Jupyter Graduates

Douglas Blank, Ph.D.
with
Kara Breeden, B.A. & Nicole Petrozzo, B.A.
Bryn Mawr College
Jupyter Graduates
Jupyter Graduates

The title of this talk can be parsed in two ways:

1. <NOUN> <VERB>
2. <ADJECTIVE> <NOUN>
Jupyter Graduates

The title of this talk can be parsed in two ways:

1. <NOUN> <VERB> - Jupyter has graduated and is ready for prime time
2. <ADJECTIVE> <NOUN> - Students have graduated via Jupyter
Jupyter Graduates

The title of this talk can be parsed in two ways:

1. **<NOUN> <VERB>** - Jupyter has graduated and is ready for prime time
2. **<ADJECTIVE> <NOUN>** - Students have graduated via Jupyter

I mean both of them.
Jupyter Graduates

The title of this talk can be parsed in two ways:

1. <NOUN> <VERB> - Jupyter has graduated and is ready for prime time
2. <ADJECTIVE> <NOUN> - Students have graduated via Jupyter

I mean both of them.

Four years ago "Jupyter" got its name and the "JupyterHub" project was created. Four years later, the Jupyter Project is a solid, well-defined platform. It has left "beta" status and has graduated to become a mature community.

Four years ago, a group of students started their college careers at Bryn Mawr College and I began exclusively using Jupyter in the classroom. This Spring, two of these students finished their senior theses with Jupyter, graduated, and are mature members of the community.
Four years and scores of notebooks ago...
Four years and scores of notebooks ago...


Calico: A Multi-Programming-Language, Multi-Context Framework Designed for Computer Science Education

Douglas Blank¹, Jennifer S. Kay², James B. Marshall³, Keith O'Hara⁴, and Mark Russo⁴

¹Computer Science Department, Bryn Mawr College, Bryn Mawr, PA (USA) (1) 610-526-6501 dbblank@brynmawr.edu
²Computer Science Department, Rowan University, Glassboro, NJ (USA) (1) 856-256-4593 kay@rowan.edu
³Computer Science Department, Sarah Lawrence College, Bronxville, NY (USA) (1) 914-395-2673 jmarshall@slc.edu
⁴Computer Science Program, Bard College, Annandale-on-Hudson, NY (1) 845-752-2359 kohara@bard.edu
I was working on a multi-language environment for education.

- Easy to use on all platforms; no dependency issues
- Libraries could be used natively by any of the languages
- Common API across languages
- Debugger, stepper
- Built-in graphics (all in one context), and sound
- Written in C#
- Robots!
Four years and scores of notebooks ago...

I was working on a multi-language environment for education.

- Easy to use on all platforms; no dependency issues
- Libraries could be used natively by any of the languages
- Common API across languages
- Debugger, stepper
- Built-in graphics (all in one context), and sound
- Written in C#
- Robots!
Four years and scores of notebooks ago...

I was working on a multi-language environment for education.

- Easy to use on all platforms; no dependency issues
- Libraries could be used natively by any of the languages
- Common API across languages
- Debugger, stepper
- Built-in graphics (all in one context), and sound
- Written in C#
- Robots!

*7:00 PM Tonight [Gotham Comedy Club](http://www.example.com)*
Four years and scores of notebooks ago...

January 2014: Have seen what they have been doing to IPython lately?

- multi-language
- in the browser
- written in Python
Four years and scores of notebooks ago...

January 2014: Have seen what they have been doing to IPython lately?

- multi-language
- in the browser
- written in Python


Four years and scores of notebooks ago...

4. July 2014, Made the decision to try JupyterHub in the Fall.
5. September 2014, I began teaching two courses with JupyterHub.
First Public JupyterHub Deployment

Min Ragan-Kelley

Welcome to Jupyter!
To login into Jupyter, enter the username and password provided to you by your instructor in the fields on the left. This is not the same as your school/department login. If you would like an account or need assistance, please email Matt Rice.

You may also be interested in services on our supercomputer, Athena.

Useful Links
- Jupyter Notebook Users Manual

News
- New Jupyter computer installed and operational! September 1, 2017.

Courses using Jupyter
  - 2017-2018
    - CS110: Introduction to Computing, Fall 2017
    - CS240: Principles of Computer Organization, Fall 2017
  - 2016-2017
    - CS110: Introduction to Computing, Spring 2017
    - CS206: Data Structures, Spring 2017
    - CS245: Principles of Programming Languages, Fall 2016
    - CS371: Introduction to Cognitive Science, Fall 2016
  - 2015-2016
    - Physics 250: Computational Methods, Spring 2016
    - BioCS115: Computing through Biology, Spring 2016
    - CS206: Data Structures, Spring 2016
    - CS110: Introduction to Computing, Fall 2015
    - CS240: Principles of Computer Organization, Fall 2015
  - 2014-2015
    - CS110: Introduction to Computing, Spring 2015
    - CS245: Principles of Programming Languages, Fall 2014
    - ESEM: Humanity and Technology, Fall 2014
Four Years of Jupyter and Students in the Classroom

Five levels of integration:

- Modules
- Kernels
- Magics
- Extensions
- Services

Source code found under the GitHub Calysto* project.

