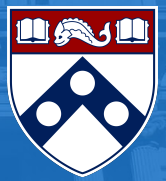
Finding the Right Data:
Scraping the Internet (For Free)

City of
Philadelphia



First: My Background

Previously…

- Background is in astrophysics, **not** finance, government, etc

And now…

- Teach MUSA 550: Python-based geospatial data science and data visualization
- Director of the Finance, Policy, and Data team in the Controller's Office

What is the Controller's Office?

- Financial watchdog for the City of Philadelphia
- **Functions:**
  - Efficient and effective use of taxpayer money
  - Focus on cost savings
  - Investigate fraud and mismanagement
- **Goals:**
  - Good government
  - Accountability
  - Transparency
  - Accessibility

City Controller, Rebecca Rynhart

Goals for today

Focus on the **lessons learned** and **open-source software** behind our data projects

By the end of today: you'll have some **guiding principles** and the **tools to get started.**

Lots of **examples and use cases** to learn from.

Finding the right data

Question: When data is the “new oil”, how do you find the right dataset?

Answer: Scraping

Scraping data gives you superpowers:

1. *Create* new, never before seen datasets
2. *Combine* related datasets in new ways
3. *Track changes over time* to discover new insights

Example: Changes over time

Finding the right data

Additional themes to consider:

- Make messy data more accessible
- Tracking changes over time can provide a measure of accountability (aka “having the receipts”)
- Automated scraping enables large enough datasets to answer more complex questions

Some good examples: 1) **Inside Airbnb** and 2) **Craigslist scraping**

Before we start: Leveraging open-source software

Data wrangling & scraping

Frontend visualization

Productivity

GitHub Actions

Visual Studio Code

Chart.js

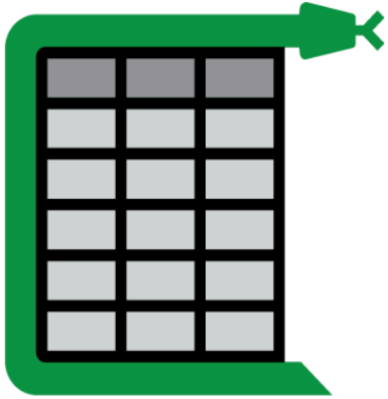
Option #1
Old-Fashioned Scraping: PDFs

Old-fashioned scraping: Extracting data from PDFs....
A messy, but invaluable skill

Tabula

Tabula is a tool for liberating data tables locked inside PDF files.

Camelot: PDF Table Extraction for Humans



Example: Make important financial data more understandable and accessible

City of Philadelphia

Quarterly City Managers Report

FOR THE PERIOD ENDING JUNE 30, 2019

Extracting data from PDFs.... A messy, but invaluable skill

Projection as of December 31, 2020

Amounts in Millions

	July 31	Aug 31	Sept 30	Oct 31	Nov 30	Dec 31	Jan 31	Feb 28	March 31	April 30	May 31	June 30	Total	Accrued	Not Accrued	Es Re
REVENUES																
Real Estate Tax	9.1	11.0	8.5	7.9	5.3	25.1	41.8	30.6	365.0	146.3	15.1	8.9	674.6			
Total Wage, Earnings, Net Profits	134.1	135.4	120.8	110.6	134.8	112.7	138.5	135.8	132.0	146.7	128.3	109.7	1539.5	(12.2)		
Realty Transfer Tax	36.4	22.8	24.6	28.8	26.2	32.4	27.2	15.0	19.0	20.9	20.2	22.2	295.7	(7.5)		
Sales Tax	24.4	29.7	12.5	13.4	15.2	12.5	14.2	15.2	11.0	12.2	15.8	25.9	201.9	2.4		
Business Income & Receipts Tax	266.4	26.8	19.6	34.6	2.8	21.8	15.7	5.1	50.0	250.3	49.7	8.7	751.4	(269.2)		
Beverage Tax	5.7	6.2	5.6	6.9	5.3	4.9	5.1	4.3	4.4	4.7	4.8	5.0	63.0			
Other Taxes	2.7	2.6	4.2	4.3	4.9	4.1	3.9	3.8	4.1	4.4	4.2	4.5	47.7	2.1		
Locally Generated Non-tax	22.8	20.8	28.5	23.5	34.7	29.0	23.2	34.1	32.8	32.4	35.9	30.0	347.6			
Total Other Governments	14.4	45.2	82.7	13.2	19.0	12.6	49.4	15.8	5.0	9.2	9.8	13.6	289.9	88.3		
Total PICA Other Governments	53.0	37.8	32.6	23.2	62.1	30.5	28.3	26.5	45.9	41.7	41.4	38.7	461.6			
Interfund Transfers	0.0	0.0	0.0	0.0	34.3	0.0	0.0	0.0	0.0	0.0	0.0	25.5	59.8		65.8	
Total Current Revenue	568.9	338.2	339.4	266.4	344.4	285.4	347.4	286.2	669.4	668.9	325.2	292.7	4732.6	(196.1)	65.8	
Collection of prior year(s) revenue	(71.3)	17.7	11.9	1.8	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	(39.9)			
Other fund balance adjustments		.														
TOTAL CASH RECEIPTS	497.6	356.0	351.3	268.2	344.5	285.4	347.4	286.2	669.4	668.9	325.2	292.7	4692.7			

