

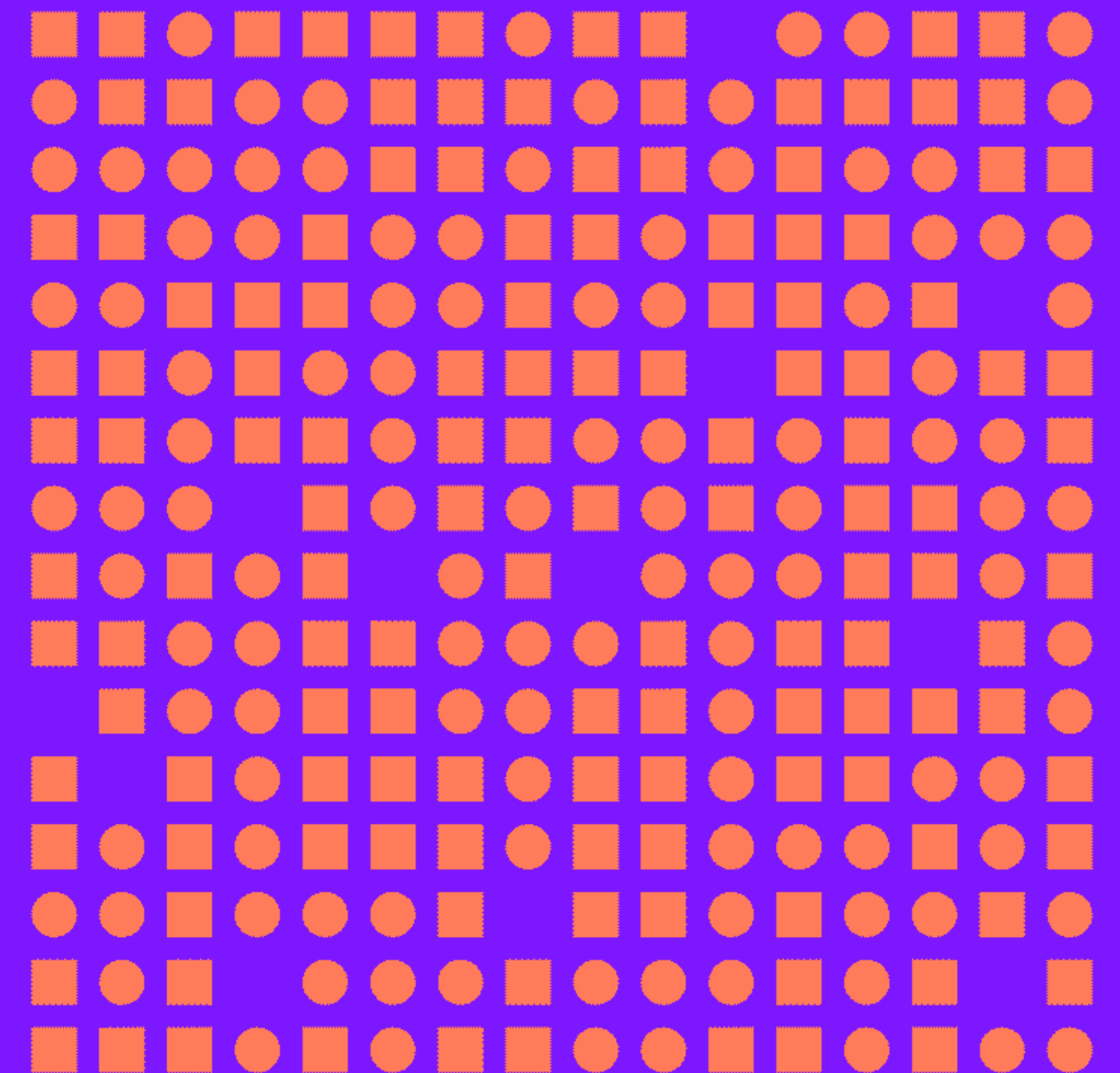
TypeFox

Bringing Notebook Experiences to Modern IDEs

Mark Sujew – LangDev '24

YZVE

 ProxyHands



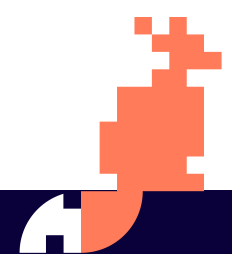
Who am I?
Mark Sujew

TypeFox

HAW
HAMBURG

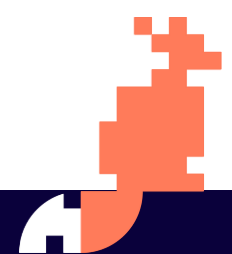
THEIA

Langium



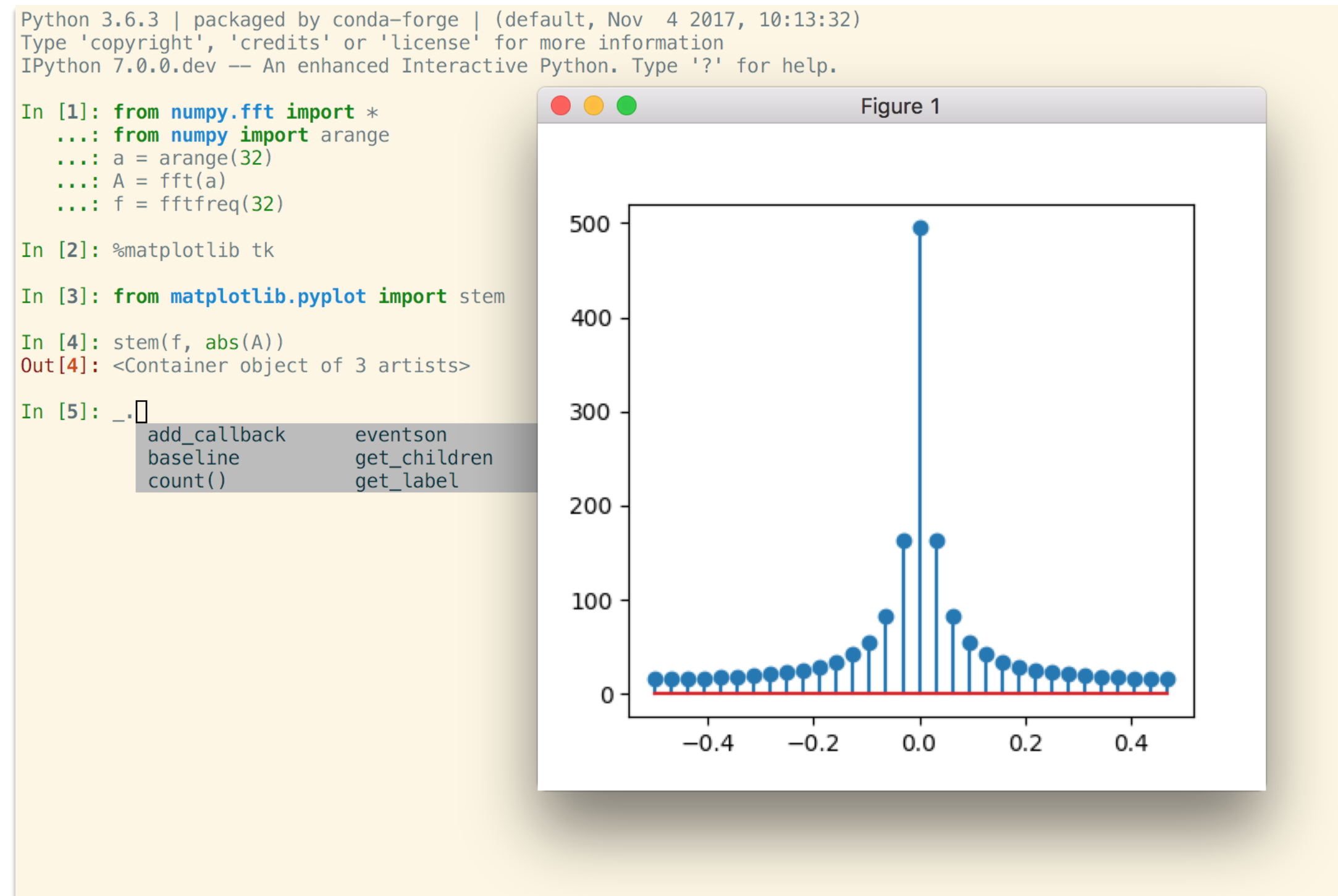
Agenda

1. Explaining Notebooks
2. Initial IDE Integrations
3. Theia IDE Project
4. Native IDE Integration
5. Demo

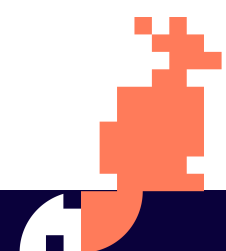


Explaining Notebooks

Interactive Python (IPython)



- Started in 2001
- Execute Python code snippets
- Multimedia Output



Python Notebooks

IP[y]: Notebook Modulation Last Checkpoint: Jan 05 11:01 (autosaved)

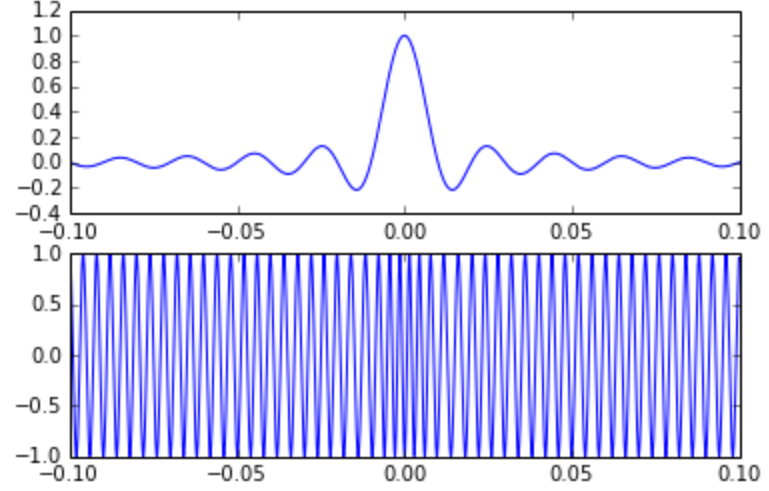
File Edit View Insert Cell Kernel Help

Code Cell Toolbar: None

An angle modulated signal generally can be written as
 $u(t) = A_c \cos(2\pi f_c t + \phi(t))$
In a phase modulated (PM) system, the phase is proportional to the message
 $\phi(t) = k_p m(t)$
In a frequency modulated (FM) system, instantaneous frequency deviation is proportional to the message
 $f_i(t) - f_c = k_f m(t) = \frac{1}{2\pi} \frac{d}{dt} \phi(t)$

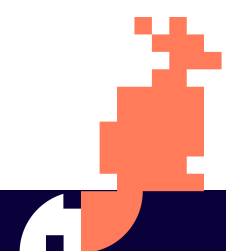
```
In [12]: from numpy.fft import fft,fftfreq
t = arange(-0.1,0.1,0.0001)
m = sinc(100*t)
int_m = empty(len(t))
for k in range(len(t)):
    int_m[k] = trapz(m[0:k],t[0:k])
u = cos(2*pi*250*t + 2*pi*100*int_m)
subplot(211)
plot(t,m)
subplot(212)
plot(t,u)
```

Out[12]: [<matplotlib.lines.Line2D at 0xd3a490c>]

