Git, APIs, & Python

Hack Iowa
University of Iowa

John W. Miller
November 14th, 2017

Goal:

Show how easy it is to play with open source libraries and start grabbing data from public APIs

Overview

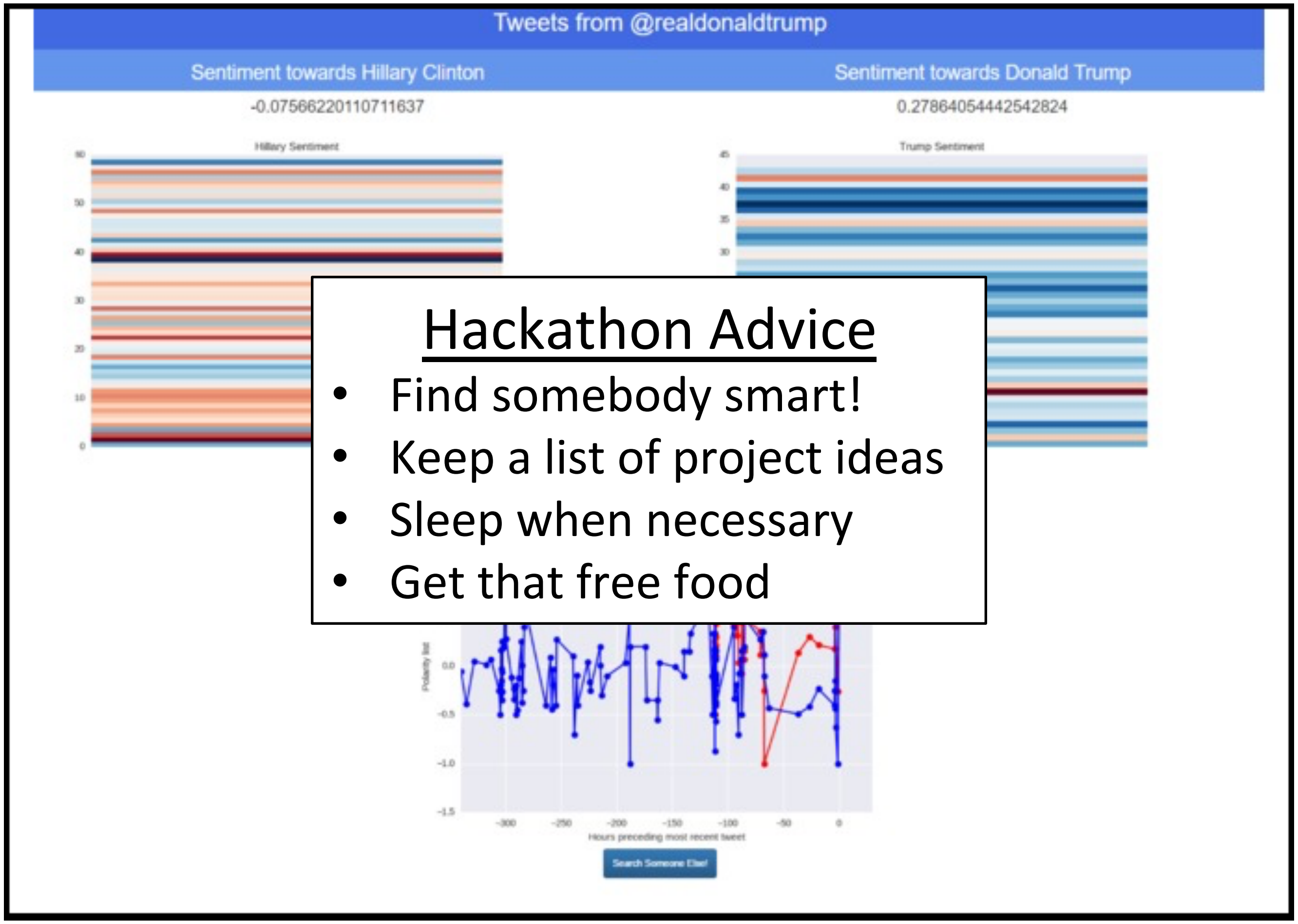
1. My background
2. Git and GitHub
3. APIs
4. Python and Jupyter notebooks
5. Demos

My background

- Physics @ Goshen College, graduated 2014
- Iowa Neurosurgery Department, 2014-2016
- ECE M.S. @ University of Iowa, will graduate 2018
  - Signal processing
  - Image processing
  - Machine learning