	July 31	Aug 31	Sept 30	Oct 31	Nov 30	Dec 31	Jan 31	Feb 28	March 31	April 30	May 31	June 30	Total	Vouchers Payable	Encum-brances	Es Ob
EXPENSES AND OBLIGATIONS																
Payroll	50.4	202.6	136.7	150.8	158.5	183.1	134.6	142.1	142.8	142.8	156.4	165.7	1766.4	65.1	4.1	
Employee Benefits	45.9	41.5	46.3	48.8	60.9	45.9	43.8	51.7	51.9	51.9	56.6	53.5	598.7	37.8	0.5	
Pension	3.6	(0.5)	16.8	81.0	(0.3)	3.2	(0.3)	(0.3)	542.6	(5.2)	(0.3)	(0.3)	640.0	13.9		
Purchase of Services	33.3	49.4	68.6	77.6	63.1	74.7	48.7	42.2	191.1	67.8	65.3	48.2	830.0	25.5	144.5	
Materials, Equipment	5.1	2.1	3.6	6.1	5.7	2.7	6.0	7.4	9.2	7.7	7.8	10.0	73.5	3.0	59.0	
Contributions, Indemnities	19.4	1.6	11.9	65.4	13.9	66.0	7.8	16.5	71.2	10.0	7.6	103.0	394.3			
Debt Service-Short Term	0.2	0.0	0.1	0.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0	3.9	5.2			
Debt Service-Long Term	118.6	0.3	0.0	0.0	0.0	5.2	43.0	0.0	0.4	0.2	0.2	12.6	180.5			
Interfund Charges	0.0	0.4	0.0	0.0	20.2	4.4	0.0	3.7	0.0	4.8	0.0	11.7	45.2	22.0		
Advances & Misc. Pmts. / Labor Obligations	0.0	0.0	0.0	0.0	0.0	0.0	0.0	5.0	5.0	5.0	5.0	5.0	25.0			
Current Year Appropriation	276.3	297.2	283.9	429.7	322.0	385.2	284.6	268.3	1014.5	285.1	298.7	413.4	4558.8	167.3	208.0	

Example: Quarterly Cash Reports

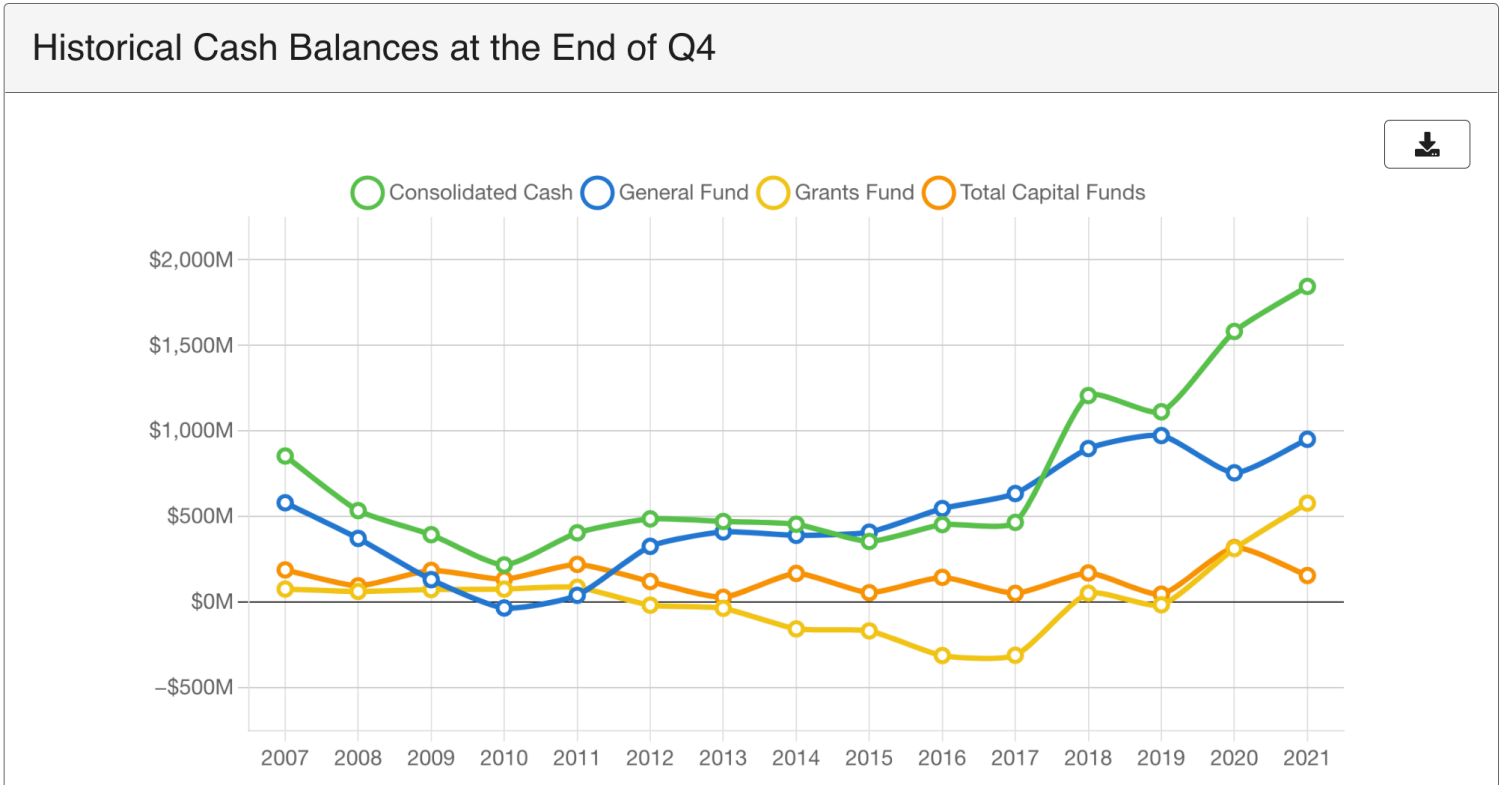
What is it? — Analysis of the city's monthly cash balances and revenue/spending totals

