In [1]:

```python
import seaborn as sns
import pandas as pd
import matplotlib.pyplot as plt
from ipywidgets import interact
%matplotlib inline
%config InlineBackend.figure_format = 'svg'
plt.style.use('seaborn-talk')
```

Connect code and reports with

jupyter

Typical guidelines for keeping a notebook of wet-lab work

1. Record everything you do in the lab, even if you are following a published procedure.
2. If you make a mistake, put a line through the mistake and write the new information next to it.
3. Use a ball point pen so that marks will not smear nor will they be erasable.
4. Use a bound notebook so that tear-out would be visible.
5. When you finish a page, put a corner-to corner line through any blank parts that could still be used for data entry.
6. All pages must be pre-numbered.
7. Write a title for each and every new set of entries.
8. It is critical that you enter all procedures and data directly into your notebook in a timely manner.
9. Properly introduce and summarize each experiment.
10. The investigator and supervisor must sign each page.

etc...

Typical guidelines for keeping a notebook of dry-lab work

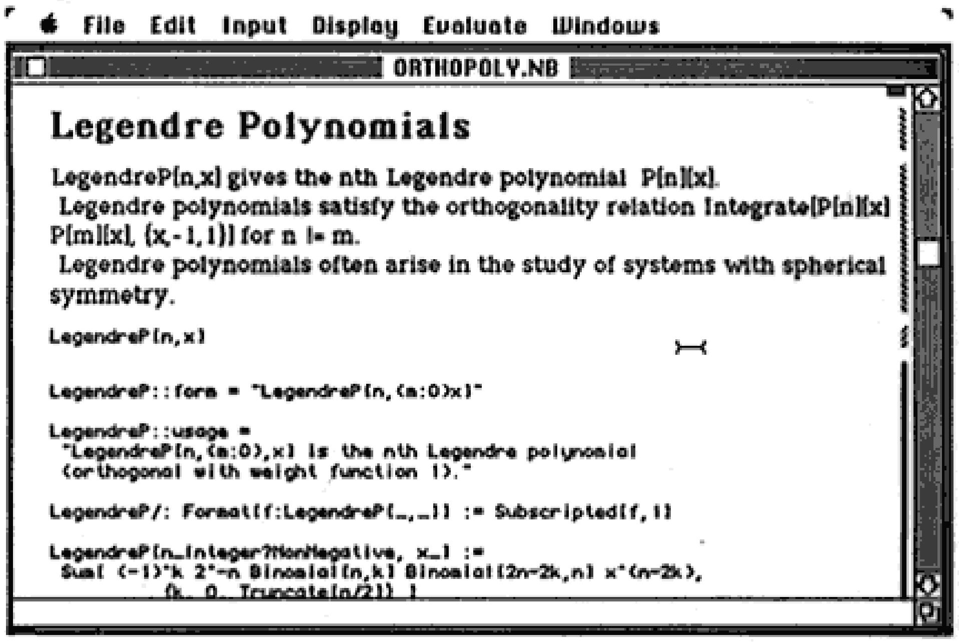
Literate programming

> Instead of imagining that our main task is to instruct a computer what to do, let us concentrate rather on explaining to human beings what we want a computer to do. - *Donald Knuth (1984)*

Literate computing

> A literate computing environment is one that allows users not only to execute commands interactively, but also to store in a literate document the results of these commands along with figures and free-form text. - *Millman KJ and Perez F (2014)*

Wolfram Mathematica notebook (1987)



Legendre Polynomials

LegendreP[n,x] gives the nth Legendre polynomial P[n][x].
Legendre polynomials satisfy the orthogonality relation Integrate[P[n][x] P[m][x], {x,-1,1}] for n != m.
Legendre polynomials often arise in the study of systems with spherical symmetry.

```
LegendreP[n,x]

LegendreP::form = "LegendreP[n,(m:0)x]"

LegendreP::usage =
 "LegendreP[n,(m:0),x] is the nth Legendre polynomial
 (orthogonal with weight function 1)."

LegendreP/: Format[f:LegendreP[_,_]] := Subscripted[f,1]

LegendreP[n_Integer?NonNegative, x_] :=
 Sum[ (-1)^k 2^-n Binomial[n,k] Binomial[2n-2k,n] x^(n-2k),
          {k, 0, Truncate[n/2]} ]
```

The Jupyter notebook

The Jupyter Notebook is a web application for **interactive** data science and scientific computing.