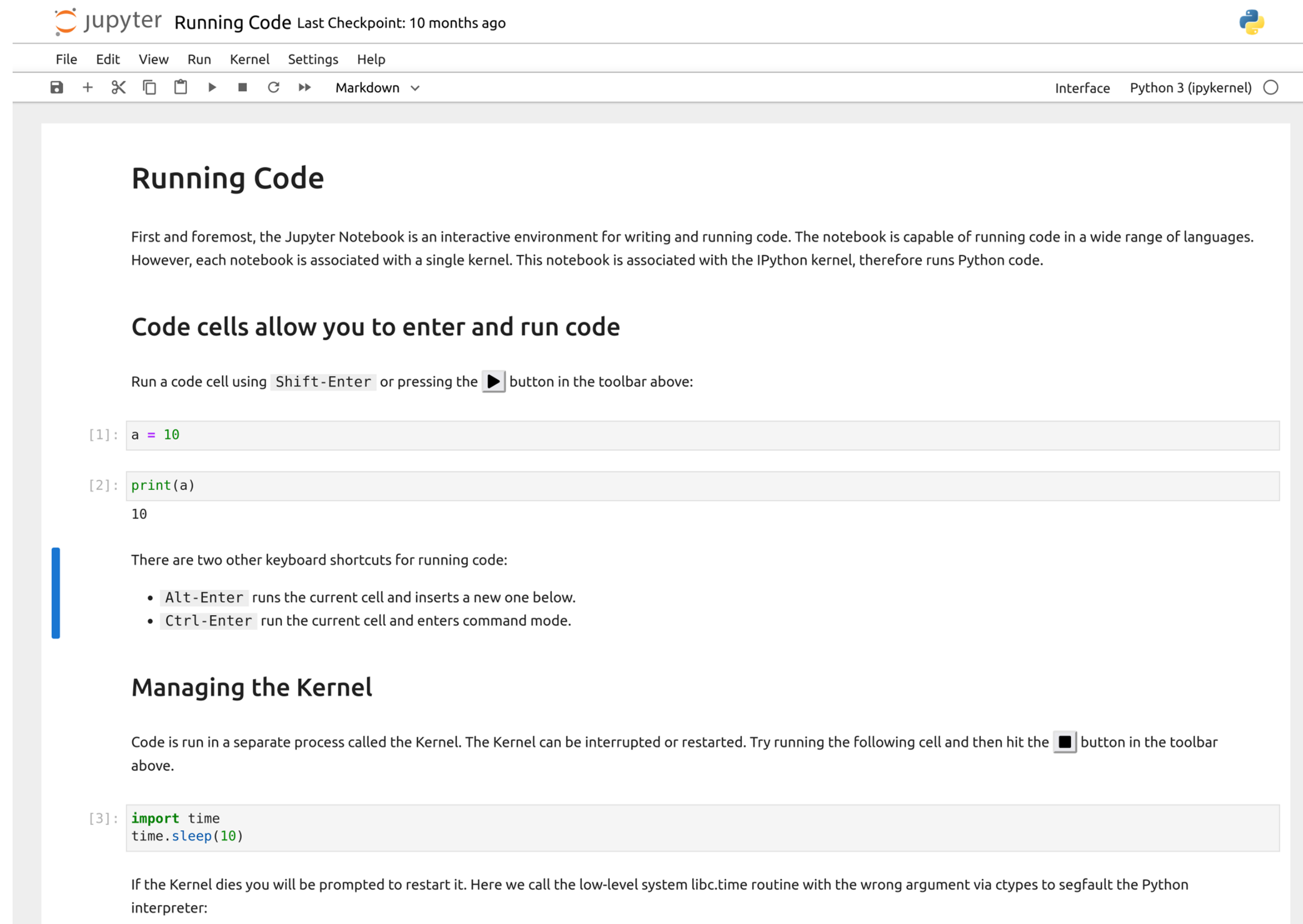


The figure displays two vertically stacked subplots. The top subplot shows a sinc function, which is a smooth curve with a central peak at t=0 and decaying oscillations on either side. The x-axis ranges from -0.10 to 0.10, and the y-axis ranges from -0.4 to 1.2. The bottom subplot shows a high-frequency cosine wave whose frequency varies over time, creating a chirp effect. The x-axis ranges from -0.10 to 0.10, and the y-axis ranges from -1.0 to 1.0.

- Released in 2011
- Main Features from IPython
- Code **and documentation**
- Presented as web app

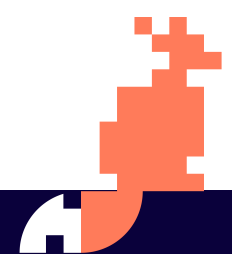


Jupyter Notebook

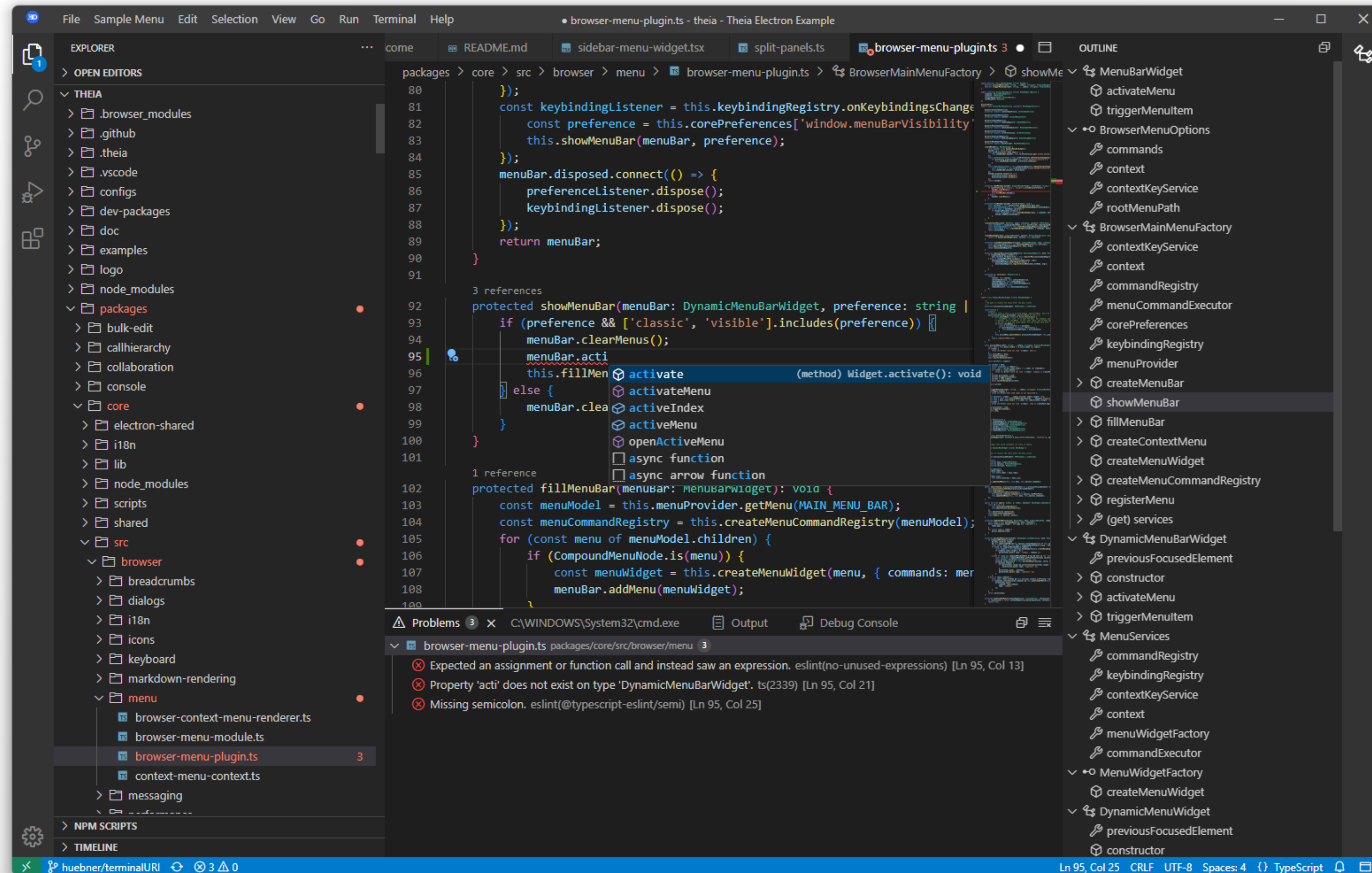


The screenshot shows a Jupyter Notebook interface. At the top, it says 'jupyter Running Code Last Checkpoint: 10 months ago'. Below that is a menu bar with 'File', 'Edit', 'View', 'Run', 'Kernel', 'Settings', and 'Help'. A toolbar contains icons for file operations and a 'Markdown' dropdown. The main content area has a title 'Running Code' and a paragraph: 'First and foremost, the Jupyter Notebook is an interactive environment for writing and running code. The notebook is capable of running code in a wide range of languages. However, each notebook is associated with a single kernel. This notebook is associated with the IPython kernel, therefore runs Python code.' Below this is a section 'Code cells allow you to enter and run code' with instructions: 'Run a code cell using Shift-Enter or pressing the [run] button in the toolbar above:'. There are two code cells: [1]: `a = 10` and [2]: `print(a)` with the output '10'. A list of keyboard shortcuts follows: 'Alt-Enter runs the current cell and inserts a new one below.' and 'Ctrl-Enter run the current cell and enters command mode.' The next section is 'Managing the Kernel' with text: 'Code is run in a separate process called the Kernel. The Kernel can be interrupted or restarted. Try running the following cell and then hit the [stop] button in the toolbar above.' Below is a code cell [3]: `import time` and `time.sleep(10)`. The final paragraph says: 'If the Kernel dies you will be prompted to restart it. Here we call the low-level system libc.time routine with the wrong argument via ctypes to segfault the Python interpreter:'.

- Split off in 2014
- Language agnostic
- Uses IPython as a *Kernel*
- Hugely popular