Goals

- **Add historical data** — use Python software to digitize records back to 2007
- **Add context** — narrative piece to explain the concepts
- **Add interactivity** — Visualization library for simple interactive charts

The latest report: Fiscal Year 2021

Apexcharts + Vue.js for easy, interactive charts to explore a complex dataset



Option #2

Unlocking new data

Web Scraping + Git = “Git Scraping”

Version control

Coined by Simon Willison in this blog post

The Power of “Git Scraping”

Web Scraping

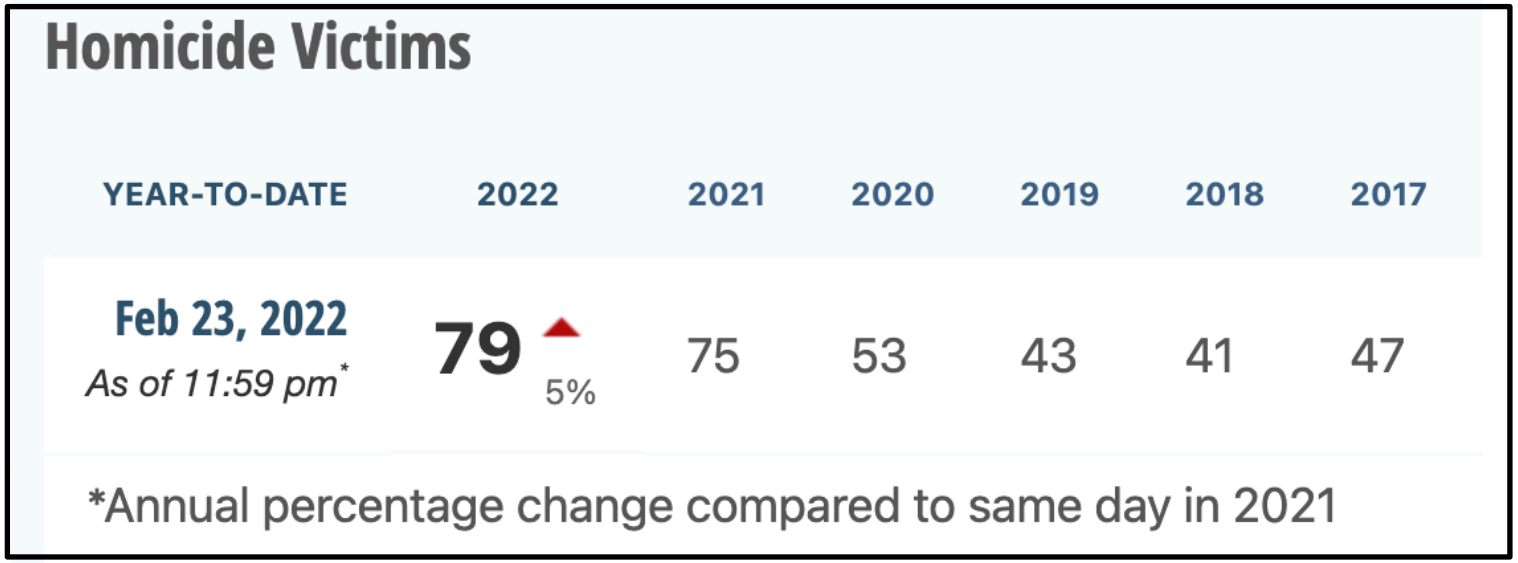
- The Internet is full of interesting data — only a small fraction is available via an API or a similar “clean” format
- Scraping gives you the power to extract this “messy” data

