JupyterNotebookSetup

July 16, 2021

0.1 Jupyter Notebook learning environment

- Jupyter Notebook (popular tool in Data Science and Machine Learning) is merely a learning environment
  - not a professional development environment esp. for C++ programs
- using Jupyter Notebook you can interactively learn many programming languages such as Python, C, Rust, NodeJS, Bash, etc.
- allows you to interactively execute code, take notes with text and HTML, and embed multimedia files (image, audio, video, etc.)
- most importantly, one can execute codes line by line and save the output result
  - can focus on the the line of code to really master all the details and basics
- you can also read and execute these notebooks from VS Code with right extensions
- pdf version of these notebooks are provided in pdfs folder in the GitHub repository
  - pdfs are readonly; can't write and/or execute embedded code in PDF files

0.2 Setup Jupyter Notebook Environment

- it's recommended to install Jupyter Notebook program on your personal computer
- you'll also need the xeus-cling C++ kernel to interpret the C++ code and see the results
- follow the instructions from `https://github.com/rambasnet/DevEnvSetup` to setup Jupyter Notebook on various platforms

0.3 Online Rendering and Executing Jupyter Notebooks

- GitHub (most of the time) renders these notebooks, so one can read the contents directly from GitHub repo
  - readonly; can't execute embedded code on GitHub
  - rendered output is good enough but not as great
- jupyter notebooks can be better rendered in read-only mode using mybinder.org
- jupyter notebooks can be interactively executed as well using Binder
  - Go to GitHub Repository and click launch binder logo on the README page

[1]:
```
// in Jupyter Notebook, main() is defined implicitly per notebook!
// simply, include libraries and write code to execute without main()
#include <iostream>
```

[3]:
```
std::cout << "Hello World!" << std::endl;
```

```
Hello World!
```

[4]:
```
// include this line so you don't have to type std::
using namespace std;
```

[5]:
```
// by running the above statement, you don't need to type std:: after many␣
↪symbols
cout << "Hello World!" << endl;
```

```
Hello World!
```

[6]:
```
cout << "Good bye..." << endl;
```

```
Good bye...
```