In-browser editing for code, with automatic syntax highlighting, indentation, and tab completion/introspection.

Document your work in Markdown

Penguin data analysis

Here we will investigate the Penguin dataset.

The species included in this set are:

- *Adelie*
- *Chinstrap*
- *Gentoo*

Execute code directly from the browser, with results attached to the code which generated them

In [2]:
```
data = sns.load_dataset("penguins")
data.groupby("species").mean()
```

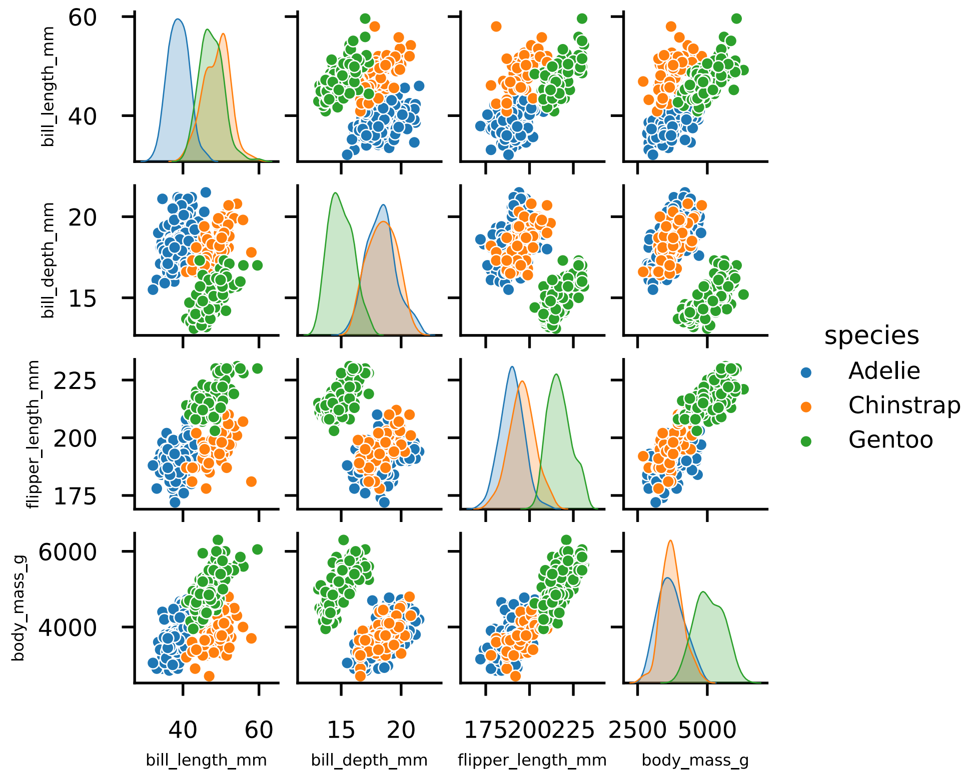
Out[2]:

	bill_length_mm	bill_depth_mm	flipper_length_mm	body_mass_g
species				
Adelie	38.791391	18.346358	189.953642	3700.662252
Chinstrap	48.833824	18.420588	195.823529	3733.088235
Gentoo	47.504878	14.982114	217.186992	5076.016260

Generate plots directly in the browser and/or save to file.