Git

- The Internet is a living document — sometimes the changes over time are more interesting than a static snapshot
- Git gives you the power to track versions of files over time

Example: @PHLHomicides

Current YTD homicide total updates daily on the Police Department's website



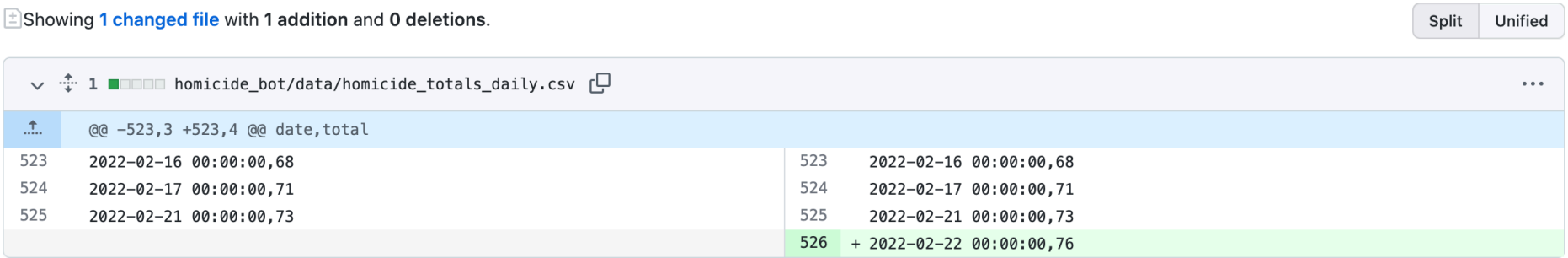
Homicide Victims

YEAR-TO-DATE	2022	2021	2020	2019	2018	2017
Feb 23, 2022 *As of 11:59 pm**	79 ▲ 5%	75	53	43	41	47

*Annual percentage change compared to same day in 2021

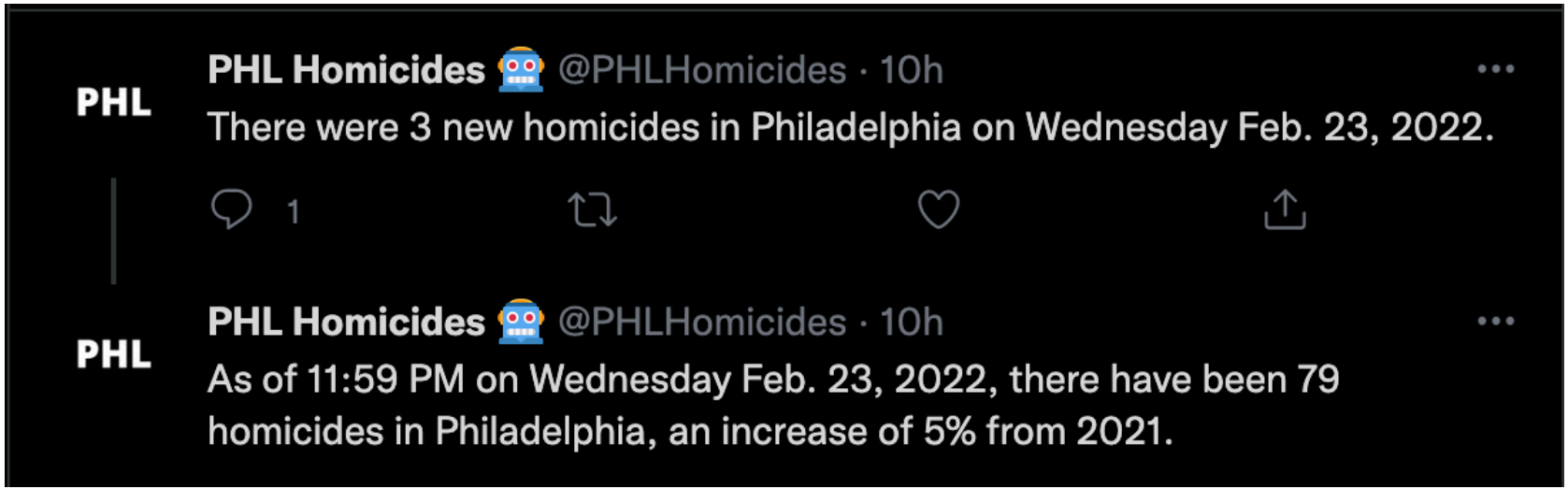
Example: @PHLHomicides

Data is scraped daily, saved to a CSV file, and added to a Github repository



Example: @PHLHomicides

Data is then tweeted daily, providing an easily accessible record of homicides over time