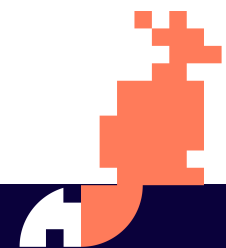
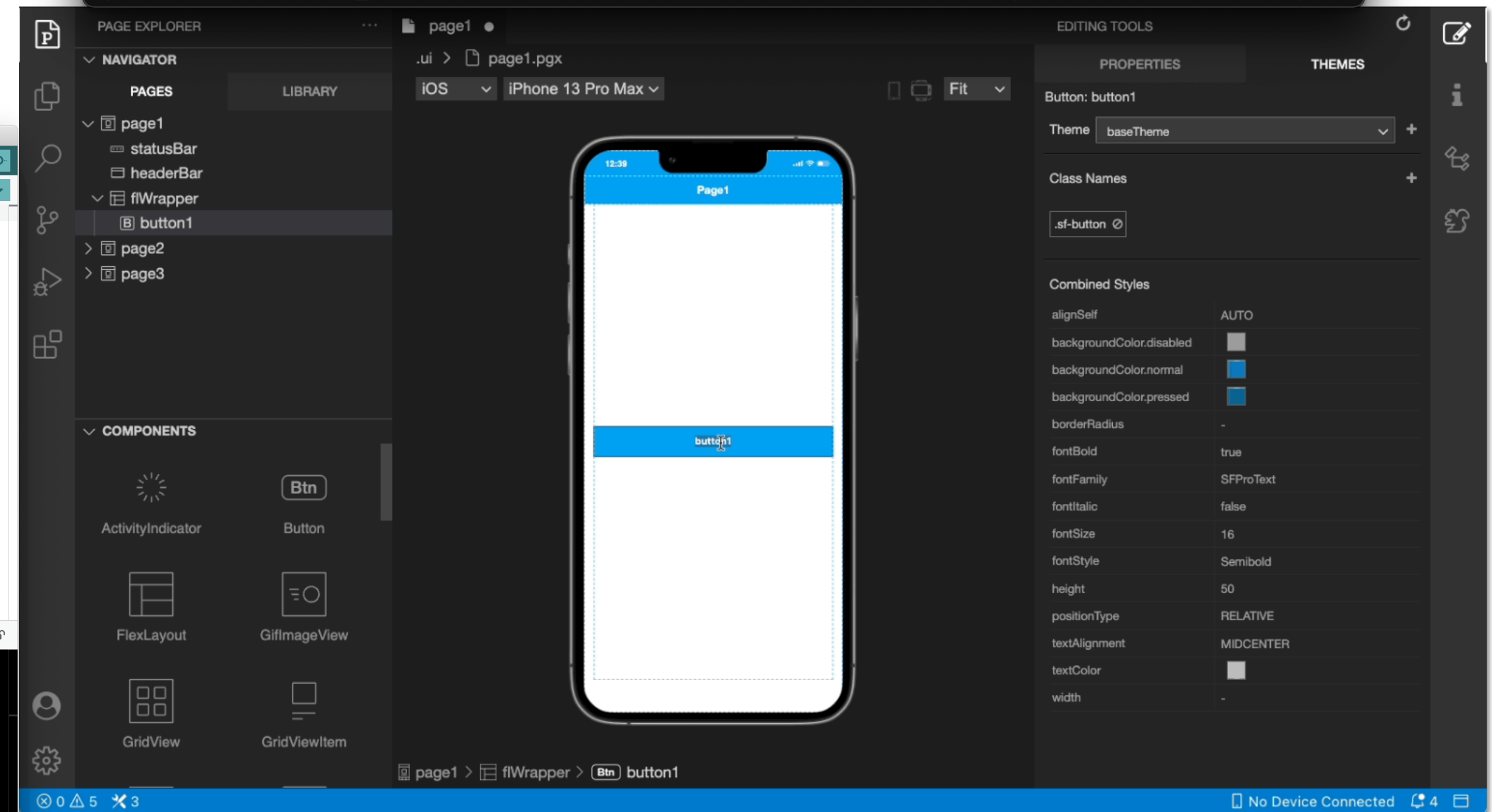
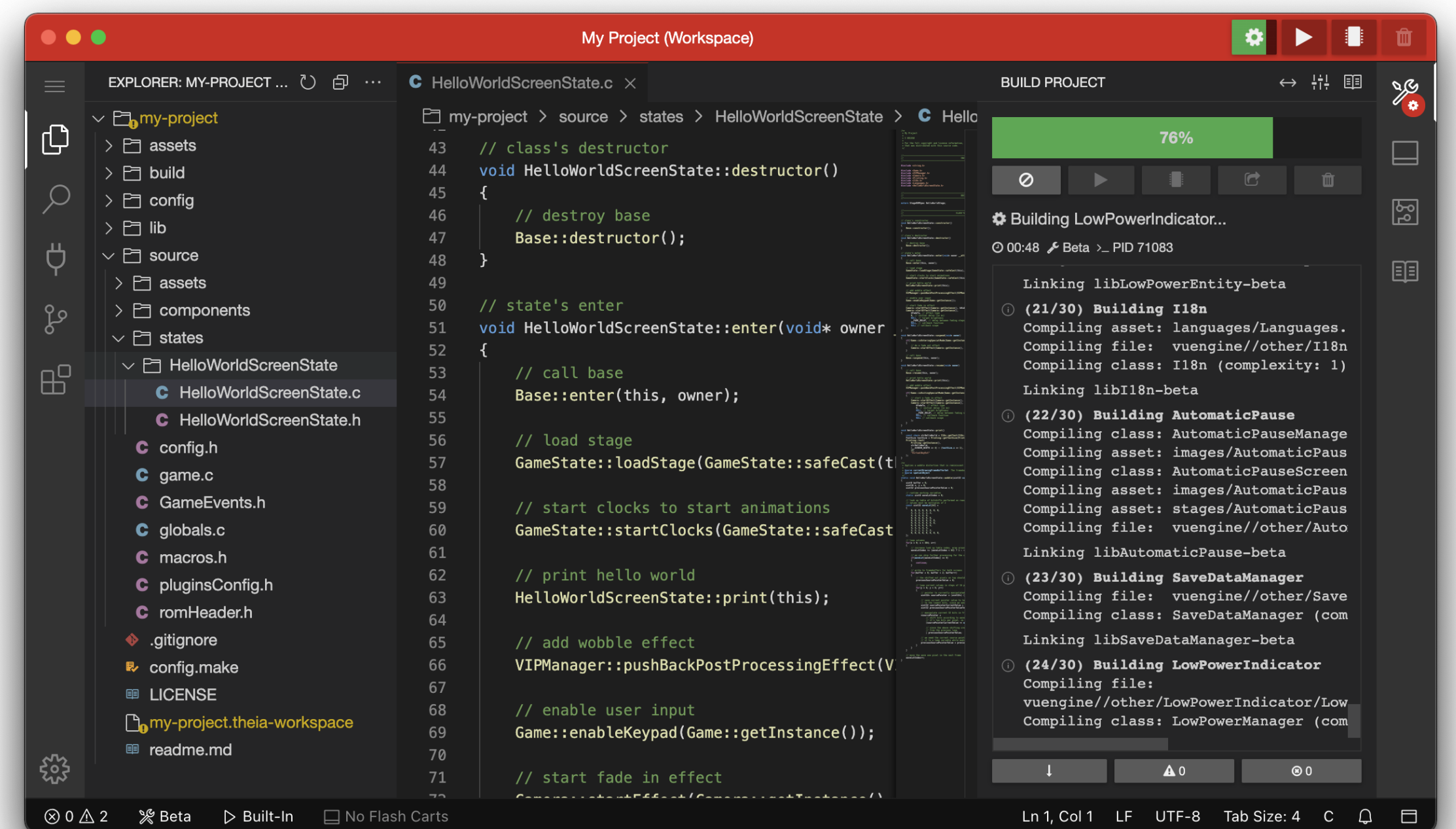
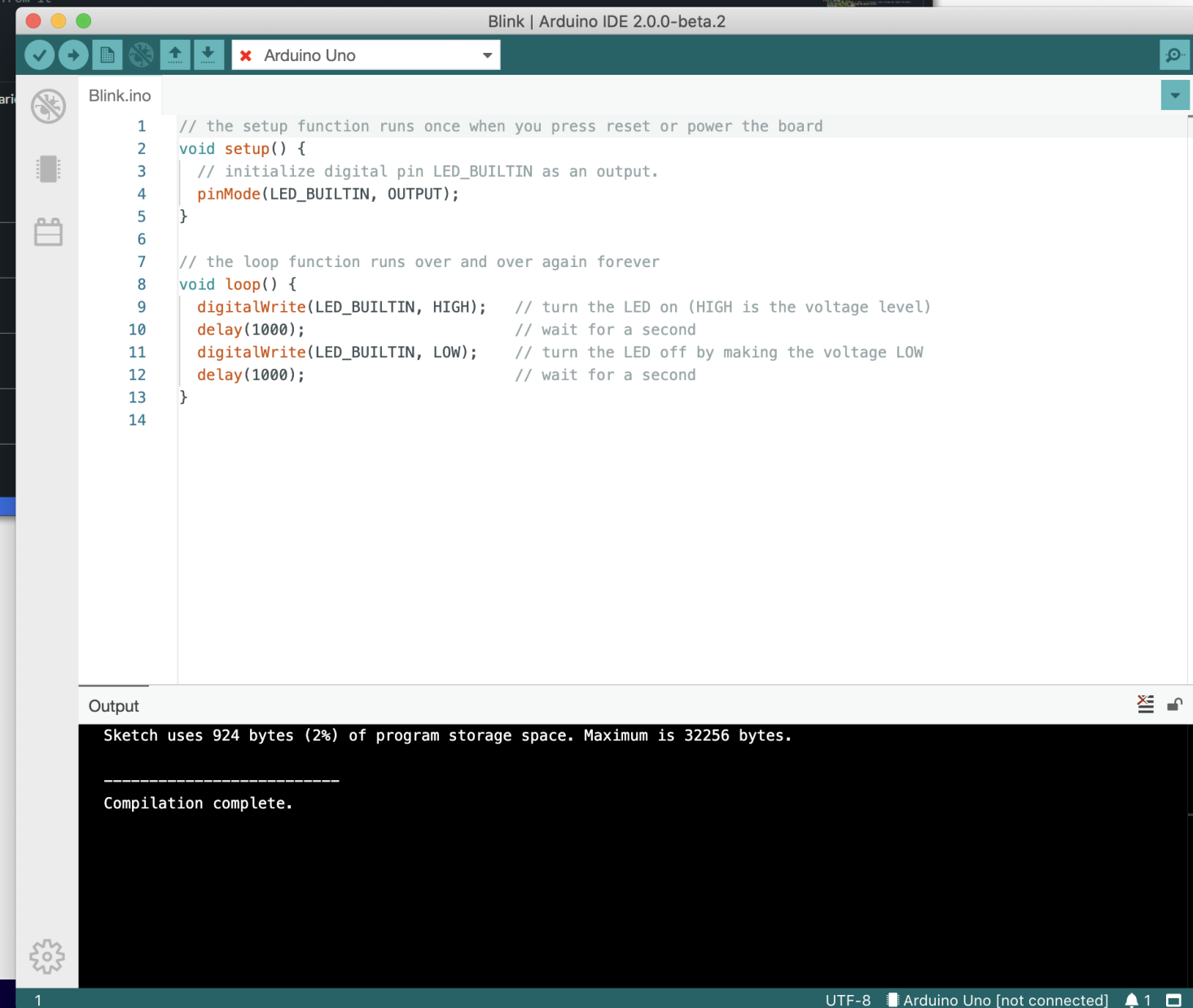
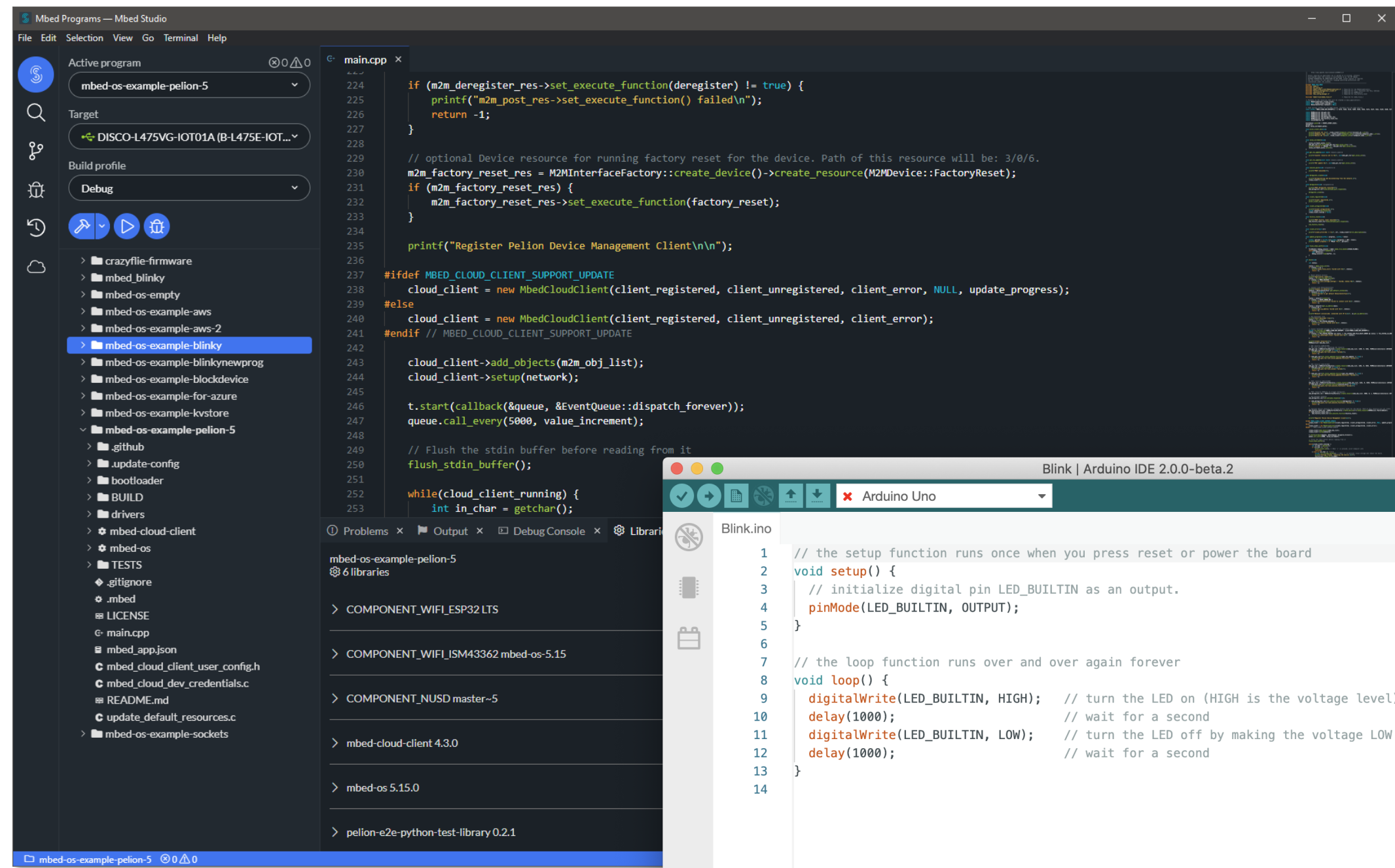