In [3]:
```
sns.set_context("paper", rc={"axes.labelsize":6})
```

In [4]:
```
ax = sns.pairplot(data, hue="species", height=1,
                  plot_kws=dict(s=20, linewidth=0.5),
                  diag_kws=dict(linewidth=0.5))
```



In [5]:
```
%load_ext rpy2.ipython
```

```
/Users/john/python/anaconda3/envs/lectures/lib/python3.8/site-packages/rpy2/robjects/panda
s2ri.py:14: FutureWarning: pandas.core.index is deprecated and will be removed in a future
version.  The public classes are available in the top-level namespace.
  from pandas.core.index import Index as PandasIndex
```

Mix and match languages in addition to `python` (*e.g.* `R`, `bash`, `ruby`)

In [6]:
```
%%R
x <- 1:12
sample(x, replace = TRUE)
```

```
 [1]  2  1  9 12  6  3  7  4  2  6  6  3
```

In [7]:
```
%%bash
uname -v
```

```
Darwin Kernel Version 19.6.0: Tue Oct 12 18:34:05 PDT 2021; root:xnu-6153.141.43~1/RELEASE
_X86_64
```

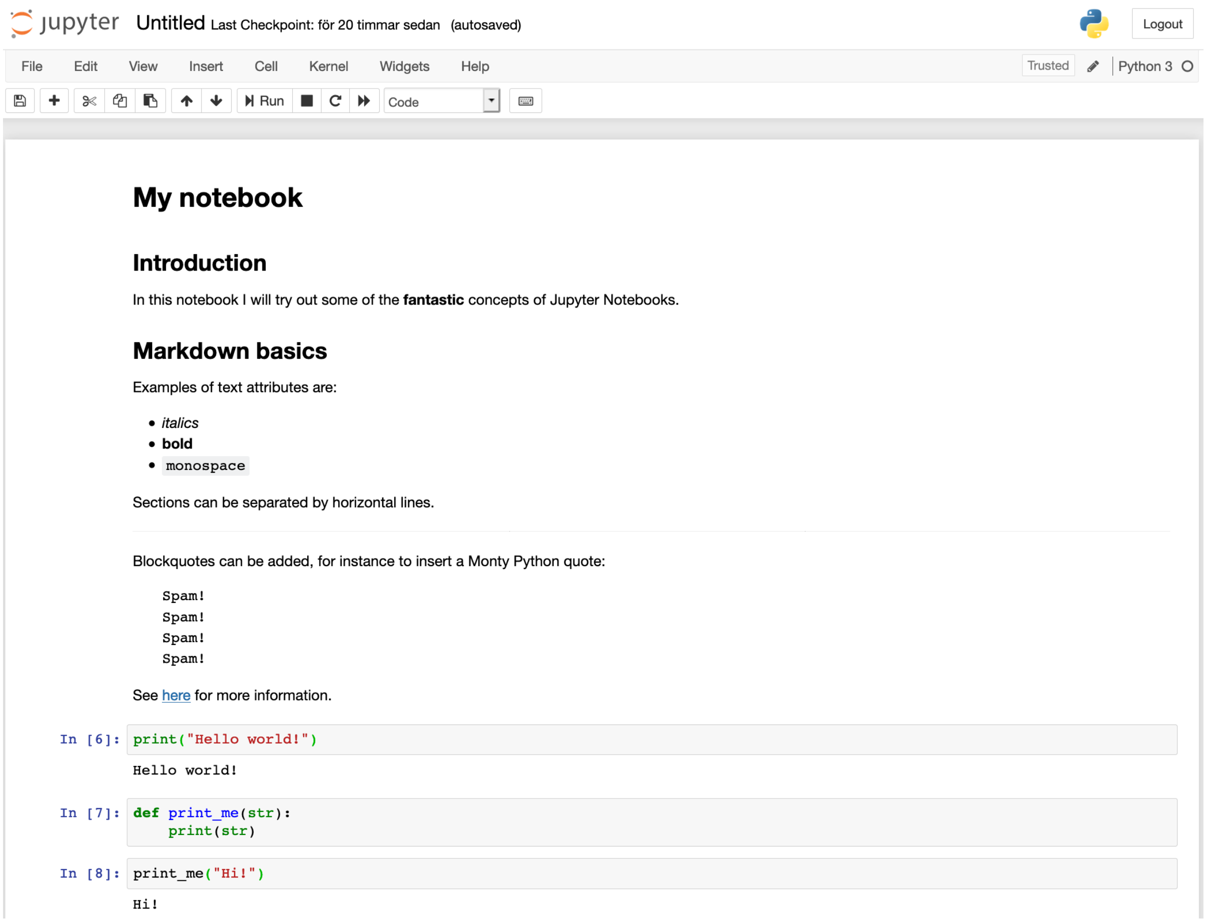
Create interactive widgets

In [8]:
```
def f(palette, x, y):
    plt.figure(1, figsize=(3,3))
    ax = sns.scatterplot(data=data, x=x, y=y, hue="species", palette=palette)
    ax.legend(bbox_to_anchor=(1,1))

_ = interact(f, palette=["Set1","Set2","Dark2","Paired"],
             y=["bill_length_mm", "bill_depth_mm", "flipper_length_mm", "body_mass_g"],