The Power of “Git Scraping”

One more key piece: automation

- Github (and “Github Actions”) gives you the power to combine web scraping and git.
- Github Actions can run the same task on a pre-defined schedule — generous usage limits for open-source repositories
- Allows you to automate a git-scraping pipeline, for free!

Lots of examples to learn from!

In the Controller's Office, we track:

- Daily shooting victims
- New asbestos repair projects in Philadelphia schools
- New criminal filings
- Trash and recycling pickup times

Other examples:

- COVID-19 data tracker from the LA Times
- History of fires in CA
- For more: see the "git-scraping" topic on Github

3 Case Studies

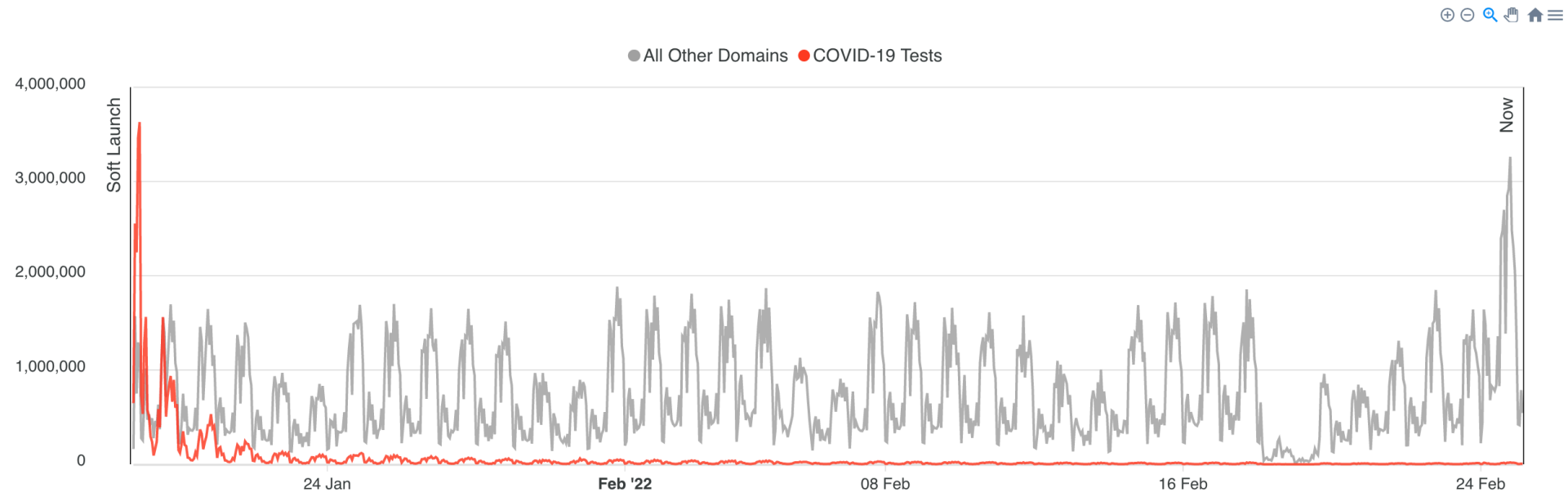
1. Tracking traffic to covidtests.gov
2. Mapping shooting victims
3. Mapping asbestos repairs in Philadelphia schools

Example #1: Tracking traffic to covidtests.gov

- The federal government's covidtests.gov launched in mid-January
- The realtime traffic to .gov websites is published via an API on analytics.usa.gov
- Data is provided via an API: no complex scraping is needed
- Perfect use case for git scraping — track traffic over time

Hourly Traffic on Government Websites for COVID-19 Tests

Last updated at 10:28 PM EST on Thursday, February 24, 2022



Github's Flat Data

- Github's version of "git scraping"
- Uses a "Flat Action" to download and save a data file on a periodic schedule
- Instructions are [available here]

Run every 20 minutes

```yaml
name: data
on:
  schedule:
    - cron: "*/20 * * * *"
  workflow_dispatch: {}
  push:
    paths:
      - .github/workflows/flat.yml
jobs:
  scheduled:
    runs-on: ubuntu-latest
    steps:
    - name: Setup deno
      uses: denoland/setup-deno@mair
      with:
        deno-version: v1.x
    - name: Check out repo
      uses: actions/checkout@v2
    - name: Fetch data
      uses: githubocto/flat@v3
      with:
        http_url: https://analytics.usa.gov/data/live/all-pages-realtime.json
        downloaded_filename: data.json
```

