# 4. Basic use of Jupyter Widgets



Jupyter Widgets is an interactive tool that can be used to create dynamic and interactive user interfaces in Jupyter labs; it can be used to build various widgets such as buttons, sliders, text boxes, etc. so that users can interact in Jupyter labs.

## 1. Use Widgets

Import the Jupyter Widgets library and add the following content to the Jupyter lab code block:

```
import ipywidgets as widgets
```

## 2. Use of widgets

### Button

Used to trigger specific actions or events, such as executing specific code by clicking a button.

```
import ipywidgets as widgets
from IPython.display import display

button = widgets.Button(description="Click me")
display(button)

def on_button_clicked(b):
    print("Button clicked")

button.on_click(on_button_clicked)
```

### Slider

Used to select a value by dragging.

```
slider = widgets.IntSlider(value=5, min=0, max=10, step=1)
display(slider)

def handle_slider_change(change):
    print("Slider value:", change.new)

slider.observe(handle_slider_change, names='value')
```

## Text box

Used to enter text data.

```
text = widgets.Text(value="HelloWorld", description="Input:")
display(text)

def handle_text_change(change):
    print("Text value:", change.new)

text.observe(handle_text_change, names='value')
```

## Checkbox

Used for on/off selection.

```
checkbox = widgets.Checkbox(value=False, description='Check me')
display(checkbox)

def handle_checkbox_change(change):
    print("Checkbox value:", change.new)

checkbox.observe(handle_checkbox_change, names='value')
```

## Drop-down menu

Provides a selection list with an option.

```
options = ['Option 1', 'Option 2', 'Option 3']
dropdown = widgets.Dropdown(options=options, value='Option 1',
description='Choose an option:')
display(dropdown)

def handle_dropdown_change(change):
    print("Dropdown value:", change.new)

dropdown.observe(handle_dropdown_change, names='value')
```

## Progress bar

Used to display task progress (this example requires importing ipywidgets and time modules).

```
import ipywidgets as widgets
import time
```

```
progress = widgets.IntProgress(value=50, min=0, max=100,
description='Progress:')
display(progress)

def update_progress():
    for i in range(100):
        progress.value = i
        time.sleep(0.1)

update_progress()
```

## Effect demonstration

Refer to the video in this folder.