Theia IDE Project

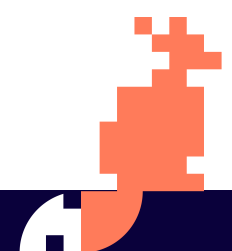
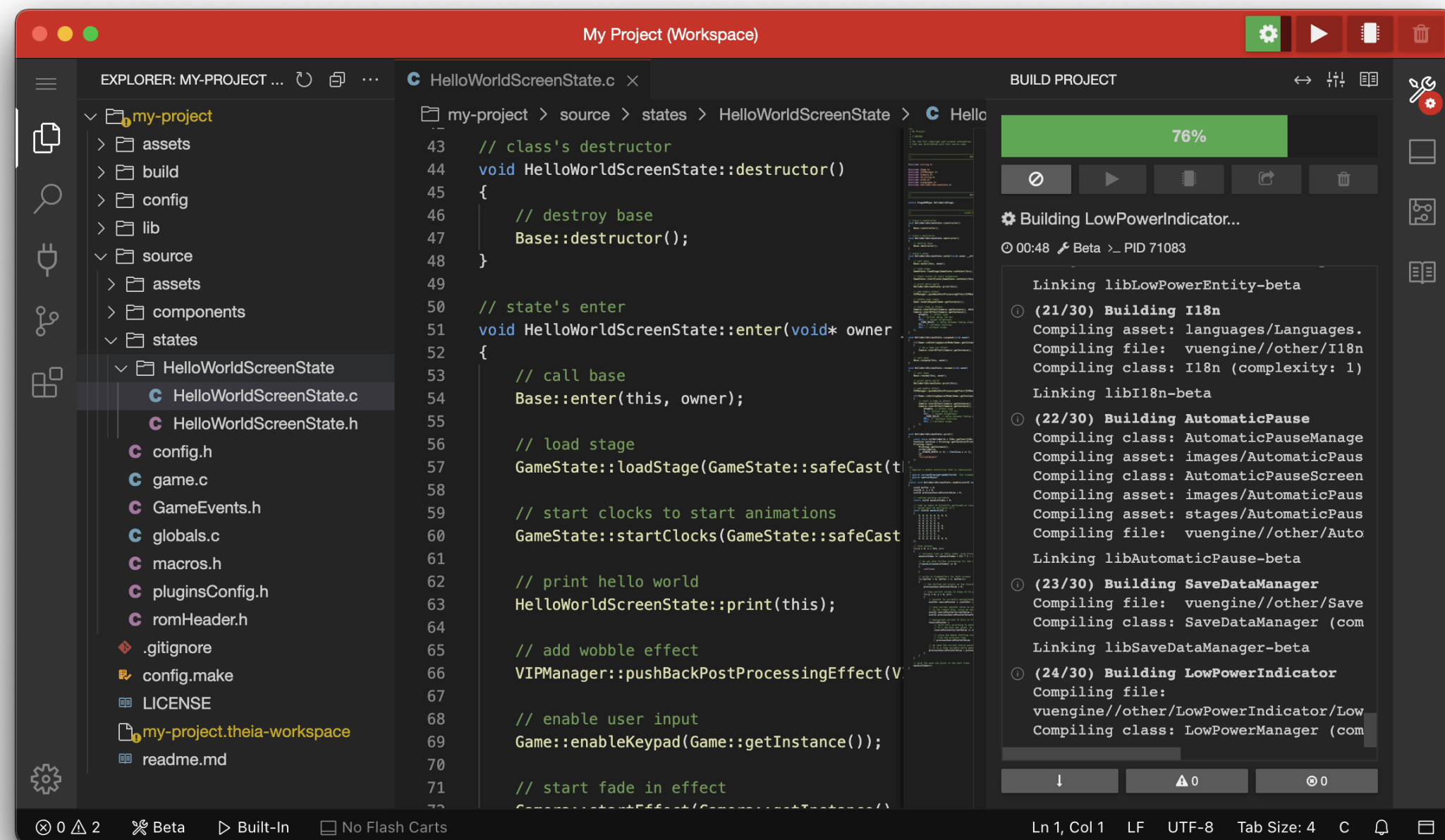


- Created in 2017 by TypeFox
- Designed as a successor to Eclipse Rich Client Platform (RCP)
- Custom extension mechanism
- Support for VS Code plugins

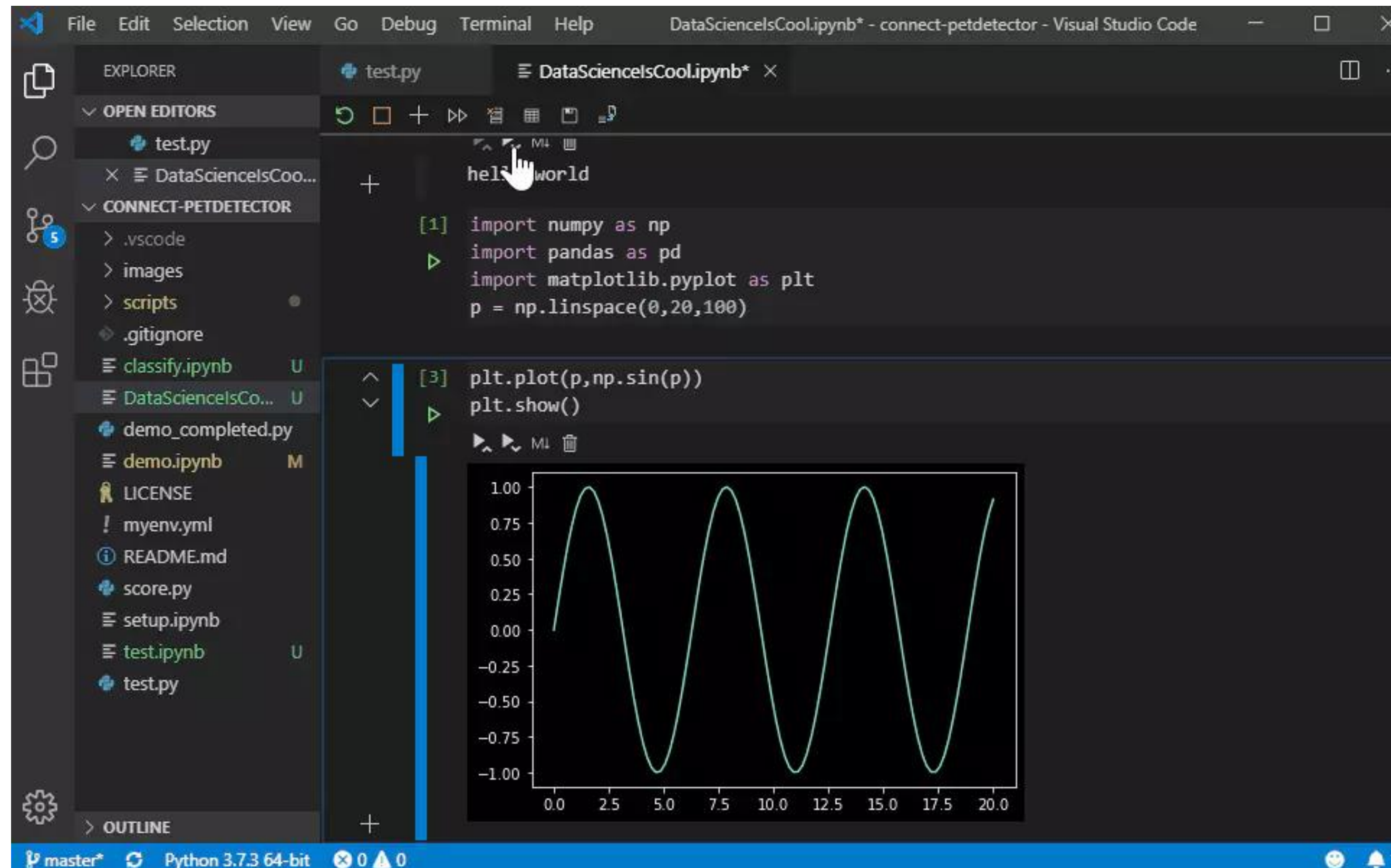
Theia IDE Project Popular Usage



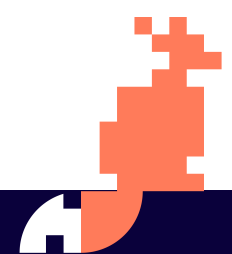
Nintendo Virtual Boy IDE built on Theia



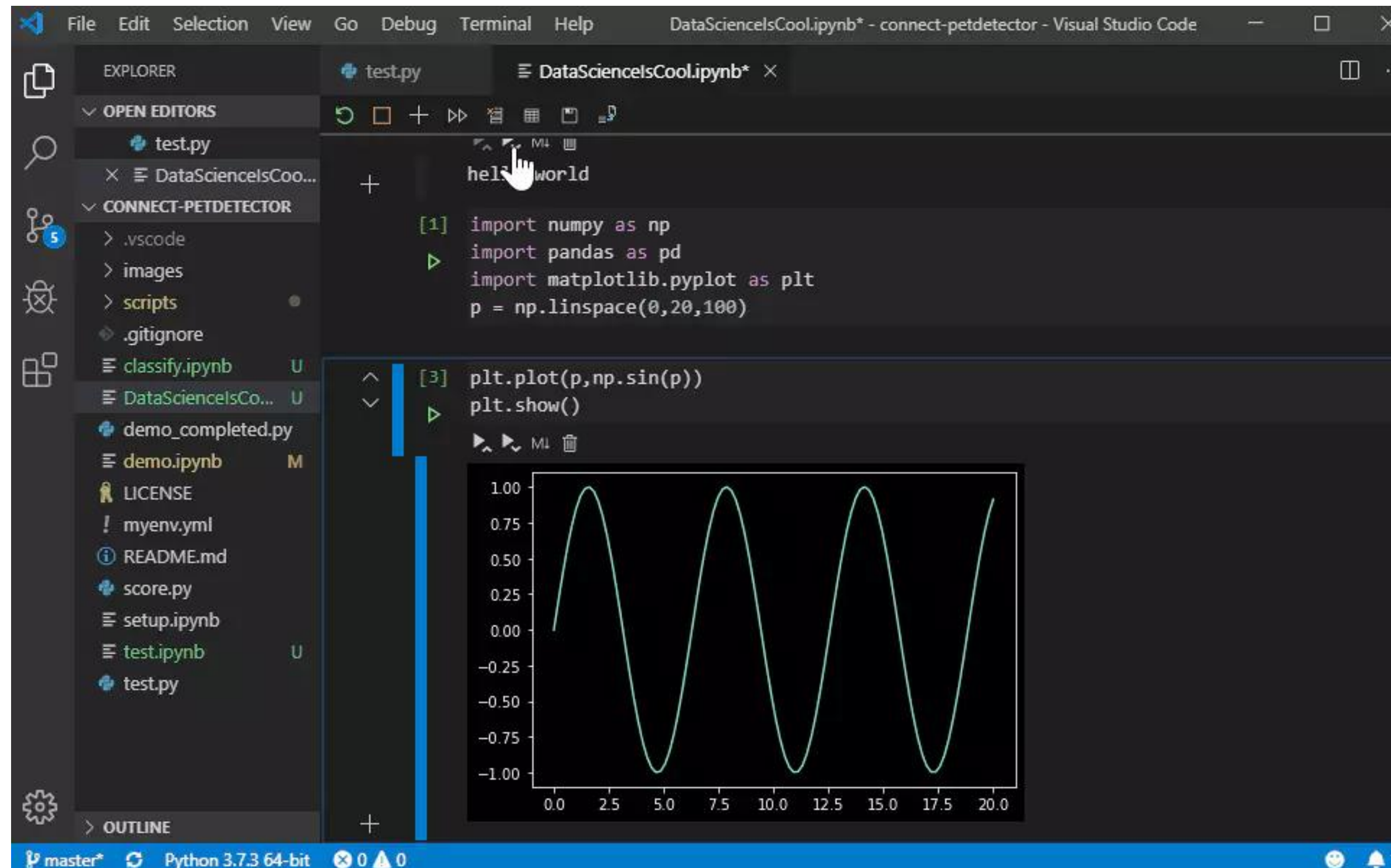
Jupyter Extension for VS Code



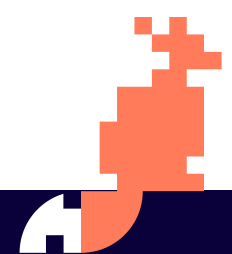
- Created by Microsoft in 2019
- Displays as a *Webview*
- Complicated Python support
- Fairly slow