Twitter Political

(University of Iowa Hackathon 2016)



Tweets from @realdonaldtrump

Sentiment towards Hillary Clinton

-0.07566220110711637

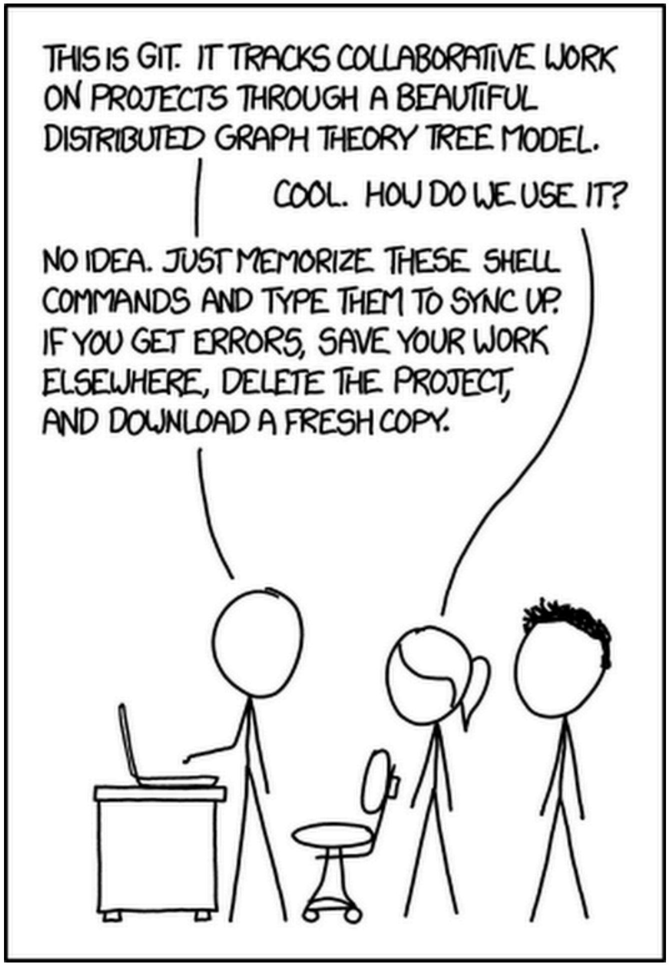
Sentiment towards Donald Trump

0.27864054442542824

Hackathon Advice

- Find somebody smart!
- Keep a list of project ideas
- Sleep when necessary
- Get that free food

Git & GitHub


THIS IS GIT. IT TRACKS COLLABORATIVE WORK ON PROJECTS THROUGH A BEAUTIFUL DISTRIBUTED GRAPH THEORY TREE MODEL.
COOL. HOW DO WE USE IT?
NO IDEA. JUST MEMORIZE THESE SHELL COMMANDS AND TYPE THEM TO SYNC UP. IF YOU GET ERRORS, SAVE YOUR WORK ELSEWHERE, DELETE THE PROJECT, AND DOWNLOAD A FRESH COPY.

Git & GitHub

- **Git** is a system for version control
- **GitHub** is a web-based platform for hosting and sharing Git repositories
- Other version control software exists
  - Apache Subversion (SVN)
  - Mercurial
  - Veracity

But GitHub has helped make Git very popular.

Git & GitHub

- Version control
  - Keeps record of your changes
  - Allows for collaborative development
  - Keeps details record of *who* made *what* changes and *when*
  - Allows you to revert to any and all changes **previously committed to** the repository