             x=["bill_depth_mm", "bill_length_mm", "flipper_length_mm", "body_mass_g"])
```

Notebook basics

- Runs as a local web server
- Load/save/manage notebooks from the menu



My notebook

Introduction

In this notebook I will try out some of the **fantastic** concepts of Jupyter Notebooks.

Markdown basics

Examples of text attributes are:

- *italics*
- **bold**
- `monospace`

Sections can be separated by horizontal lines.

Blockquotes can be added, for instance to insert a Monty Python quote:

> Spam!
> Spam!
> Spam!
> Spam!

See here for more information.

```
In [6]: print("Hello world!")
```

```
Hello world!
```

```
In [7]: def print_me(str):
            print(str)
```

```
In [8]: print_me("Hi!")
```

```
Hi!
```

The notebook itself is a JSON file

```
In [9]: !head -20 jupyter.ipynb
```

```
{
 "cells": [
  {
   "cell_type": "code",
   "execution_count": 2,
   "metadata": {
    "slideshow": {
     "slide_type": "skip"
    }
   },
   "outputs": [],
   "source": [
    "import seaborn as sns\n",
    "import pandas as pd\n",
    "import matplotlib.pyplot as plt\n",
    "from ipywidgets import interact\n",
    "%matplotlib inline\n",
    "%config InlineBackend.figure_format = 'svg'\n",
    "plt.style.use('seaborn-talk')"
   ]
```