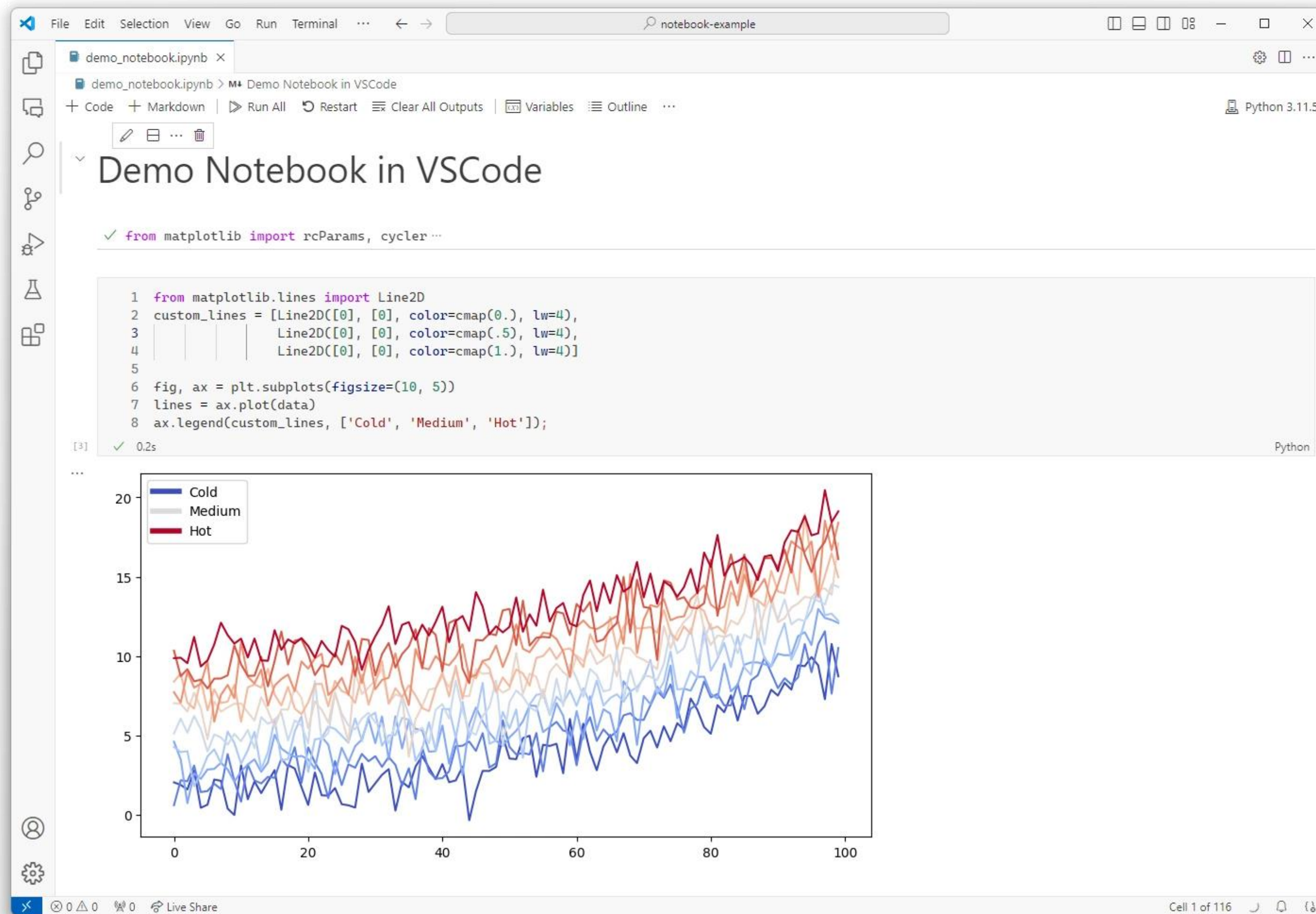
Jupyter Extension for VS Code + Theia



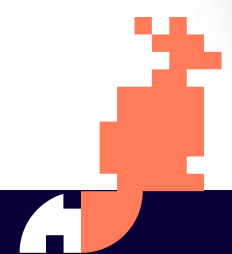
- Created by Microsoft in 2019
- Displays as a *Webview*
- Complicated Python support
- Fairly slow
- **Works in Theia out-of-the-box**



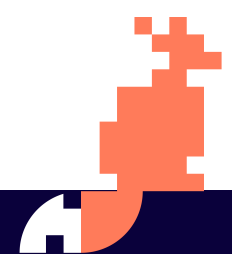
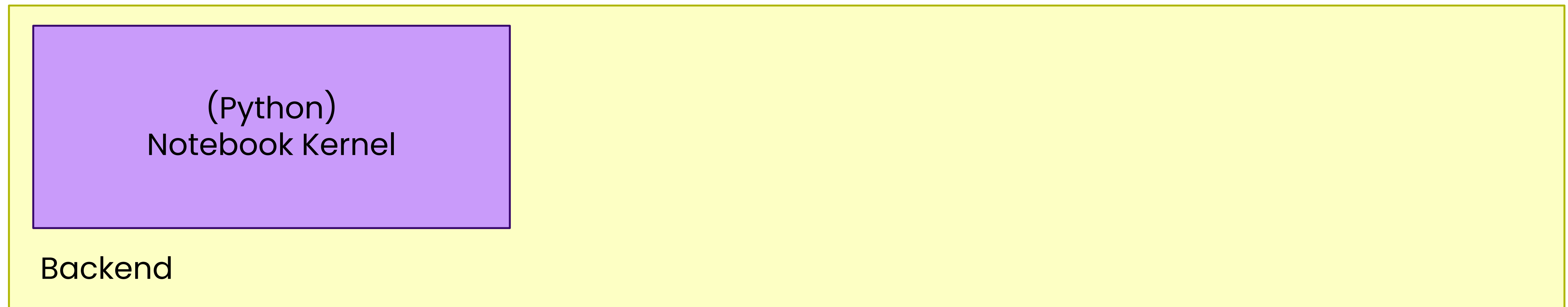
Native Notebooks in VS Code



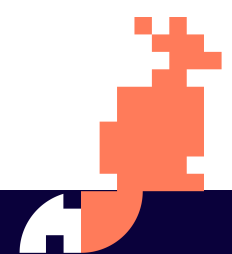
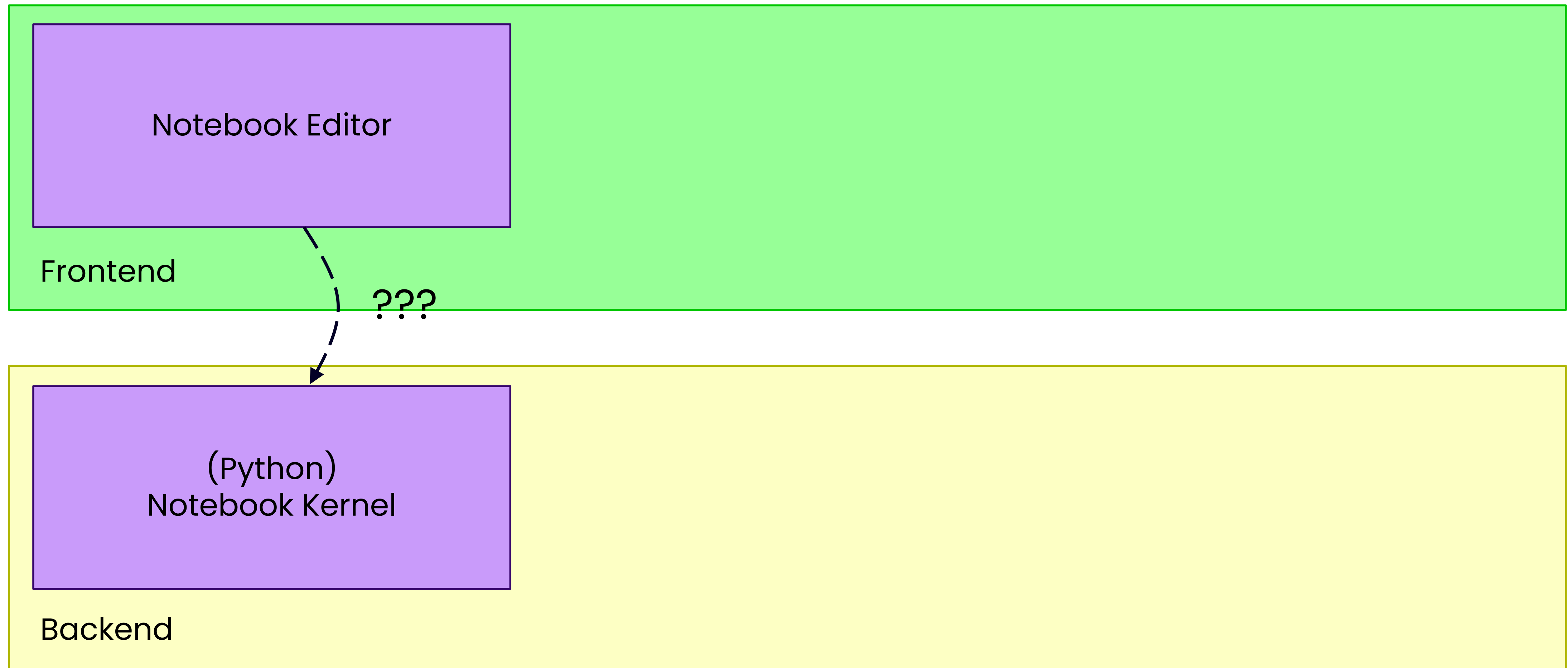
- New UI feature in 2021
- Language agnostic
- Support for custom kernels
- VS Code API