Git & GitHub

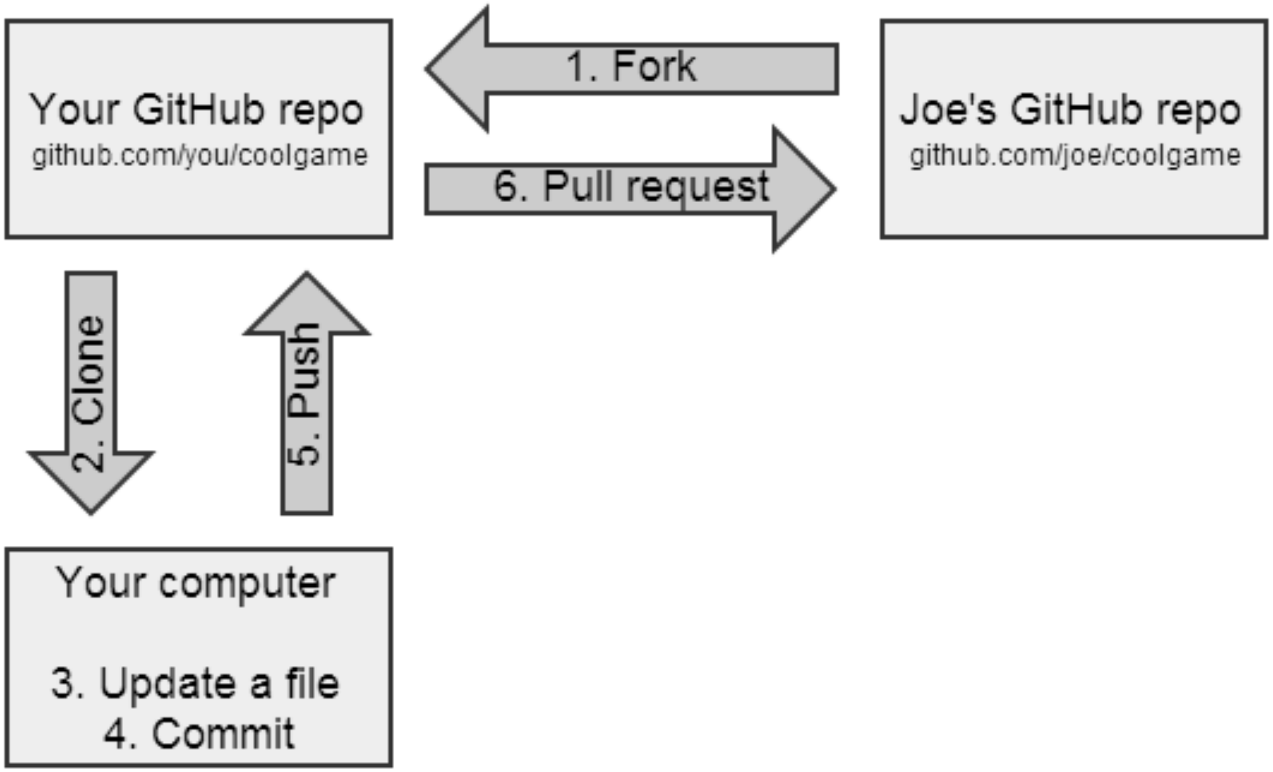
Typical workflow

1. Clone a repository from GitHub (or create your own)
2. Change the code, add & commit your changes
3. Pull any changes from the master repository
4. Push your committed changes to the remote repo

- Repeat steps 2-4 for the rest of your life.

*This is an example with a *very* limited scope, but it hopefully illustrates a simple common practice

Git & GitHub



Your GitHub repo

github.com/you/coolgame

1. Fork

6. Pull request

Joe's GitHub repo

github.com/joe/coolgame

2. Clone

5. Push

Your computer

3. Update a file

4. Commit

http://www.dataschool.io/simple-guide-to-forks-in-github-and-git/

Git concepts

- Repository
- Commit
- Push
- Branch
- Pull
- Fork
- Merge

Git concepts

- Repository
  - Stores current and historical code, “repo” for short
  - Can be local or remote
- Commit
- Push
- Branch
- Pull
- Fork
- Merge

Git concepts

- Repository
- Commit
  - Used to save modified code to the repo
- Push
- Branch
- Pull
- Fork
- Merge

Git concepts

- Repository
- Commit
- Push
  - Sends committed changes to the remote repo
- Branch
- Pull
- Fork
- Merge

Git concepts

- Repository
- Commit
- Push
- Branch
  - – Separate path for new code, can later be merged into main, “master,” branch
- Pull
- Fork
- Merge

Git concepts

- Repository
- Commit
- Push
- Branch
- Pull
  - Request to add modified code from one branch to another
- Fork
- Merge

