

CS1210 Lecture 41

Dec. 3, 2021

- HW 10 due tomorrow
- HW 11 has been published, due 8pm, Sunday, Dec. 12
- DS 11 will be available Monday, due 5pm Thursday, Dec. 9. It is helpful practice for the “markers” part of HW11
- Once again: Course grading scales on next slide

Today:

- HW11 and Twitter

Next week

- More on HW11
- Some interesting/fun CS theory/advanced topics
 - The Halting Problem, P vs NP, Turing machines, ...

Final Exam is optional. The default is that you are NOT taking it. You must OPT IN and notify me if you want to take it

Grade scales the are same percentage-wise except for rounding differences

Without final

Possible points: 168

Grade	Points	approx %
A+	163	97
A	148	88.1
A-	141	83.9
B+	133	79.2
B	122	72.6
B-	117	69.6
C+	109	64.9
C	92	54.8
C-	83	49.4
D+	80	47.6
D	73	43.5
D-	67	39.9

So far, maximum possible points: 156

HW: $8 \times 6 = 48$

DS: $10 \times 3 = 30$

Quizzes: $18 + 3 * 10 = 48$

Maximum without final is: 168

HW9: 6 HW10: 5 HW11: up to 7 additional

DS11: up to 3 additional

I will provide a with-final scale in a few days. The percentages to achieve grade levels will be very nearly the same.

NOTE: POINTS ARE THE OFFICIAL SCALE – NOT PERCENTAGE

Any final questions on HW 10?

- HW10 consists of key first steps you add onto in HW11. So ... you really need to get it done correctly!
 - add Entry widget so you can change location
 - enable zooming
 - enable changing map type
 - display pin at map center
- Initial code presents GUI with a map created via use of Google Static Maps API <https://developers.google.com/maps/documentation/maps-static/intro>
- Also uses Google Geocoding API <https://developers.google.com/maps/documentation/geocoding/intro>

Last time

Details of HW10

- Structure of HW 10
- using `urllib.urlopen()`, `read()`, `decode()` functions to retrieve info – page source or JSON data - from website/URL
- Basics of JSON. `json.loads()` to convert JSON-formatted data retrieved from web into Python dictionaries, so we can work with them easily and extract relevant info after studying API documentation (e.g. Google Geocoding and Static Maps APIs)
- `geocodeAddress`

Today

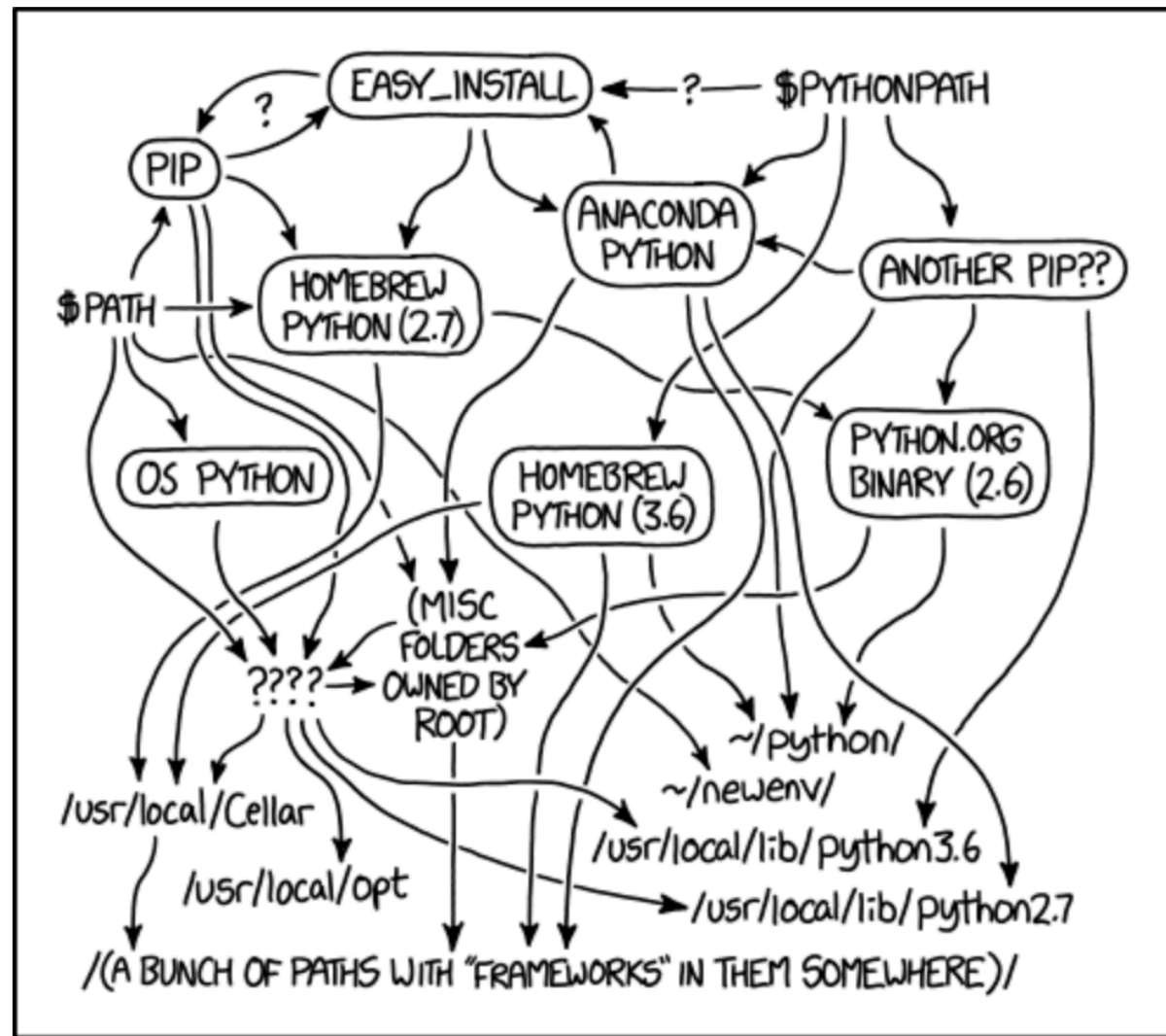
Details and important information about HW11

