Interactive web-app from jupyter notebooks using *Ipywidgets* and *Voila*

Notebook with ipywidgets

voila →

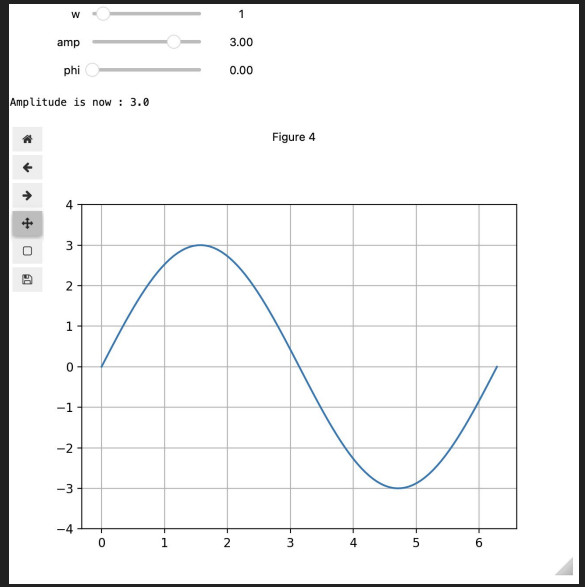
Interactive webpage

```python
# set up plot
fig, ax = plt.subplots()
ax.set_ylim([-4, 4])
ax.grid(True)

# generate x values
x = np.linspace(0, 2 * np.pi, 100) # 100 datapoints, regularly spaced on [0; 2.Pi]


def sinewave(x, w, amp, phi):
    """
    Return a sine for x with angular frequeny w and amplitude amp.
    """
    return amp * np.sin(w * (x-phi))


@widgets.interact(w = (0, 10, 1), amp=(0, 4, .1), phi=(0, 2*np.pi+0.01, 0.01))
def update_plot(w = 1.0, amp=1, phi=0):
    """
    Update the plot/replot, happens each time a slider value is changed.
    remove old lines from plot and plot new one.
    """
    print("Amplitude is now :", amp)
    [line.remove() for line in ax.lines]
    ax.plot(x, sinewave(x, w, amp, phi), color='C0')
```



Examples in this presentation : https://github.com/LauLauThom/ipywidgets-and-voila/
directly test with binder : https://mybinder.org/v2/gh/LauLauThom/ipywidgets-and-voila/HEAD

Fancy examples : https://voila-gallery.org/

Laurent Thomas - 2025

Use cases

- create interactive plots / dashboards
- make simple UI for a python application
- share a notebook with end-users / hide the code
- create training material
- make small self-contained data-apps (e.g data-visualization)
- prototype a user-interface (UI) before going for something more complex

Installation / Requirements

- **matplotlib** to plot stuff (or other equivalent : seaborn…)
- **ipywidgets** provides the UI interactive elements in the notebooks
- **ipympl** provides the matplotlib interactive backend in jupyter notebooks (`@matplotlib widget`)
- **voila** to render the notebook to an html page

- **With conda**

```
conda install matplotlib voila ipympl ipywidgets
```

- **With pip**

One of the following

- `pip install voila`
- `python  -m pip install matplotlib voila ipympl ipywidgets`
- `python3 -m pip install matplotlib voila ipympl ipywidgets`

Getting started

To start voila in the current terminal working directory (show a directory browser like jupyter)

```
voila
```

To directly open a notebook with voila (as an app)

```
voila path/to/notebook.ipynb
```

Demo - Using *@widgets.interact* decorator

Simplest approach (minimal changes to an existing notebook)

```
%matplotlib widget
import ipywidgets as widgets
```

- Widget type (dropdown, slider) inferred from data-type
- Value-range and step-size defined in the decorator (for numerical types)
- Initial widget value taken from default value of function signature OR passed in the decorator

```
@widgets.interact(w = (0, 10, 1), amp=(0, 4, .1), phi=(0, 2*np.pi+0.01, 0.01))
def update_plot(w = 1.0, amp=1, phi=0):
```

More advanced widgets usage / layout

See the calculator notebook

Bonus : launcher script

How to open the notebook in voila without opening a terminal ?

We can create a bash script, that when double-clicked will open the notebook in a browser.
Below described for Mac but would work similarly for Windows…

In directory of notebook, open a terminal and type

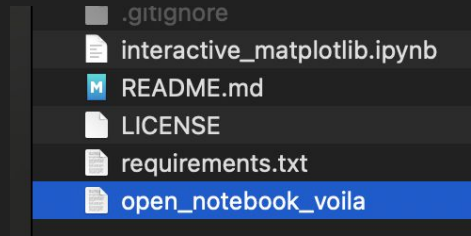
```
touch open_notebook_voila
```

The command above create the file.
The one below makes the file executable.

```
chmod 755 open_notebook_voila
```

Open the created file in a text editor or VScode, and paste

```
#! /bin/sh
cd "$(dirname "$0")" # set the working directory to the one of the script
voila interactive_matplotlib.ipynb # or whatever filename
```



Deploy/distribute the app ?

- Distribute a self-contained python installation (for another session ?)
- mybinder.org (for public repo)

Other solutions see https://voila.readthedocs.io/en/stable/deploy.html

Pros

- Works in both the notebook (within VS code) and the browser
- Minimal modification of the notebook needed
- Reasonable number of dependencies (if already using notebooks)
- Great for relatively simple apps / use cases

Cons

- Not ideal for more complex UI requirements like real-life apps -> code complexity increasing disproportionally

More about ipywidgets / voila

- Ipywidgets with matplotlib – Kapernikov
- Using Interact — Jupyter Widgets 8.1.1 documentation
- Voici : a combination of Voila and Jupyter Lite

Alternatives

- Streamlit
- Plotly and Dash
- NiceGUI