Git concepts

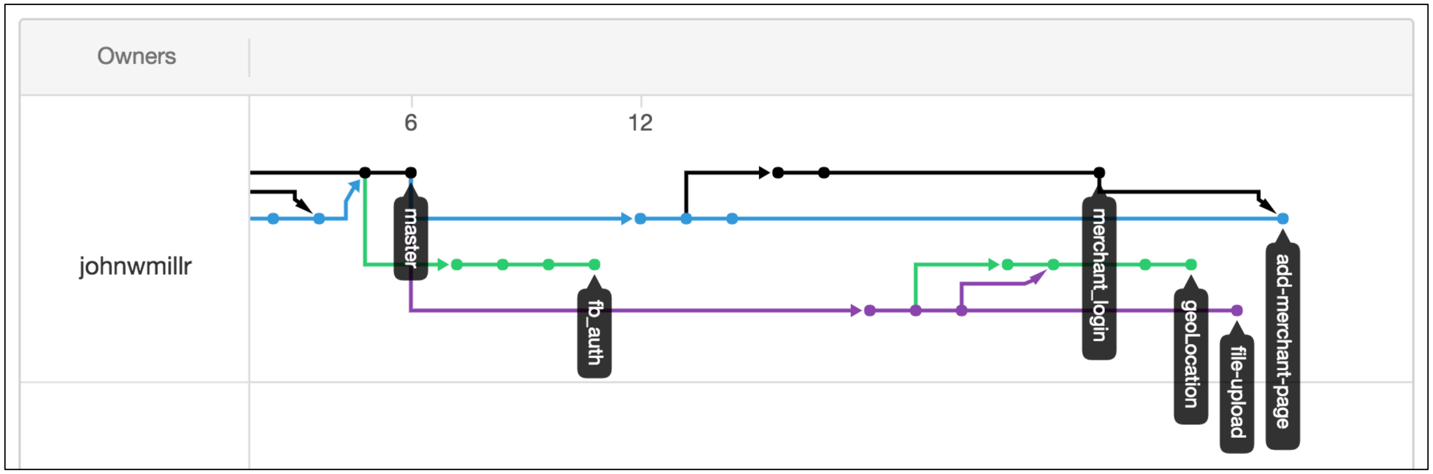
- Repository
- Commit
- Push
- Branch
- Pull
- Fork
  - Diverging copy of a repo, for new development
- Merge

Git concepts

- Repository
- Commit
- Push
- Branch
- Pull
- Fork
- Merge
  - Combine code from two branches