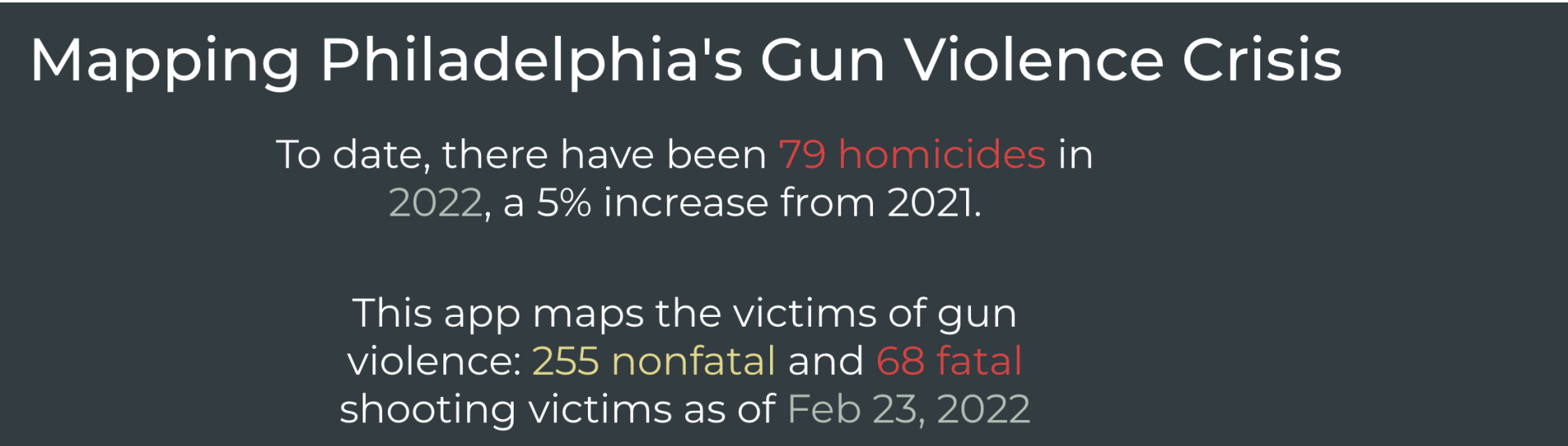
Set up and check out code

Flat Action

Flat Action =
No custom scraping code required!

Example #2: Automatically tracking shooting victims

- Download data from city's open data portal data, clean, and add info scraped from the PA court system portal
- Feed data into an interactive dashboard mapping the locations of shooting victims across the city