* not the Canadian one
Current JupyterHub Deployment

Two JupyterHub services:

1. "Public" - a built-in nbviewer-like rendering of notebooks in public_html
2. "Accounts" - a form for administrators to quickly create accounts, XKCD passwords, and send email notifications

github.com/BrynMawrCollege/jupyterhub
Using Jupyter in the cloud is a commitment to equity - no one has a better "compute engine" than anyone else.

Even cell phones and tablets can work if necessary.

(It is also really convenient! Does require an internet connection.)
First Courses/First Students

- **Firstyear Writing Seminar Experience**
  - 11 Students
  - 3 became CS majors
  - 2 used Jupyter in their senior theses
  - Class used Jupyter to write a short paper, with some data analysis

- **Principles of Programming Languages**
  - 12 students
  - Class used Jupyter with Scheme and Python

Kara Breeden
Nicole Petrozzo
Course: Firstyear Writing Seminar

- Used the Python kernel, with Google Charts magics
- Spelling Checker and Document Tools

I was hoping to build on their knowledge of writing text/markdown to bootstrap into "computational thinking."

1. top to bottom ordering
2. shift+enter render, or execute
3. cut and paste

The notebook interface was a shock. Students forgot how to write. Challenging problem.

But, I still have a vision that our firstyear writing seminar could embrace some data visualization & analysis, if not computational thinking.
Notebook Extensions

- Publish
- Submit
- Spelling Checker
- Move Section Up/Down
- Number Sections
- Table of Contents
- Generate References
- Tabbed In/Out
- Two-column In/Out
- Drawing Annotations

github.com/Calysto/notebook-extensions
Course: Programming Languages

- Calysto Scheme
  - Written in Scheme
  - Transpiled into C#/Python
  - Can use Python libraries
  - Visual Stepper/Debugger
  - Real Scheme: call/cc, etc.
  - Proper tail call recursive

- Main goals:
  - Programs are data
  - Implement Scheme in Python
  - Implement Python in Scheme

Not previously possible!
Four Years of Bryn Mawr College Courses using Jupyter

- **2017-2018**
  - CS110: Introduction to Computing, Fall 2017
  - CS240: Principles of Computer Organization, Fall 2017

- **2016-2017**
  - CS110: Introduction to Computing, Spring 2017
  - CS206: Data Structures, Spring 2017
  - CS245: Principles of Programming Languages, Fall 2016
  - CS371: Introduction to Cognitive Science, Fall 2016

- **2015-2016**
  - Physics 250: Computational Methods, Spring 2016
  - BioCS115: Computing through Biology, Spring 2016
  - CS206: Data Structures, Spring 2016
  - CS110: Introduction to Computing, Fall 2015
  - CS240: Principles of Computer Organization, Fall 2015

- **2014-2015**
  - CS110: Introduction to Computing, Spring 2015
  - CS245: Principles of Programming Languages, Fall 2014
  - ESEM: Humanity and Technology, Fall 2014
Nothing but Notebooks for 4 Years

- **Processing** - Introduction to Computing
- **Assembly Language** - Computer Organization
- **Java** - Data Structures
- Python - **Computing through Biology**
- Python - **Introduction to Cognitive Science**

Every syllabus, "lecture", handout, assignment, live-coding, in-class exercise, and example was a notebook available via JupyterHub.

(Exams were still on paper.)
Nothing but Notebooks for 4 Years

- **Processing** - Introduction to Computing
- **Assembly Language** - Computer Organization
- **Java** - Data Structures
- **Python** - Computing through Biology
- **Python** - Introduction to Cognitive Science

First day of class: 
Writing code within 20 minutes!
Course: Introduction to Computing, Lab1
Nothing but Notebooks for 4 Years

- Processing - Introduction to Computing
- Assembly Language - Computer Organization
- Java - Data Structures
- Python - Computing through Biology
- Python - Introduction to Cognitive Science

Machine Language, and Assembly Language for the LC3

1.1.3 Putting it all together

Now we just put the .ORIG directive, the machine code, and you have the next cell highlighted:

```
In [ ]:  .ORIG x3000
        0001 000 000 1 00001
        1111 0000 00100101
    .END
```

You just assembled your first machine language program!
Nothing but Notebooks for 4 Years

- **Processing** - Introduction to Computing
- **Assembly Language** - Computer Organization
- **Java** - Data Structures
- Python - **Computing through Biology**
- Python - **Introduction to Cognitive Science**
Nothing but Notebooks for 4 Years

Joshua Shapiro, Bryn Mawr College Biology
The medium is the message: a %processing "sketch" magic to play Jeopardy is a demo, and a good example of data separation (CSV file), but also used for reviews. And a bit of fun!
Metakernel

- Common magics (meta-commands, starts with percentage symbol)
- Clear separation of meta-commands from language
- Provides magics, parallel processing, and shell for all languages
  - Magics can be used in IPython, too!
- Popular Metakernel Languages include: Octave, Matlab, Scheme, Prolog, Processing, Java9, Xonsh, ROOT, Gentoo Science Bash, Cling, Wolfram, Hy, Assembly Language and more!

