JOD Labs in Jupyter

January 29, 2019

1 JOD Labs in Jupyter

1.0.1 J Labs and Jupyter

For many years J systems included a "labs" facility. J labs descended from even earlier APL labs. Labs exploited the interactive nature of APL, a radical facility in the 1960s, and the pioneers of APL, particularly Kenneth Iverson, made heavy use of labs for teaching APL and related subjects.

The idea behind labs is simple: we learn best by doing!

Nobody expects portrait painters to master painting by reading about it, attending tedious lectures, or watching boob-tube videos. Painters have to get their hands dirty and paint! Yet, in the early days of computing, programmers were expected to master programming by reading about it and attending even more tedious lectures.

Labs were created to ameliorate this ridiculous situation.

1.0.2 Jupyter Notebooks are Labs on Steroids

The modern Jupyter notebook has extended interactive "Labs". Well crafted Jupyter notebooks are excellent live texts. They encourage wide randing interactive exploration, documentation, and illustration.

When I first encountered Jupyter notebooks I immediately thought, "Hey this would be a great delivery mechanism for J labs."

To use a particular programming language with Jupyter you a need a "kernel." ***The single best design decision the Jupyter notebook developers made was to create a "kernel agnostic notebook."***

Notebooks, like labs, are not new. Mathematica notebooks have been around for decades and there were *notebook'ey* software systems even before Mathematica. Remember, the word "notebook" refers to *ancient* paper, parchment and papyrus documents.

What distinguishes Jupyter from other notebooks systems is its open-source nature. The Jupyter project encourages developers to create kernels for a variety of programming languages and many have responded. You can find Jupyter kernels for scores or programming languages including J.

1.0.3 Martin Saurer's J Kernel

Martin Sauer implemented a J 8.04 kernel: see this Github repository.

His kernel works with later versions of J. I am using it with J 8.07. To execute JOD Jupyter labs you need to:

1. Install Jupyter. I recommend Anaconda.
2. Install a current version of J.
3. Use the J package manager to install the J addons, jod, jodsource, and joddocument.
4. Follow Martin's J kernel instructions to install a J kernel on your machine.
5. Download the JOD Jupyter labs and save them in a Jupyter directory.

I know it's a lot of hoops to jump through but having a working J/Jupyter environment is worth it. You can judge for yourself by browsing the following JOD addon labs that have been converted to Jupyter notebooks.

1. JOD Introduction Lab
2. JOD Source Code Dump Scripts Lab
3. JOD Best Practices Lab

1.0.4 Some differences between JOD labs in J and Jupyter.

If you are familiar with how J labs work in J front ends like JQT or JHS you will see some differences when running J labs in Jupyter.

Use portable box drawing characters. J box characters, used to display the structure of nested arrays, are a persistent pain. Many fonts do not have box characters and if present they may not line up properly.

In Jupyter notebooks box characters make it more difficult to *download* notebooks as `LateX` files. Check out the options under the Jupyter `File` menu.

The simplest way to avoid box character hell is to use standard ASCII characters. (`portchars`) defines a set of box characters that work in most contexts.

```
In [1]: NB. use portable box drawing characters
        NB. simplifies rendering notebooks as (*.tex)
        portchars_ijod_=:[: 9!:7 '+++++++++|-'"_ [ ]
        portchars_ijod_ ''
```

Jupyter notebooks can be easily converted to a variety of formats. If you have taken care to use portable box drawing characters Jupyter notebooks can be easily converted to various formats. I've converted these notebooks to `tex` and `pdf`. See this Git directory.

https://github.com/bakerjd99/jod/blob/master/jodnotebooks/

Jupyter only displays the last result of a code block.

```
In [2]: NB. no visible ouput
        ;:'not by the hair of my chinny chin chin'

        NB. only the last statement displays output
        i. 2 3

0 1 2
3 4 5
```

To see other statements use `smoutput`

```
In [3]: NB. use smoutput to show result
        smoutput ;:'not by the hair of my chinny chin chin'

        NB. only the last statement displays output
        i. 2 3
```

```
+---+--+---+----+--+--+------+----+----+
|not|by|the|hair|of|my|chinny|chin|chin|
+---+--+---+----+--+--+------+----+----+
0 1 2
3 4 5
```

Large deeply nested outputs may wrap poorly

```
In [4]: NB. <^:10 ] (i. 30 50) ; < ;:'+/@, 34 5'
```

J plot graphics work but forms that request user input may not

```
In [5]: load 'plot'
        'bar' plot i. 5
```

1.0.5 Final words

Jupyter versions of J labs provide a richer environment for experimenting with and documenting J systems.

It isn't difficult to convert existing J labs to Jupyter notebooks and since both J labs and Jupyter notebooks are stored as simple text it wouldn't be difficult to implement an addon that converted J labs to Jupyter notebooks. The inverse, while more challenging, is also feasible.

```
In [ ]:
```