Example #2: Automatically tracking shooting victims

Open Data Philly

Data and Resources

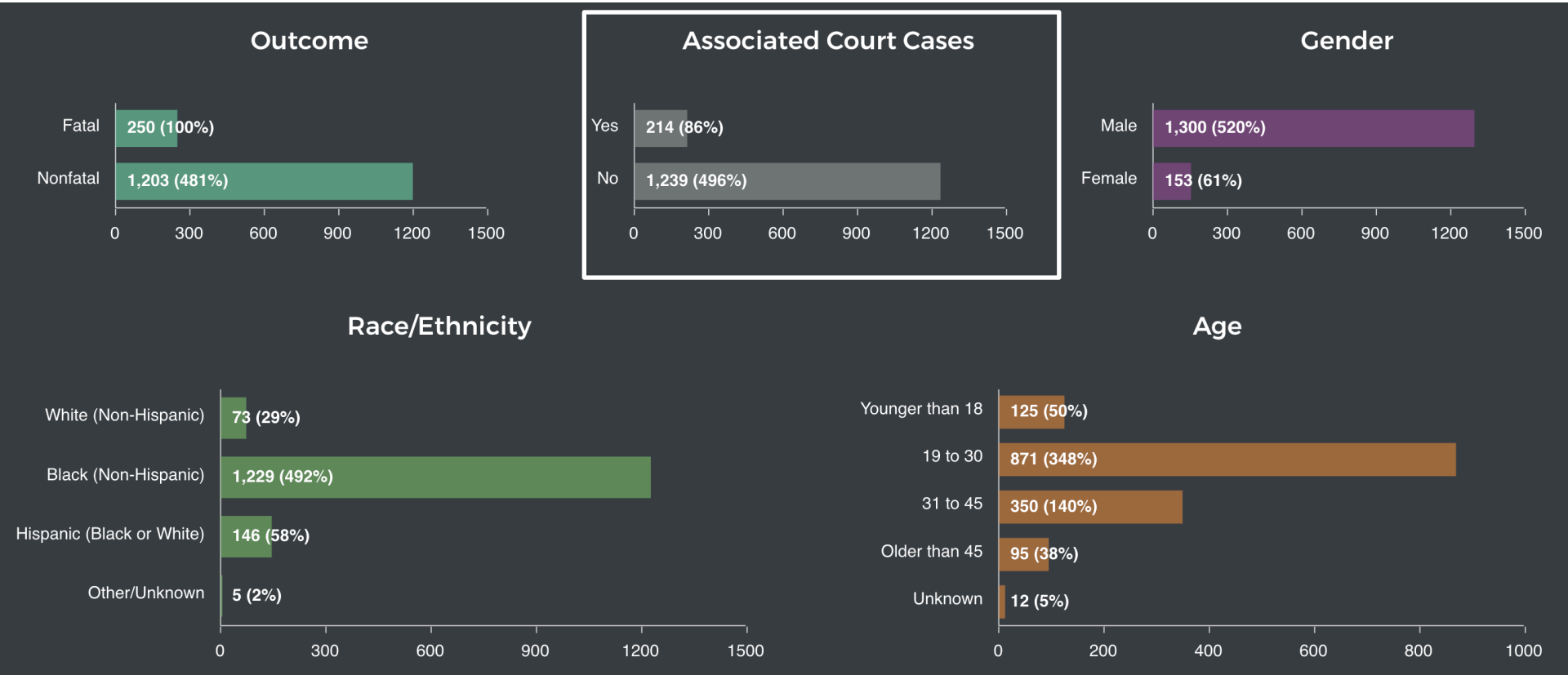
- **Shooting Victims (Map Journal)**
  Guided tour of the data, contextualized with other datasets
- **Shooting Victims (Visualization)**
- **Shooting Victims (CSV)**
- **Shooting Victims (GeoJSON)**
- **Shooting Victims (SHP)**
- **Shooting Victims (API Documentation)**
- **Shooting Victims (Metadata)**

The UJS Portal

Takeaway: Make data more accessible and combine in new ways

Example #2: Automatically tracking shooting victims

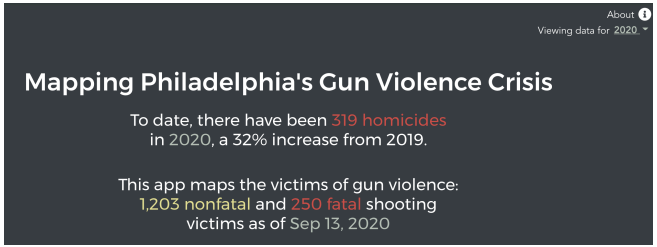
Data scraped from court portal



The power of Github Actions

Update data automatically each day with Github Actions

Dashboard pulls new data from Github



Data sources

Dashboard

The Update Workflow

1. **Download** latest CSV data for shooting victims from Open Data Philly
2. For every incident number, **scrape** associated court case information from the PA Unified Judicial System's online portal
3. **Save** a fresh copy of the cleaned data to upload into the

```yaml
on:
  schedule:
    - cron: "30 11 * * 1-5"
    - cron: "30 15 * * 1-5"
    - cron: "30 17 * * 1-5"

jobs:
  daily-sync-shootings:
    name: Daily Shootings Data Sync
    runs-on: ubuntu-latest
    steps:
      - uses: actions/checkout@v2
        with:
          persist-credentials: false
          fetch-depth: 0
      - uses: actions/setup-python@v2
        with:
          python-version: "3.8"
      - name: Run image
        uses: abatilo/actions-poetry@v2.0.0
        with:
          poetry-version: "1.1.11"
      - name: Install dependencies
        run: sudo apt-get install -y libspatialindex-dev
      - name: Download files
        run: |
          poetry install
          git pull origin master
          poetry run gv-dashboard-data daily-update --debug --shootings-only
      - name: Commit files
        continue-on-error: true
        run: |
          git config --local user.email "action@github.com"
          git config --local user.name "GitHub Action"
          git add -f gun_violence_dashboard_data/data/**/*.json
          git commit -a -m "Add daily download changes"
      - name: Push changes
        uses: ad-m/github-push-action@master
        with:
          github_token: ${{ secrets.GITHUB_TOKEN }}
```