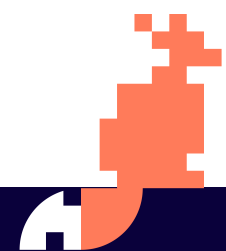
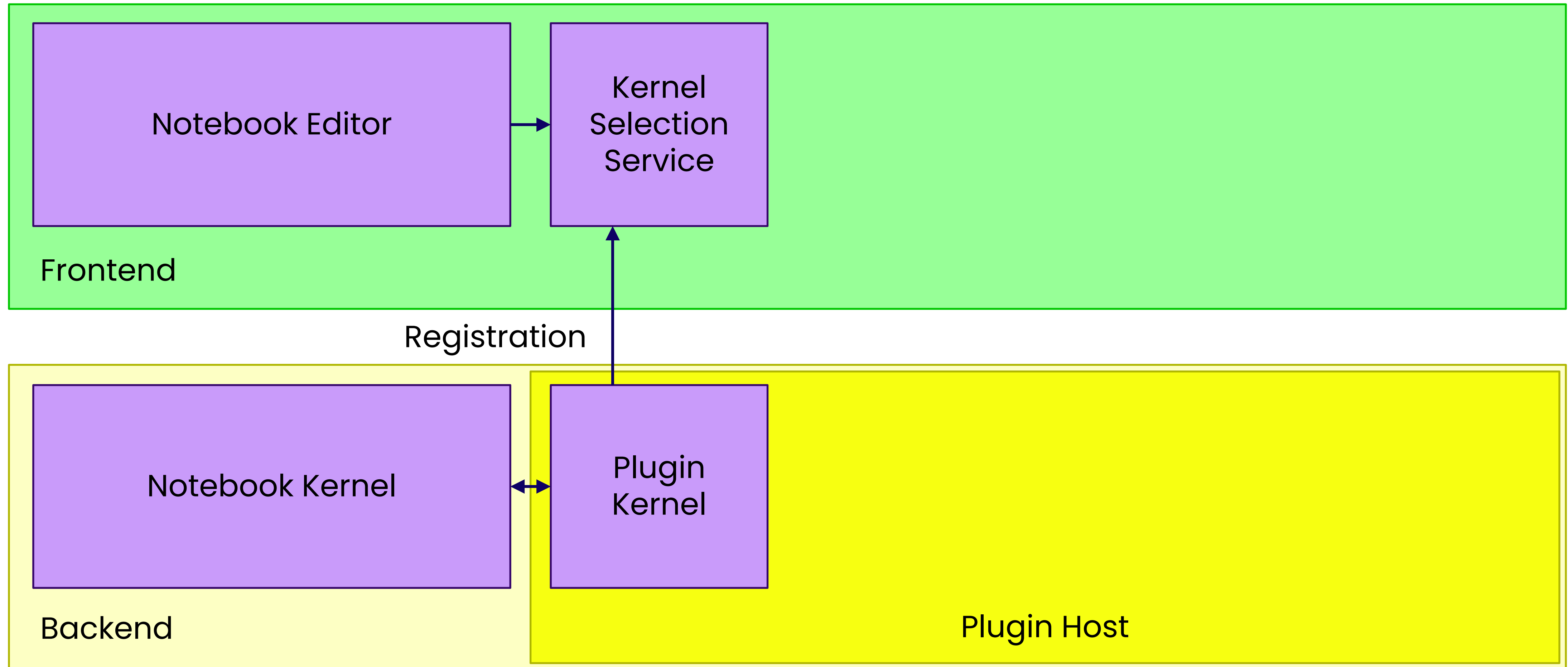
Notebook Communication



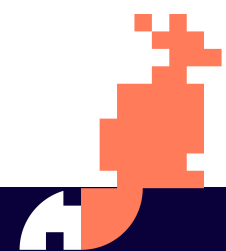
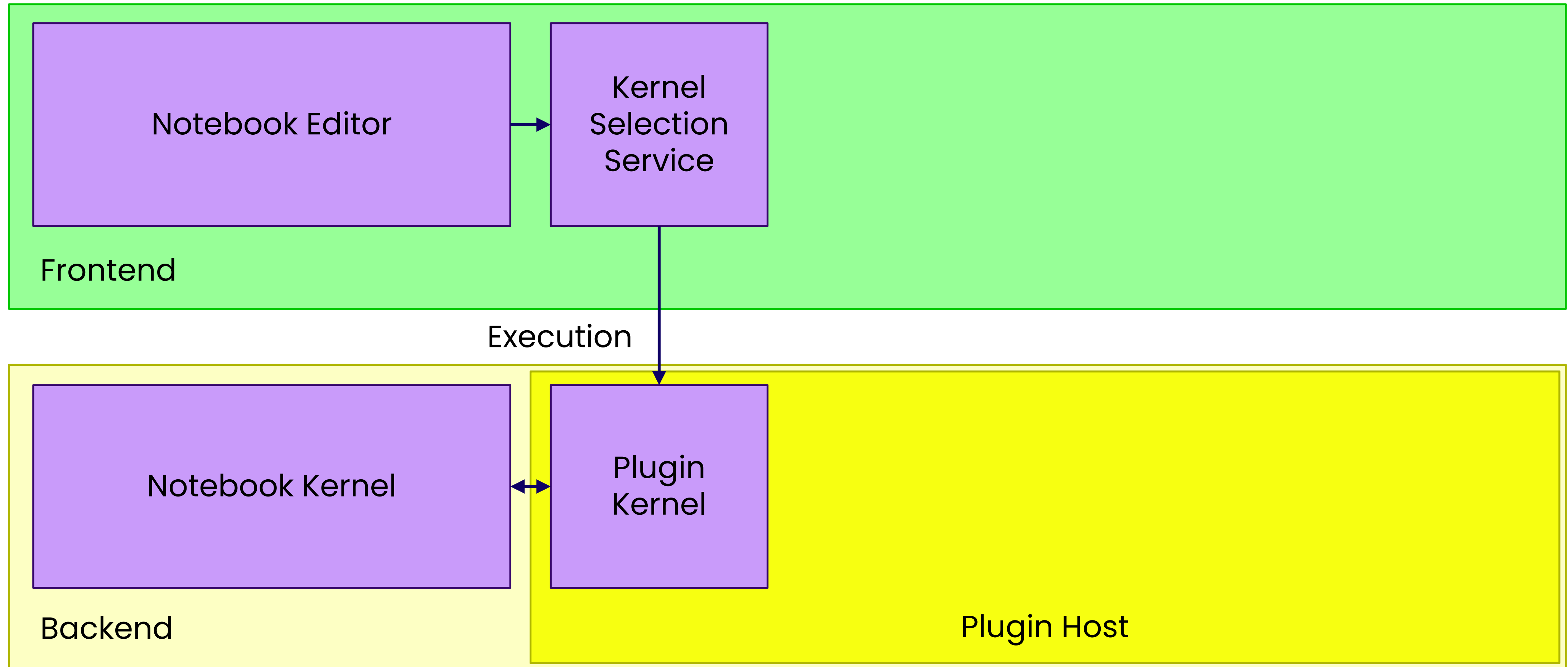
Notebook Communication



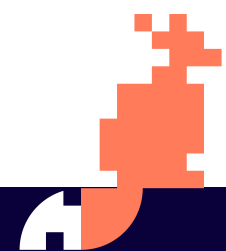
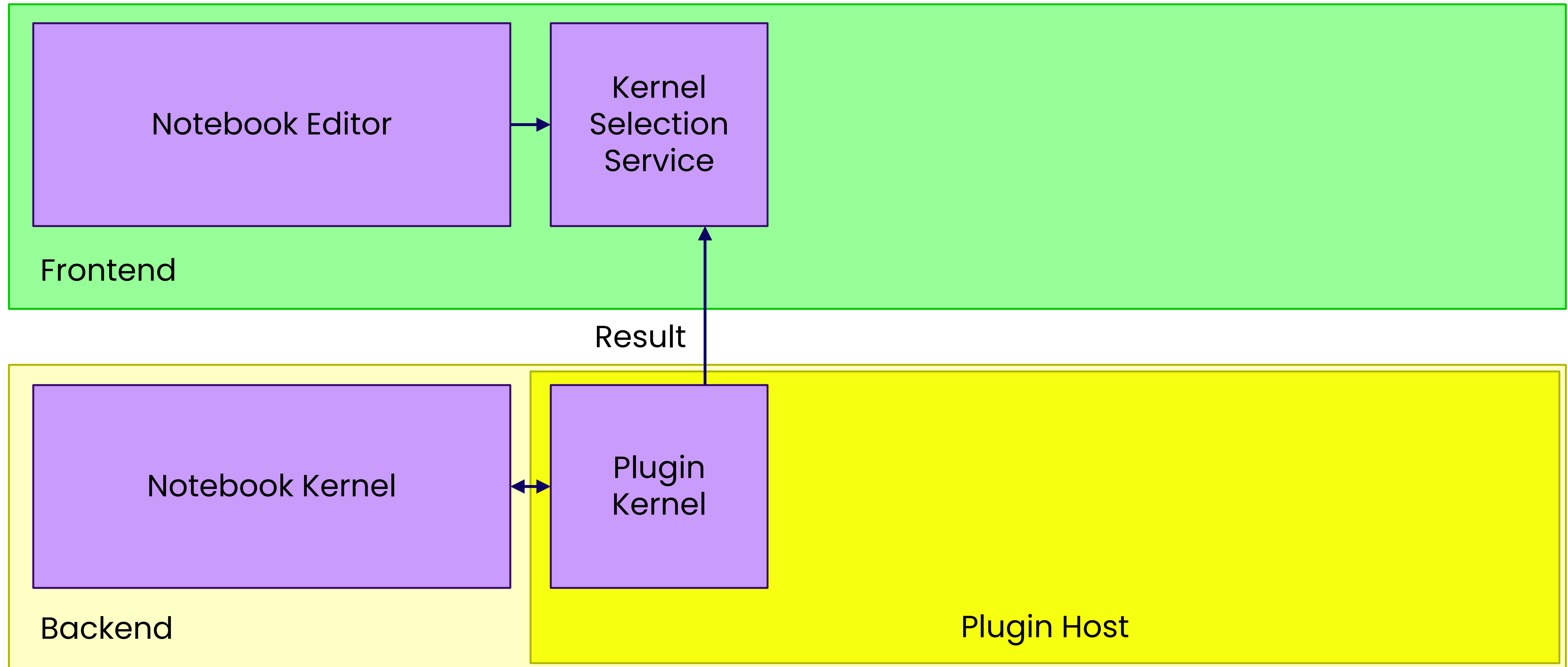
Notebook Communication



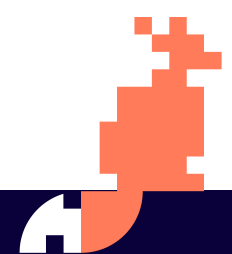
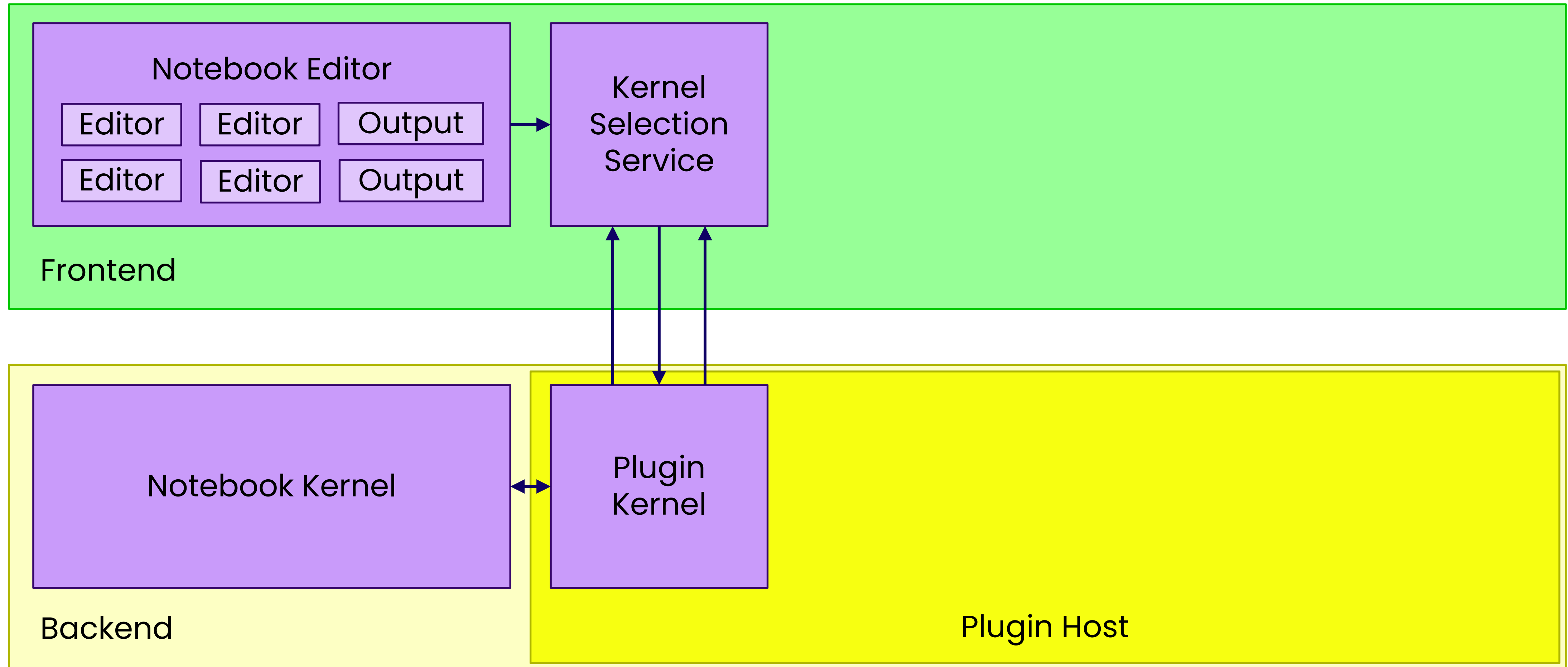
Notebook Communication