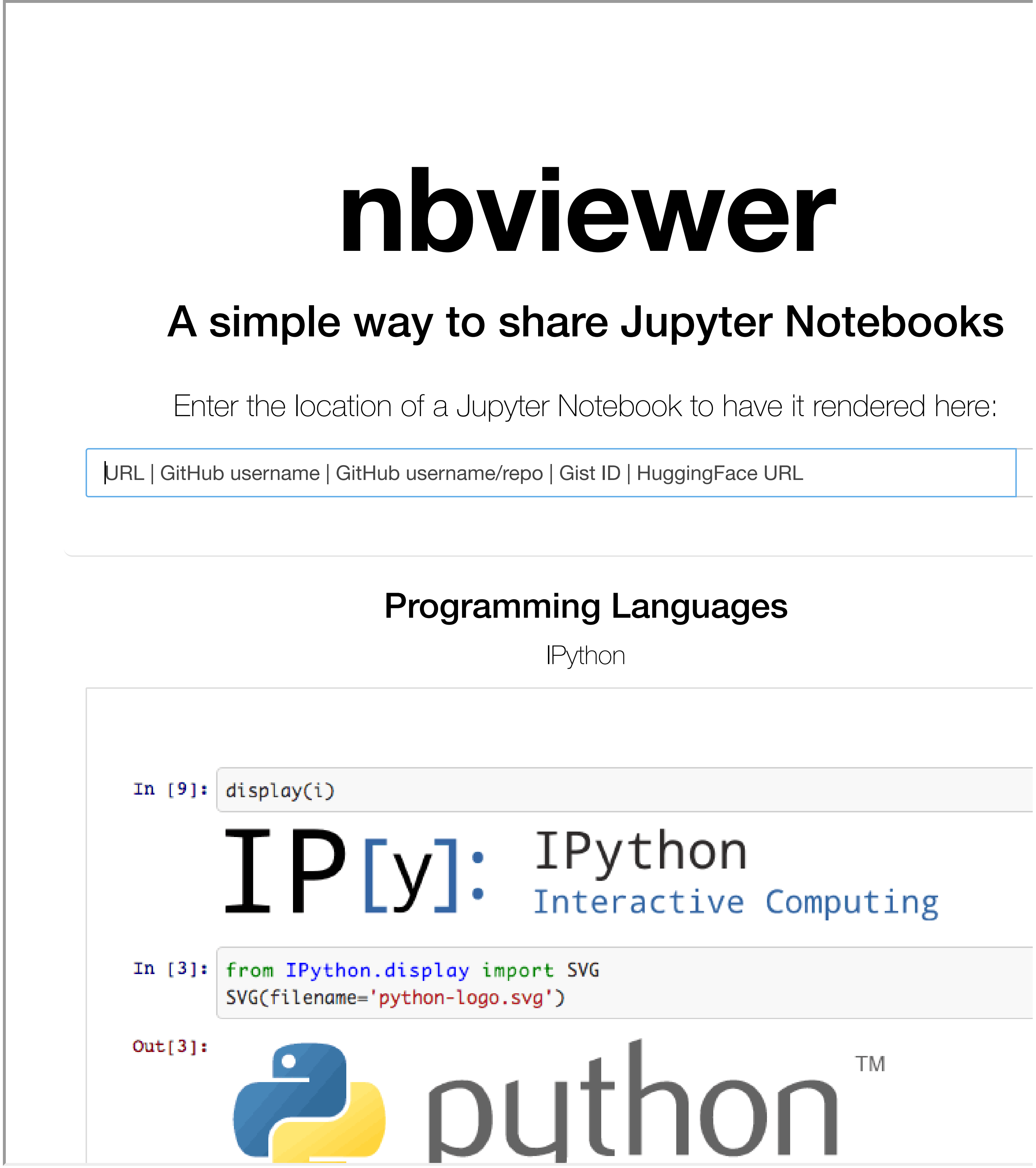
Sharing is caring

- Put the notebook on GitHub/Bitbucket and it will be rendered there...

- ... or export to one of many different formats, *e.g.* HTML, PDF, code, **slides** etc. (this presentation is a Jupyter notebook)

Or paste a link to any Jupyter notebook at nbviewer.jupyter.org and it will be rendered for you.

In [10]:
```
%%html
<!-- MRSA Notebook that you'll work on in the tutorial -->
<!-- https://github.com/NBISweden/workshop-reproducible-research/blob/main/tutorials/jupyt
<iframe src="https://nbviewer.jupyter.org/" height="800" width="800"></iframe>
```



nbviewer

A simple way to share Jupyter Notebooks

Enter the location of a Jupyter Notebook to have it rendered here:

