 'or' y 'to move the robot dog left or right.

step：Input range X: [- 25,25], Y: [- 18,18], this parameter represents the translation step size. According to the direction, positive value represents forward or left shift, and negative value represents backward or right shift. When the input value exceeds the range, move according to the limit value.

turn(step)：Rotate.

The remainder range of step is [- 100, 100]. This parameter represents the rotation speed, in ° / s. positive values are left turns and negative values are right turns.

pace(mode)：Change the step frequency, speed = step frequency x step width.

mode：Input string, input range ['normal ',' slow ',' high '], normal is the default step frequency, low is the slow step frequency, and high is the high-speed step frequency.

2.3 Experimental process

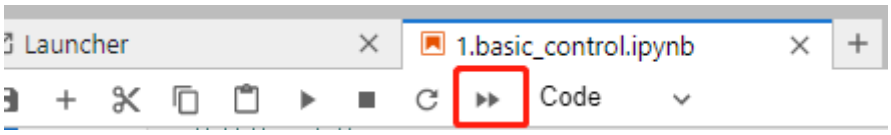
Open the jupyterlab client and find the code path：

```
DOGZILLA/Samples/2_Control/2.speed_adjustment.ipynb
```

By default, G_ ENABLE_ Chinese = false, if you need to display Chinese, please set G_ ENABLE_ CHINESE=True。

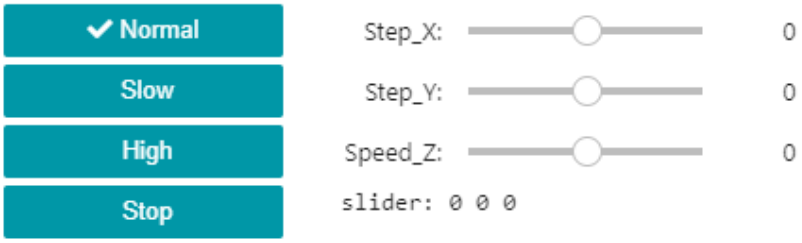
```
# 中文开关，默认为英文 Chinese switch. The default value is English
g_ENABLE_CHINESE = False

Name_widgets = {
    'Stop': ("Stop", "停止"),
    'Normal': ("Normal", "默认步频"),
    'Slow': ("Slow", "慢速步频"),
    'High': ("High", "高速步频"),
    'Step_X': ("Step_X", "前进后退X"),
    'Step_Y': ("Step_Y", "左右平移Y"),
    'Speed_Z': ("Speed_Z", "左右旋转Z")
}
```

Click run all cells, and then drag to the bottom to see the generated control.



布局控件并显示 Layout widgets and display them

```
# 布局控件并显示  Layout widgets and display them
output = widgets.Output()
box_btn = widgets.VBox([button_normal, button_slow, button_high, button_stop])
box_slider = widgets.interactive(on_slider_slide, x=slider_x, y=slider_y, z=slider_z)
box_h = widgets.HBox([box_btn, box_slider])
box_display = widgets.VBox([box_h, output])
display(box_display)
```



The left button is mainly used to set the pace frequency of the robot dog. The default value is normal, the slow speed is slow, and the fast speed is high. If it is set separately, the effect cannot be seen immediately. The effect can be achieved only when the speed is set with the slider bar. The key stop is used to stop the movement.

```
# 按键按下事件处理   Key press event processing
def on_button_clicked(b):
    if b.description == Name_widgets['Stop'][g_ENABLE_CHINESE]:
        slider_x.value = 0
        slider_y.value = 0
        slider_z.value = 0
        return
    ALL_Uncheck()
    b.icon = 'check'
    with output:
        print("Button clicked:", b.description)
    if b.description == Name_widgets['Normal'][g_ENABLE_CHINESE]:
        g_dog.pace('normal')
    elif b.description == Name_widgets['Slow'][g_ENABLE_CHINESE]:
        g_dog.pace('slow')
    elif b.description == Name_widgets['High'][g_ENABLE_CHINESE]:
        g_dog.pace('high')
    else:
        return
    if g_last_x != 0:
        g_dog.move('x', g_last_x)
    if g_last_y != 0:
        g_dog.move('y', g_last_y)
    if g_last_z != 0:
        g_dog.turn(g_last_z)
```

The sliding bar on the right is mainly responsible for controlling the width of the step and matching the step frequency above, so as to control the overall movement speed of the robot dog. Step_ X controls the forward and backward speed, step_ Y controls the speed of left and right translation, speed_ Z controls the speed of left and right rotation.

```
# 滑块滑动事件处理  Slider event handling
def on_slider_slide(x, y, z):
    global g_last_x, g_last_y, g_last_z
    print("   slider:", x, y, z)
    if g_last_x != x:
        g_last_x = x
        g_dog.move('x', x)
        with output:
            print("move x:", x)
    if g_last_y != y:
        g_last_y = y
        g_dog.move('y', y)
        with output:
            print("move y:", y)
    if g_last_z != z:
        g_last_z = z
        g_dog.turn(z)
        with output:
            print("turn z:", z)
```

2.4 Experiment summary

This time, the jupyterlab control is used to control the movement speed of dogzilla, including the step frequency and the step width functions of forward and backward, left and right translation and left and right rotation.