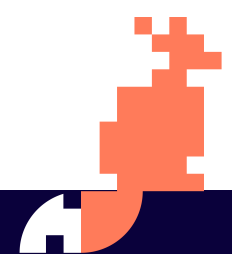
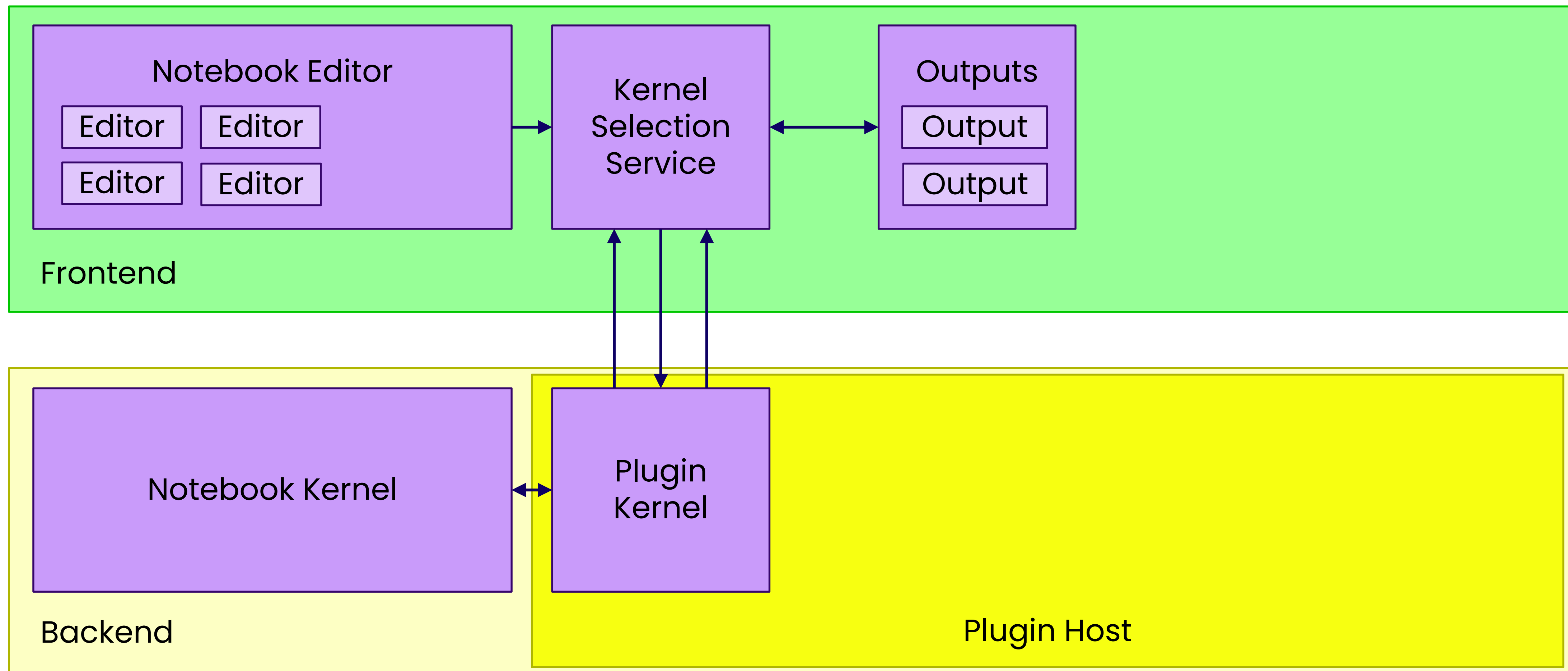
Notebook Communication



Notebook Communication

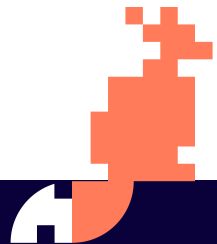
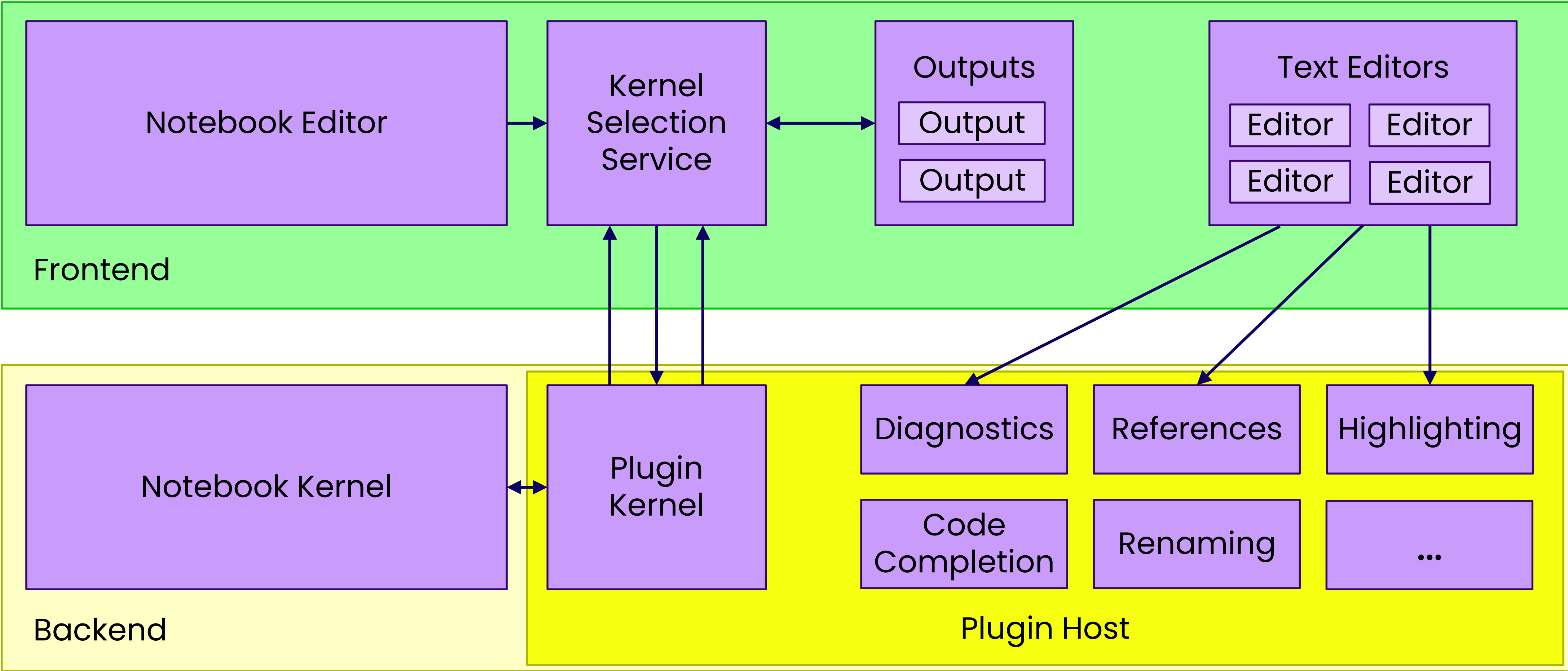


Notebook Communication



Notebook Communication

~20 000 LoC

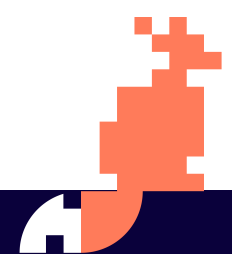
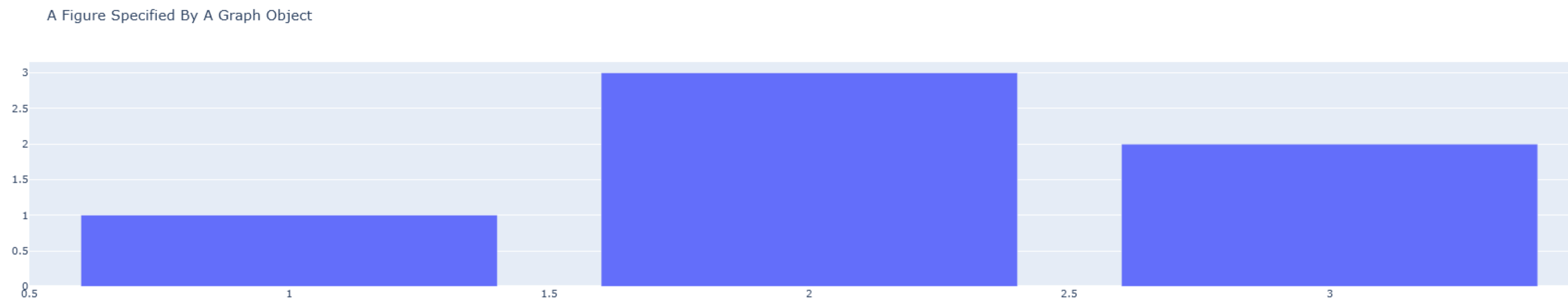