[github.com/Calysto/metakernel](https://github.com/Calysto/metakernel)
Metakernel

- Magics (on IPython, too), include:
  - %activity - exit tickets
  - %download - download a file and unzip
  - %tutor - stepper and visualizer
  - %jigsaw - Blockly + Python/Java

- Sticky magics
  - triple-%, applies to all following cells

github.com/Calysto/metakernel
%activity magic allows quizzes, surveys, or "exit ticket"-style interactions.

Metakernel

Which of the following will print "Hello" 5 times without errors?

1) for (int i=0; i < 5; i++) {
   println("Hello");
}

2) println("Hello");
println("Hello");
println("Hello");
println("Hello");
println("Hello");

3) int i = 0;
   while (i < 5) {
      println("Hello");
      i++;
   }

4) All of the above
5) None of the above

Respond: 1 2 3 4 5
Metakernel

In [2]: %jigsaw Python --workspace workspace1

```
import pyfirmata

set pin 13 to expression 13
set delay 1 to expression 1
set port /dev/ttyACM0 to expression 
set board pyfirmata.Arduino

repeat 2 times

for

don

expression board.digital[pin].write(1)
expression board.pass_time(delay)
expression board.digital[pin].write(0)
expression board.pass_time(delay)
```

Run  Generate Python Code
"High school students learning programming do better with block-based languages, and the impact is greatest for female and minority students" - Mark Guzdial
Nothing but Notebooks for 4 Years

- **Processing** - Introduction to Computing
- **Assembly Language** - Computer Organization
- **Java** - Data Structures
- Python - **Computing through Biology**
- Python - **Introduction to Cognitive Science**
Library for Deep Learning

- Interface for humans
- Uses lists of numbers
- Visualizations/Movies
- Free of ML jargon
- Dashboard
- Built on Keras/TensorFlow
- Designed to exploit Jupyter
- Follows Python's Zen

ConX - /kəˈnektz/
conx.readthedocs.io
The Zen of Python

Beautiful is better than ugly.
Explicit is better than implicit.
Simple is better than complex.
Complex is better than complicated.
Flat is better than nested.
Sparse is better than dense.
Readability counts.
Special cases aren't special enough to break the rules.
Although practicality beats purity.
Errors should never pass silently.
Unless explicitly silenced.
In the face of ambiguity, refuse the temptation to guess.
There should be one-- and preferably only one --obvious way to do it.
Although that way may not be obvious at first unless you're Dutch.
Now is better than never.
Although never is often better than *right* now.
If the implementation is hard to explain, it's a bad idea.
If the implementation is easy to explain, it may be a good idea.
Namespaces are one honking great idea -- let's do more of those!
Software Design

What's missing?

- Explain
- Teach
- Understand

Source: https://en.wikipedia.org/wiki/Software_development_process#/
Software Design

How does your design support Bloom's Taxonomy?
I have used Jupyter for class projects, papers, my thesis, and internship work. Jupyter is very useful for testing out new code because I can make tiny changes and see the output easily/immediately. I also found it useful that I can scroll up and see the results of different cells to compare results.

Jupyter is great for small and large projects. It is also really helpful at keeping your work organized and commented in a way that you and others can follow.
Nicole Petrozzo

As a student, there were two features of Jupyter that I found extremely helpful. First, the **ability to run code line by line** is definitely very easy and convenient compared to any other IDE I have worked in before. It's so easy to test functions or check the values of any variable, and if you mess something up, all you have to do is click a button to **restart the kernel**. The second feature I really love about Jupyter is the **graphical capabilities**.

For certain projects, like my thesis for example, **this was a godsend**. It was really convenient and **intuitive to have my code and a visual, graphical representation of the results all in the same spot**.
Kara Breeden

I think Jupyter could use better short cuts. I can never remember them.
I really wouldn't have been able to complete my thesis or my internship project without Jupyter. It's a great tool and it was easy for me to pick up, even as a novice computer scientist.

It would have been useful if there were more collaborative capabilities. When I worked on a notebook with a partner our only option was to sit next to each other and take turns typing on the same laptop.
Jupyter Graduates

Kara, Nicole, and their classmates have graduated! They were able to experience computer science in novel ways because of Jupyter.

Jupyter started as a seed. But it has graduated to a full-scale platform for education.

But Kara's, Nicole's, and Jupyter's stories are just beginning.
Additional Resources

1. JupyterDayPhilly: Transformative Teaching with the Jupyter Notebook
4. Bryn Mawr College CS Senior Conference - student theses
5. Engage-CSEdu.org - curated, selected assignments and activities demonstrating best-practices, including those of equity and diversity