Firefly, Python, JupyterLab and the Science Platform
Science Platform

- Concept is catching fire in astronomy community

- Working definition of Science Platform in nutshell
  - Login to a Jupyter Lab/Hub environment
  - Appropriate software and APIs
  - Close data is archived
  - Processing close to the data
  - Usually with python.

- Our Goal:
  - Leverage out work with Firefly to make this environment more powerful
New Ways to Work

Firefly + Python + Jupyter Lab = Powerful Science Platform
**Firefly Components**

- WCS Readout
- Zoom
- Flip/ Rotate/ Crop
- Color / Stretch
- Grid
- Region
- Magnifier
- Distance tools
- Markers
- Fits Headers
- Crop

- Full HiPS Support
- MOC
- Tightly integrated
- Shares all FITS functions

- Sort / Filter
- Column Controls
- Large tables, 10 Million+ rows
- Very fast response time
- brushing and linking

- Interactive
- Column math
- Zoom
- Filter
Firefly Library
Firefly Archive Visualization Library
Code Overview

Frontend
- JavaScript
- Modern JS
  ‣ ES6+, Modules
  ‣ NPM
  ‣ Webpack
- React/Redux
- Converted from GWT
  ‣ 2015 - 2016

Backend
- Java
- Tomcat
- Scalable
  ‣ Multiple instances
- JSON to Client
  ‣ GET, POST, Websocket
- Docker and Kubernetes

Frontend & Backend:
~245K Lines of code
Open Source and Releases

• GitHub
  - https://github.com/Caltech-IPAC/firefly

• Releases
  - More Formal Process
  - Builds on Docker
  - Release Notes
  - CCB - Yearly Roadmap

• Dockerized
  - Start firefly with one command
  - Tag for each release plus nightly
Ways to Use Firefly

1. Stand alone
   - Install and Run

2. Library for building Web Applications
   - Most Advanced: Work at the React/Redux level
   - Many IRSA application built from Firefly

3. JavaScript API
   - Firefly Widgets in a Web page

4. Remote API
   - Control a Firefly Application

   ➡ Key to interfacing with Python
   ➡ Start application & control it from Python
Remote API

- Firefly architecture is designed around the command pattern
- Each command tells the system to do something
  - Plot image
  - Zoom Image
  - Rotate Image
- Firefly can also listen for commands over channel
- HTTP communication and Web Sockets
- Commands sent as HTTP request
Remote API

Another Context

Examples:
- Load a FITS Image
- Load a Table
- Show a Chart
- Sort a table
- Show HiPS

Firefly
Web Server

Action
Network (http)

JSON Data

Action: {
  type: String,
  payload: {
    data
  }
}

Firefly
WebApp in Browser

Network (Web socket)
Python Binding
Connecting Python to Firefly

• https://github.com/Caltech-IPAC/firefly_client
• pip install firefly_client
• Connects to the firefly server via URL
• Hides network connectivity
• Implements API to control Firefly from Python
• Can do this in Jupyter Notebook or Lab

```python
from firefly_client import FireflyClient
fc = FireflyClient.make_client('firefly url')
fcs.launch_browser()
fcs.show_fits(URL='some url')
handle = fc.upload_file(a_file)
fcs.show_table(handle)
```
In [1]: from firefly_client import FireflyClient
....: fc = FireflyClient.make_client('127.0.0.1:8080/firefly')
```python
from firefly_client import FireFlyClient

fc = FireFlyClient.make_client('127.0.0.1:8080/firefly')
fc.launch_browser()
fc.show_fits(URL='http://web.ipac.caltech.edu/staff/roby/demo/wise-m51-band2.fits')
```
Jupyter Lab

• Jupyter Lab is Extendable

• Using Various Firefly API we have written extensions
  - https://github.com/Caltech-IPAC/jupyter_firefly_extensions

• Firefly will run in a Jupyter Lab tab

• Extensions:
  - FITS File Opener
  - Visualization Extensions using FireflyClient
  - Lab Widgets

• Result: A very integrated system
CLICK HERE TO BRING UP FIREFLY IN A LAB TAB
# Add table in cell 'main'
# 'main' is the cell id currently supported by Firefly for element type 'tables'
# This cell is shown at row = 0, col = 2 with width = 2, height = 2
r = fc.add_cell(id='main', 0, 2, 4, 2, 'tables', main=True)

if r['success']:
    fc.show_table tblId='wiseCatTbl', title='WISE catalog', target=1, layer='allwise_plx_5px_pm',
    position='target', showLimits=True, showName=True

layout_s = {'title': 'Color-Color',
    'xaxis': {'title': 'w1pro-w2pro (mag)'}
}
fc.show_chart group_id='viewer', layout=layout_s, data=table

# Add second chart - heatmap (plotly direct plot)
# in cell 0, 2, 3, 4
viewer_id = 'heatMapContainer'
fc.add_cell(id='heatMapContainer', 0, 2, 3, 'xplots', viewer_id=viewer_id)

if r['success']:
    dataMW = {
        'type': 'fireflyHeatmap',
        'title': 'w1 vs. w2',
        'tblId': 'wiseCatTbl',
        'query': {'tables': 'wpro',
            'colorscale': 'Blues'}
    }

    layout_he = {'title': 'Magnitude-magnitude densities',
    'xaxis': {'title': 'w1 photometry (mag)'}
    }

fc.show_chart 'heatmap', group_id='viewer', layout=layout_he, data=MW

fc.show_chart 'heatMapContainer', group_id='viewer', layout=layout_h, data=MW
Lesson learned: Jupyter Lab Extension

What went well

- Concept completely worked - Jupyter Lab is very flexible
- Lab uses modern JS build tools such as NPM and Webpack
  - Works with Firefly well.
- Lab did not conflict with Firefly - impressive for a complex tool
- We discovered issues with Firefly
  - Good way to test and improve the API
- We went though an upgrade cycle- is was fairly painless
Lesson learned: Jupyter Lab Extension

Challenges

- Very little extension documentation
- Difficult to do some fairly straightforward extension development
  - Look at examples (are the example correct?)
  - Get on Gitter, ask questions and hope someone will answer
  - Go thought the Lab code
- Extension development went slow
Firefly / Python / Jupyter Lab

- A lot of potential
- Fits nicely into a Web based Science Platform
- Opportunity to design many custom visualizations
- Much, much more we can do