Notebook Layouting

```
1 pop_dict
Python
...
{'Hovedstaden': 1.84,
'Sjælland': 0.84,
'Syddanmark': 1.22,
'Midtjylland': 1.32,
'Nordjylland': 0.59}

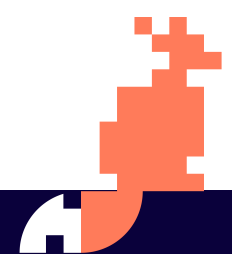
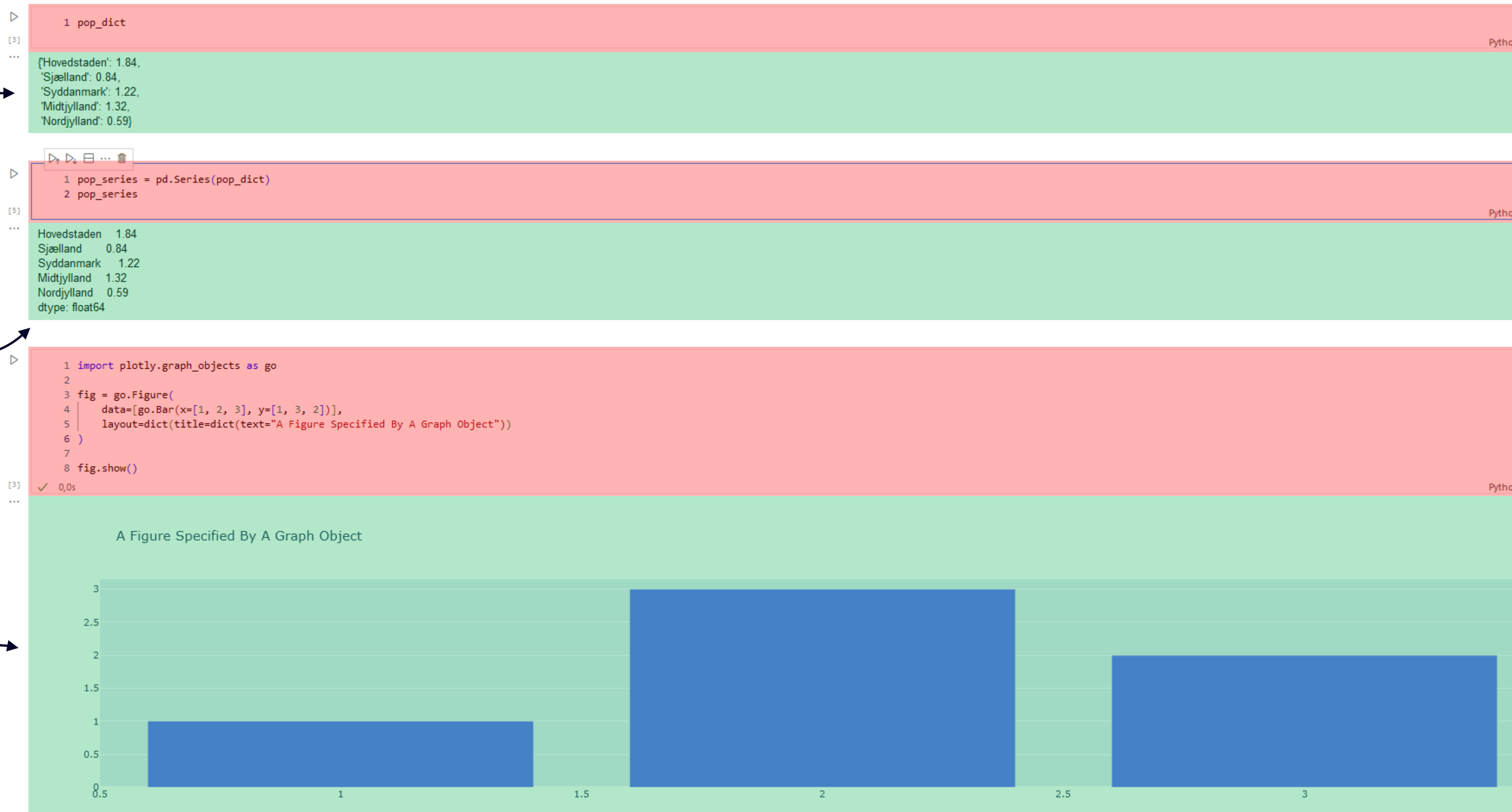
1 pop_series = pd.Series(pop_dict)
2 pop_series
Python
...
Hovedstaden 1.84
Sjælland 0.84
Syddanmark 1.22
Midtjylland 1.32
Nordjylland 0.59
dtype: float64

1 import plotly.graph_objects as go
2
3 fig = go.Figure(
4     data=[go.Bar(x=[1, 2, 3], y=[1, 3, 2])],
5     layout=dict(title=dict(text="A Figure Specified By A Graph Object"))
6 )
7
8 fig.show()
Python
...
✓ 0,0s
```



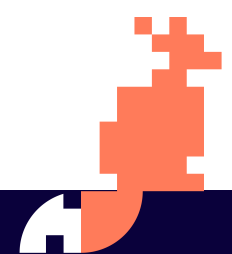
Notebook Layouting – Naive View

Individual
Iframes



Notebook Layouting – Backlayer Webview

Singleton
Iframe



Notebook Layouting – Backlayer Webview

Height Notification

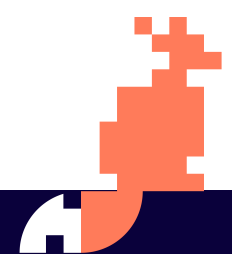
```
1 pop_dict
...
{Hovedstaden: 1.84,
'Sjælland: 0.84,
'Syddanmark: 1.22,
'Midtjylland: 1.32,
'Nordjylland: 0.59}
```

```
1 pop_series = pd.Series(pop_dict)
2 pop_series
...
Hovedstaden 1.84
Sjælland 0.84
Syddanmark 1.22
Midtjylland 1.32
Nordjylland 0.59
dtype: float64
```

```
1 import plotly.graph_objects as go
2
3 fig = go.Figure(
4     data=[go.Bar(x=[1, 2, 3], y=[1, 3, 2])],
5     layout=dict(title=dict(text="A Figure Specified By A Graph Object"))
6 )
7
8 fig.show()
```

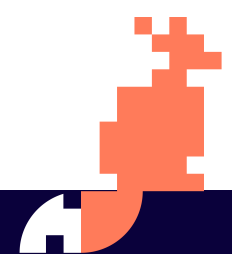
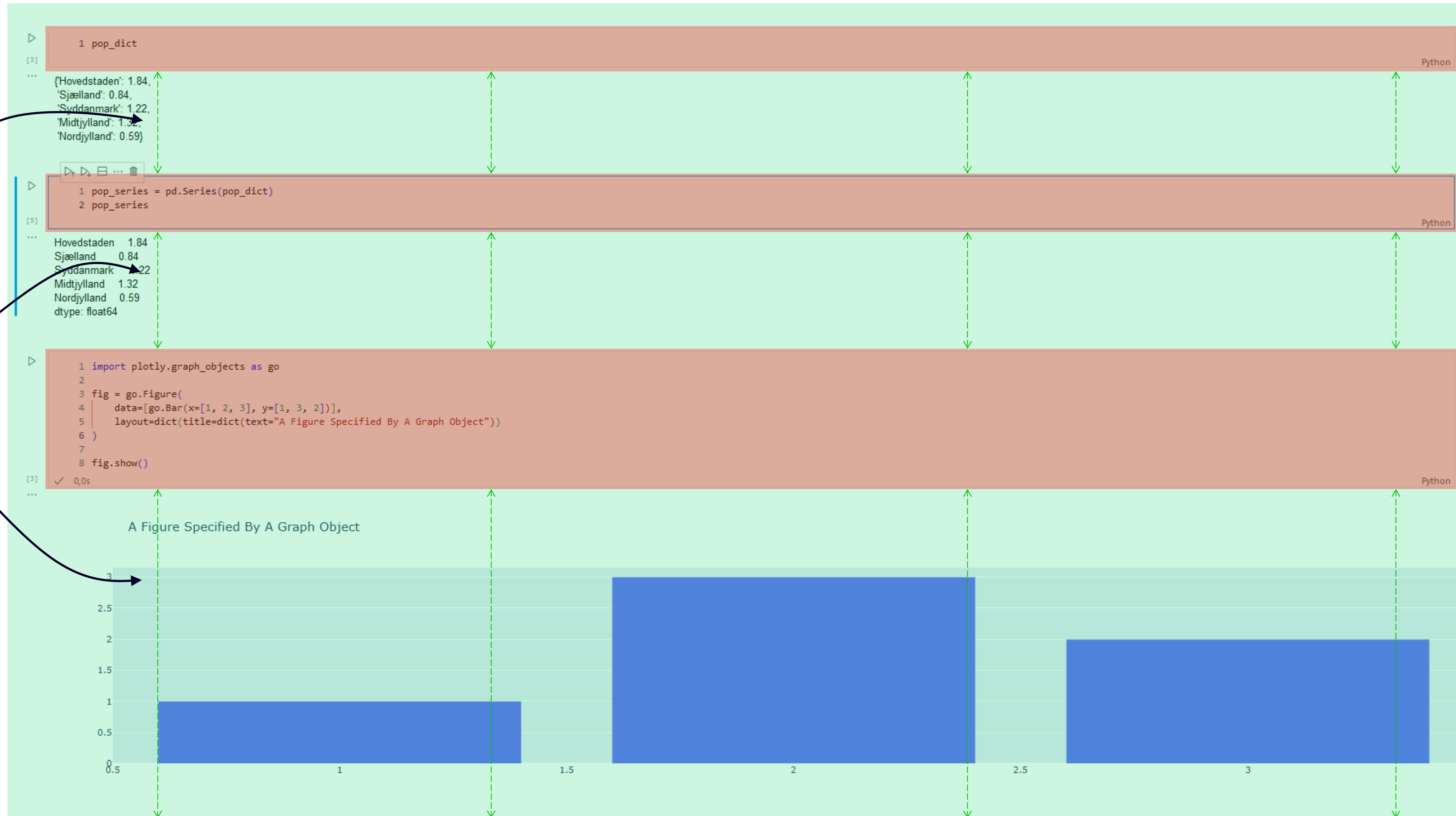
A Figure Specified By A Graph Object

x	y
1	1
2	3
3	2



Notebook Layouting – Editor Overlay

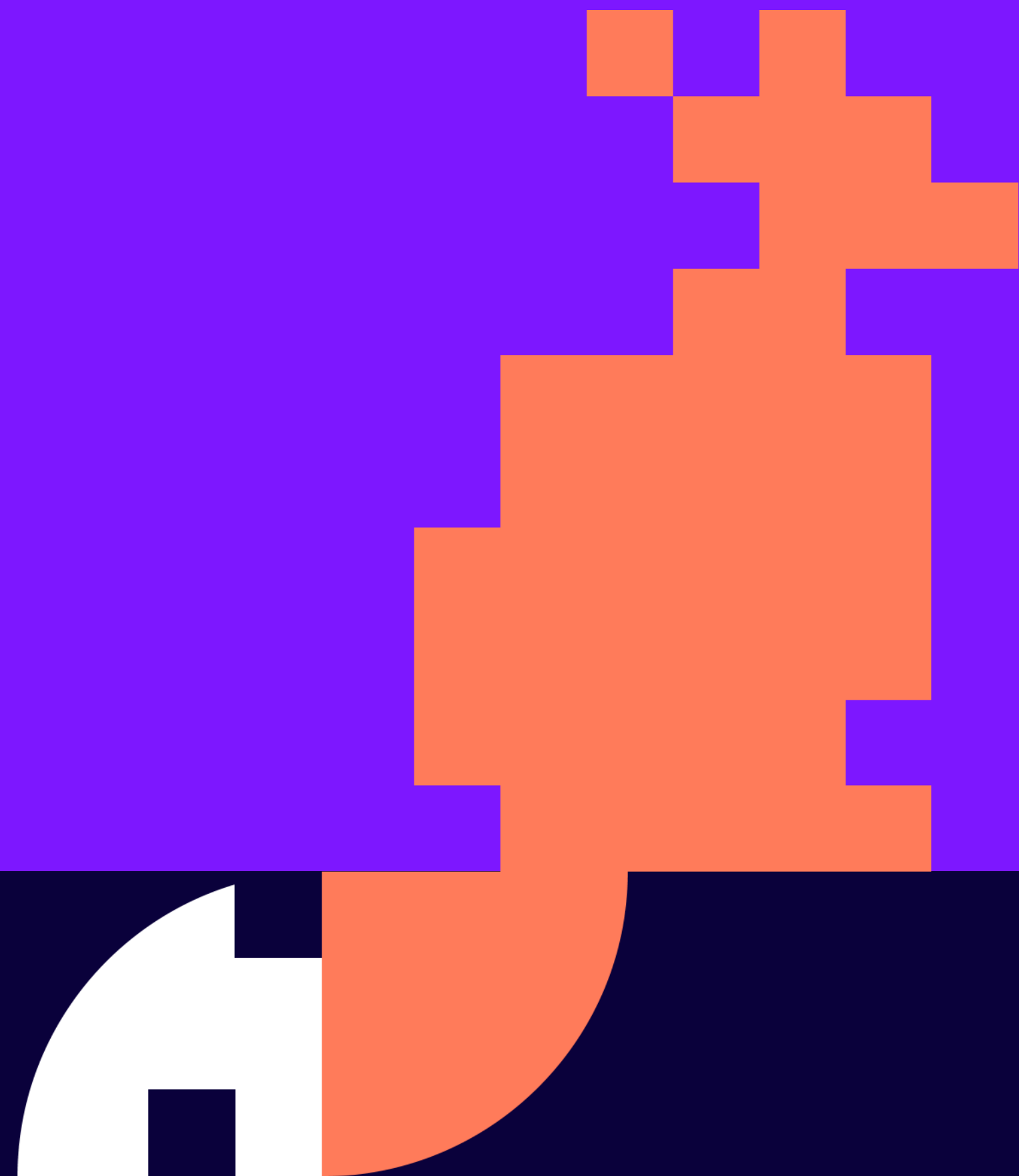
Height Notification





Demo

TypeFox



Questions?

YZVE