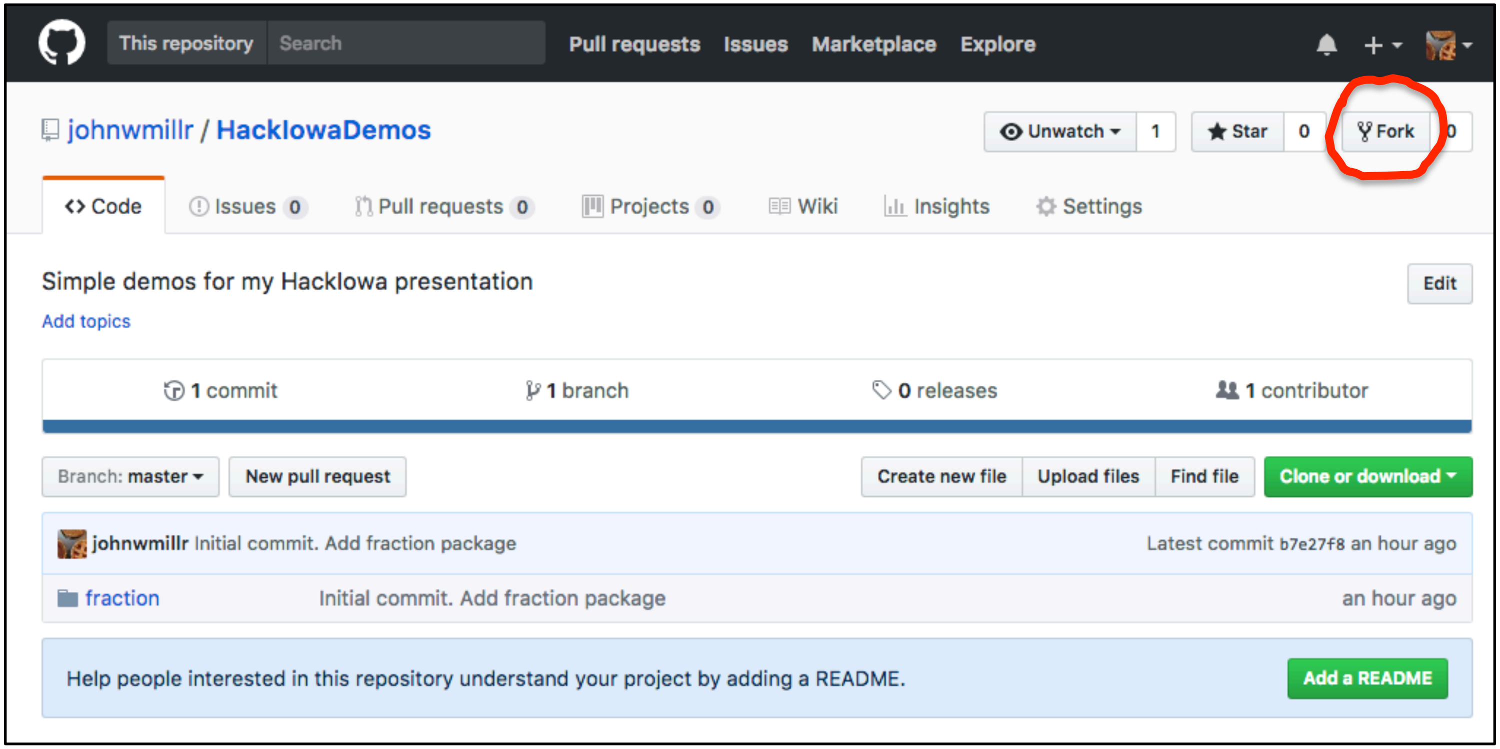
Git & GitHub

Project branches



GitHub Demo

1. Fork the repo
   - Link: https://github.com/johnwmillr/HackIowaDemos



Simple demos for my HackIowa presentation

johnwmillr Initial commit. Add fraction package — Latest commit b7e27f8 an hour ago

fraction — Initial commit. Add fraction package — an hour ago

Help people interested in this repository understand your project by adding a README.

GitHub Demo

2. Clone your forked repo

```
$git clone https://github.com/<your-name>/HackIowaDemos.git
$cd HackIowaDemos
$git status
```

3. Try out the fraction class

```
$cd fraction
$python
>>>from fraction import Fraction
>>>f1 = Fraction(1,2); print(f1)
>>>f2 = Fraction(1,6); print(f2)
>>>f1+f2
>>>f1.eval
```

GitHub Demo

4. Make a new branch

```
$git checkout –b fix-fraction-eval
```

5. Edit the code (make sure it works!)
6. **Add** and **commit** your changes

```
$git status
$git add .
$git status
$git commit –m “Fix the self.eval bug in fraction class”
```