URL | GitHub username | GitHub username/repo | Gist ID | HuggingFace URL

Programming Languages

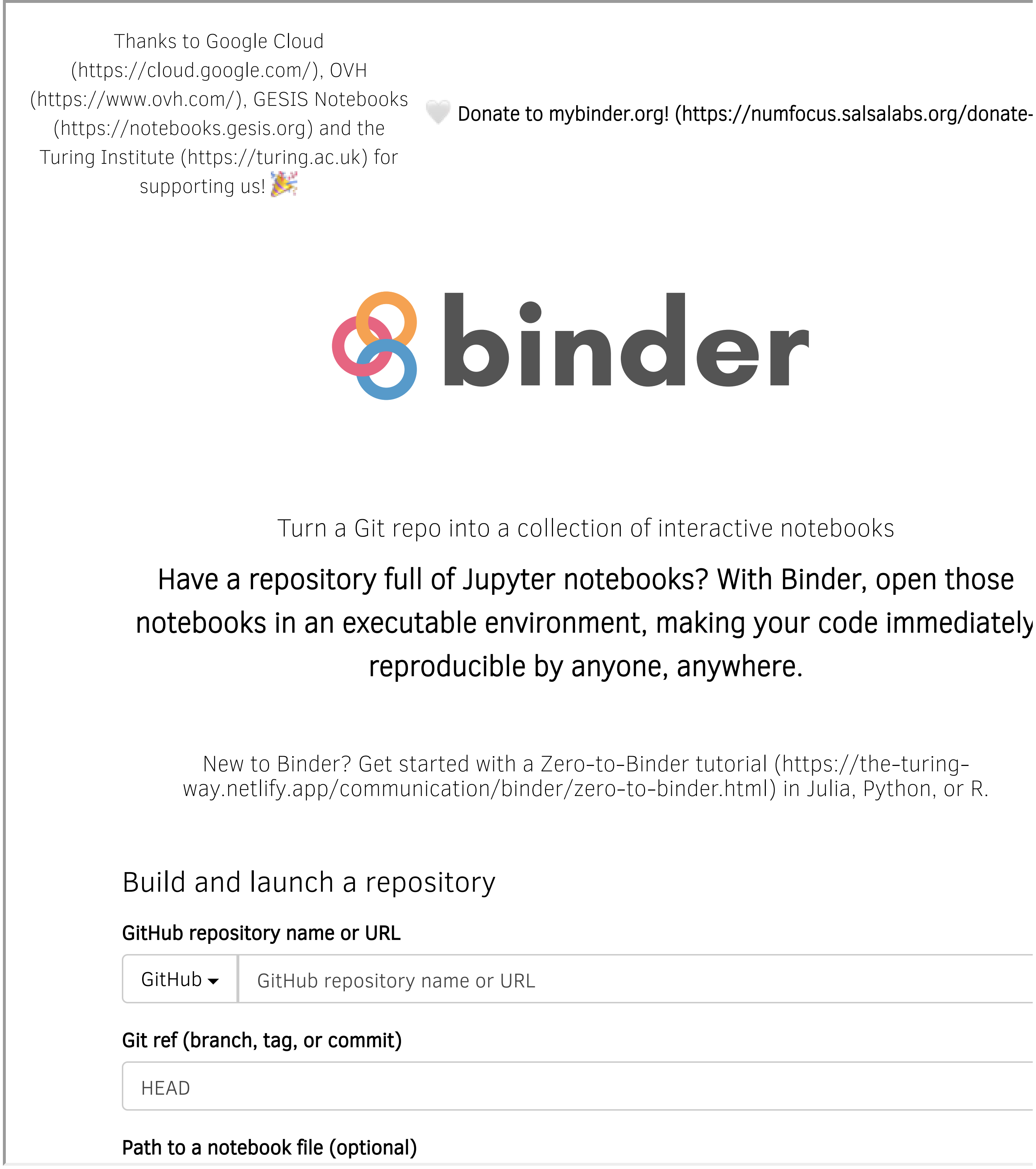
IPython

```
In [9]: display(i)
```

```
In [3]: from IPython.display import SVG
        SVG(filename='python-logo.svg')
Out[3]:
```

Or generate interactive notebooks using Binder

```
In [11]: %%HTML
         <iframe src="https://mybinder.org" height="800" width="800"></iframe>
```



Thanks to Google Cloud (https://cloud.google.com/), OVH (https://www.ovh.com/), GESIS Notebooks (https://notebooks.gesis.org) and the Turing Institute (https://turing.ac.uk) for supporting us! 🎉

🤍 Donate to mybinder.org! (https://numfocus.salsalabs.org/donate-

Turn a Git repo into a collection of interactive notebooks

Have a repository full of Jupyter notebooks? With Binder, open those notebooks in an executable environment, making your code immediately reproducible by anyone, anywhere.

New to Binder? Get started with a Zero-to-Binder tutorial (https://the-turing-way.netlify.app/communication/binder/zero-to-binder.html) in Julia, Python, or R.