Schedule

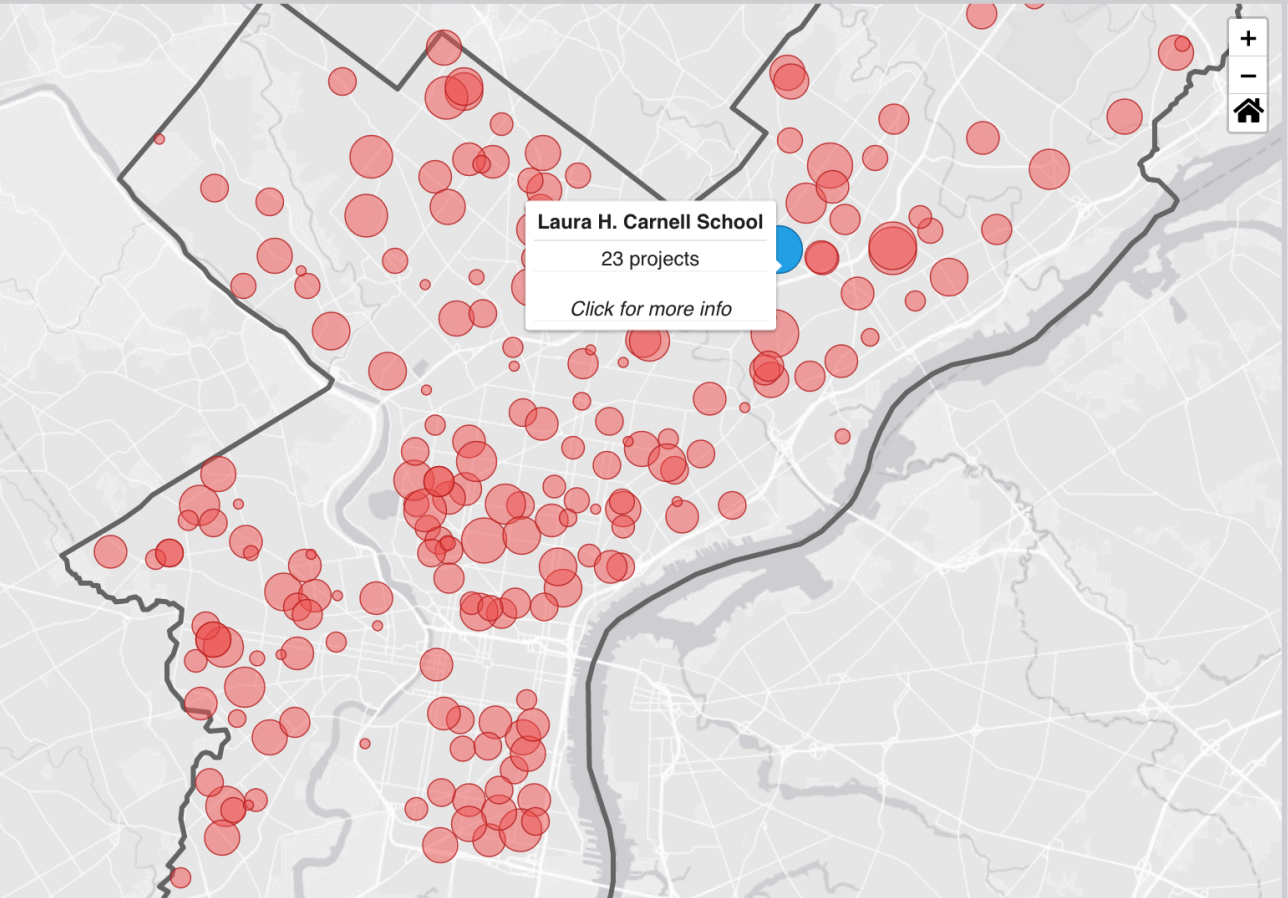
Set up Python

Run the update

Commit the new files

Side note: Selenium is amazing

Example #3: Mapping asbestos in schools



*Showing locations for **1,893** asbestos projects*

Click on a circle to view the detailed data for a particular school. Use the filters below to select data for a specific school, level, or project type.

Learn more about the data.

DOWNLOAD DATA

RESET ALL FILTERS

Filters

School Name

View projects for a specific school

Application Date

Planned Completion Date

Project Type

Operation Type

School Level

Example #3: Mapping asbestos in schools

Origin story:

- Data was publicly available on a City website but site was difficult to understand and data hard to extract
- Use git scraping (selenium) to automate download process to get a fresh copy daily
- Match projects to Philadelphia school buildings to improve the dataset's utility
- Present using an interactive mapping dashboard for easy exploration

Takeaway: Make data more accessible and combine in new ways

Wrapping up: Some things to remember

- **Git scraping can help you:**
  1. *Create* new datasets
  2. *Combine* related datasets in new ways
  3. *Track changes over time* to discover new insights
- **Github Actions & Flat Data are amazing tools**
- **Things to strive for:**
  - Make valuable data more accessible
  - Track changes over time for accountability and to build up datasets to discover more meaningful trends

Thank you!

Controller's Office


@PhilaController


@PhilaController

Personal


@nickhand


@nicholashand

All of the software behind the examples is available on Github