7. Merge your modified branch into master

```
$git checkout master
$git merge fix-fraction-eval
$git log
```

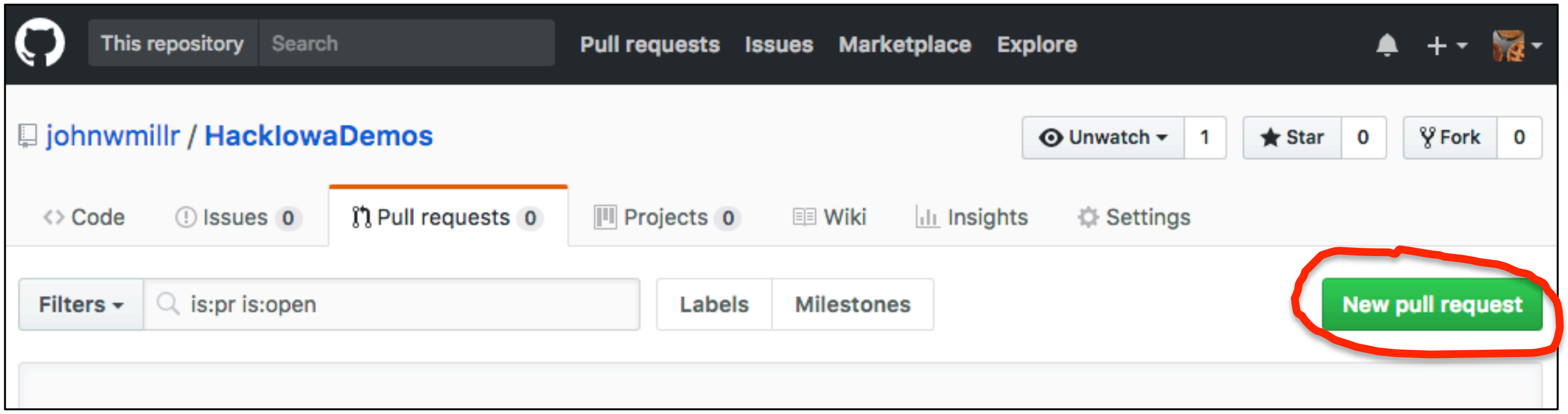
GitHub Demo

8. Push your changes to your remote repository

```
$git push origin master
$git status
```

9. View your changes on your GitHub repository
   - https://github.com/<your-name>/HackIowaDemos

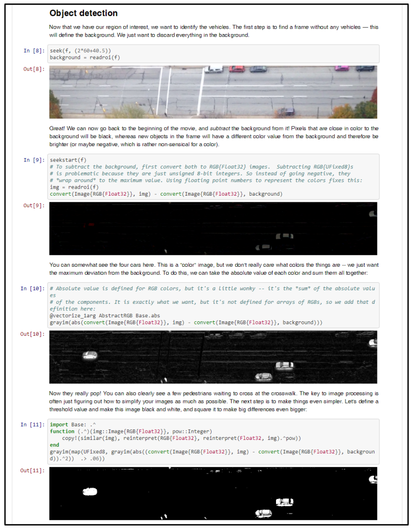
10. Create a pull request into my repository
    - https://github.com/johnwmillr/HackIowaDemos/pulls



Python & Jupyter

- Jupyter notebooks are **awesome**.
- Interactive environment for both writing and presenting code
- Great for rapid development and prototyping
- Supports other languages too
  - Julia, R, etc.

Cool example



Object detection

Now that we have our region of interest, we want to identify the vehicles. The first step is to find a frame without any vehicles — this will define the background. We just want to discard everything in the background.

In [8]:
```
seek(f, (2*60+40.5))
background = readroi(f)
```

Out[8]:

Great! We can now go back to the beginning of the movie, and *subtract* the background from it! Pixels that are close in color to the background will be black, whereas new objects in the frame will have a different color value from the background and therefore be brighter (or maybe negative, which is rather non-sensical for a color).

In [9]:
```
seekstart(f)
# To subtract the background, first convert both to RGB{Float32} images.  Subtracting RGB{UFixed8}s
# is problematic because they are just unsigned 8-bit integers. So instead of going negative, they
# *wrap around* to the maximum value. Using floating point numbers to represent the colors fixes this:
img = readroi(f)
convert(Image{RGB{Float32}}, img) - convert(Image{RGB{Float32}}, background)
```

Out[9]:

You can somewhat see the four cars here. This is a "color" image, but we don't really care what colors the things are -- we just want the maximum deviation from the background. To do this, we can take the absolute value of each color and sum them all together:

In [10]:
```
# Absolute value is defined for RGB colors, but it's a little wonky -- it's the *sum* of the absolute valu
es
# of the components. It is exactly what we want, but it's not defined for arrays of RGBs, so we add that d
efinition here:
@vectorize_1arg AbstractRGB Base.abs
grayim(abs(convert(Image{RGB{Float32}}, img) - convert(Image{RGB{Float32}}, background)))
```

Out[10]:

Now they really pop! You can also clearly see a few pedestrians waiting to cross at the crosswalk. The key to image processing is often just figuring out how to simplify your images as much as possible. The next step is to make things even simpler. Let's define a threshold value and make this image black and white, and square it to make big differences even bigger:

In [11]:
```
import Base: .^
function (.^)(img::Image{RGB{Float32}}, pow::Integer)
    copy!(similar(img), reinterpret(RGB{Float32}, reinterpret(Float32, img).^pow))
end
grayim(map(UFixed8, grayim(abs((convert(Image{RGB{Float32}}, img) - convert(Image{RGB{Float32}}, backgroun
d)).^2))  .> .06))
```

Out[11]:

*This is actually Julia, not Python, but same idea

APIs

- Application Programming Interface
- Protocol for interfacing with a website (or a database, application, etc.)
- All the cool kids have ‘em
  - Facebook, Twitter, Spotify, etc.
- Usually requires signing up for free credentials

Twitter Demo

- API credentials
  - https://python-twitter.readthedocs.io/en/latest/getting_started.html
- Tweepy
  - https://github.com/tweepy/tweepy
- Open my `tweepy-demo.ipynb` file:
  - `$jupyter notebook`

Twitter Demo

- Post a Tweet to my timeline

Genius.com Demo

- API credentials
  - https://docs.genius.com/
- Genius API
  - https://github.com/johnwmillr/geniusapi
- Open my `genius-demo.ipynb` file:
  - `$jupyter notebook`

That’s it!

- Questions?