Build and launch a repository

GitHub repository name or URL

GitHub ▾ | GitHub repository name or URL

Git ref (branch, tag, or commit)

HEAD

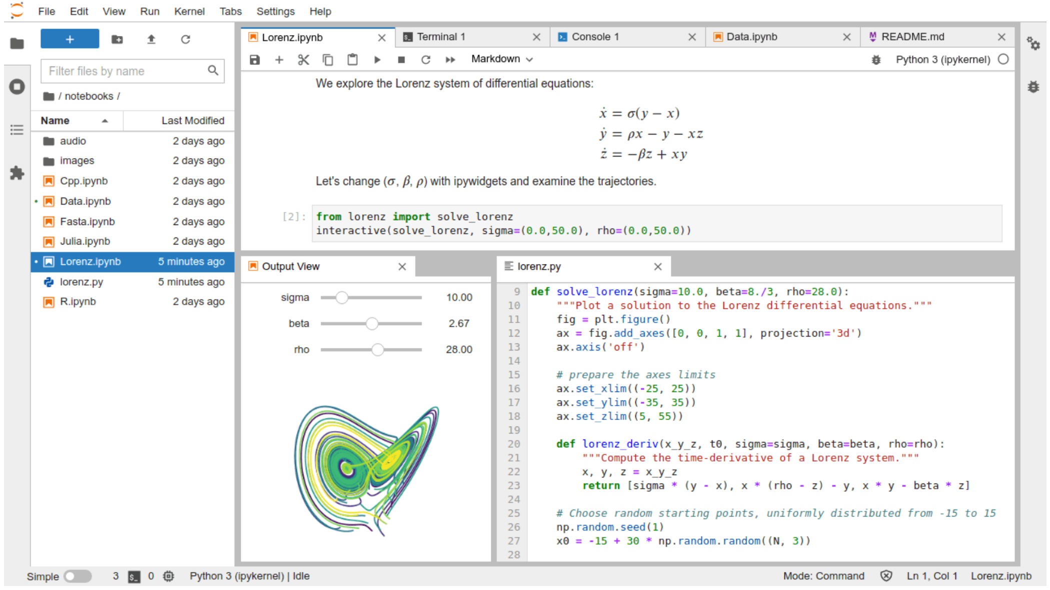
Path to a notebook file (optional)

Binder can generate a *'Binder badge'* for your repo. Clicking the badge launches an interactive version of your repository or notebook.

launch | binder

Jupyter Lab

JupyterLab is the next-generation web-based user interface for Project Jupyter.



Name	Last Modified
audio	2 days ago
images	2 days ago
Cpp.ipynb	2 days ago
Data.ipynb	2 days ago
Fasta.ipynb	2 days ago
Julia.ipynb	2 days ago
Lorenz.ipynb	5 minutes ago
lorenz.py	5 minutes ago
R.ipynb	2 days ago

We explore the Lorenz system of differential equations:

$$\dot{x} = \sigma(y - x)$$
$$\dot{y} = \rho x - y - xz$$
$$\dot{z} = -\beta z + xy$$

Let's change (σ, β, ρ) with ipywidgets and examine the trajectories.

```
[2]: from lorenz import solve_lorenz
     interactive(solve_lorenz, sigma=(0.0,50.0), rho=(0.0,50.0))
```

sigma	10.00
beta	2.67
rho	28.00

```python
def solve_lorenz(sigma=10.0, beta=8./3, rho=28.0):
    """Plot a solution to the Lorenz differential equations."""
    fig = plt.figure()
    ax = fig.add_axes([0, 0, 1, 1], projection='3d')
    ax.axis('off')

    # prepare the axes limits
    ax.set_xlim((-25, 25))
    ax.set_ylim((-35, 35))
    ax.set_zlim((5, 55))

    def lorenz_deriv(x_y_z, t0, sigma=sigma, beta=beta, rho=rho):
        """Compute the time-derivative of a Lorenz system."""
        x, y, z = x_y_z
        return [sigma * (y - x), x * (rho - z) - y, x * y - beta * z]

    # Choose random starting points, uniformly distributed from -15 to 15
    np.random.seed(1)
    x0 = -15 + 30 * np.random.random((N, 3))

```

- full-fledged IDE, similar to e.g. Rstudio.
- Tab views, Code consoles, Show output in a separate tab, Live rendering of edits

```
conda install -c conda-forge jupyterlab
```

jupyter{book}

lets you build an online book using a collection of Jupyter Notebooks and Markdown files

- Interactivity
- Citations
- Build and host it online with GitHub/GitHub Pages...
- or locally on your own laptop

For the tutorial:

`use jupyter notebook or jupyter lab`

Questions?