- Working with Twitter
 - How to access/authenticate with web services that require authorization
 - How to work with JSON to extract just the info you need from

HW11 Full Assignment Todo list

1. Complete HW10. Do not do anything beyond this step until it is complete and working! Email me/TA when you can't figure things out!
2. Do the "enabling steps" – get Twitter account, keys, "install" OAuth related modules. Make sure `twitteraccess.py` functions work for you. (HW11 page provides a super *easy* way to install the necessary OAuth-related modules.)
3. add new Entry for search term (**don't** add a new button for this! Original button should now read both entries, execute Twitter search, and retrieve and show map)
4. good practice: rename `readEntryAndDisplayMap` to, for example, `readEntriesSearchTwitterAndDisplayMap`
5. Before trying to actually display tweets on GUI and markers for tweets on map, test basic integration of twitter search. Upon button press, the `readEntriesSearchTwitterAndDisplayMap` callback function (command) should now read **both** Entry widgets, and *initially* just:
 - print tweet information the python shell
 - Maybe initially print whole tweet dictionary, but next extract just needed parts – text, name, screen_name and print those
 - Show the basic map, disregarding tweet information for now.
6. Once basic twitter integration has been checked, work on displaying tweet information in GUI and enabling "stepping through" tweets:
 - Save the retrieved tweets in a property of Globals. E.g. `Globals.tweets`
 - set a value to indicate which is the "current tweet". E.g. use another property such as `Globals.currentTweetIndex`. Set it to 0 when you first retrieve the tweets.
 - add widget(s) to GUI where tweet information can be displayed. Could be Label(s) but Text widget might be better (more on this later).
 - Implement a function `displayTweet` that updates relevant GUI widget(s) with information from 'current tweet'
 - Thus, `readEntriesSearchTwitterAndDisplayMap` should now read two Entries, call search Twitter, set some Globals properties, and call `displayTweet` *and* `displayMap`
7. Add GUI widget(s) to change "current tweet". If prior things are done well, this should be easy. Much like zoom buttons. Might have 'Next tweet', 'Prev tweet' buttons that simply change `Globals.currentTweetIndex`, and then call ???
8. Make sure to constrain tweet search by location. Pass the lat/lng of the specified location in your call to `searchTwitter`. Experiment with "radius" argument. Default radius is 2km. Large and very small radii don't seem to work well.
9. Add additional GUI widget(s) to display more tweet information, including the URLs within a tweet. This adds another layer of complication. Tweets can contain multiple URLs and you should be able to step through these as well. So, in addition to `Globals.currentTweetIndex`, you'll probably want another property such as `Globals.currentTweetURLIndex`, along with widgets to change this
10. Add code to allow opening of current tweet URL:
 - E.g. a button and callback/command function that simply executes `webbrowser.open(relevant URL)`. Note: your code will need to import `webbrowser` module
11. Add markers (pins) for tweets - this is what DS11 helps you do.

Why I often avoid “proper” installation of Python modules like oauth2 ☺



MY PYTHON ENVIRONMENT HAS BECOME SO DEGRADED THAT MY LAPTOP HAS BEEN DECLARED A SUPERFUND SITE.

Python Environment^(alt-text)



Needed for HW11: DO VERY SOON!

Definitely do step 1 before Monday

1. GET A TWITTER DEVELOPER ACCOUNT! Free

- First, get a Twitter account if you don't already have one. (if you are worried about us seeing your personal tweets, make a different Twitter account just for this class - but I promise we won't look!)
- Follow the detailed steps at <https://homepage.divms.uiowa.edu/~cremer/courses/cs1210/etc/getElevatedTwitterDevAccess.pdf> to setup up a Twitter Developer account with “Elevated” access. It looks like a lot of little steps but it just takes a few minutes.
- Then get the necessary API keys by following the steps at <https://homepage.divms.uiowa.edu/~cremer/courses/cs1210/etc/accessKeysInstructions.pdf> This also takes just a few minutes.

2. Add the keys you obtained at the end of step 1 to `twitteraccess.py`. Test that `searchTwitter()` in `twitteraccess.py` works for you.

3. Then start working on HW11...

- More important than ever: do not write many lines of code before testing! *This assignment has a lot of code and many little things can go wrong.* If you add a lot of lines and then it crashes/doesn't work, it can be very difficult to debug/find where the error is.
- Add a few lines, test, add a few lines, test, ...

Working with Twitter – twitteraccess.py

1. Twitter uses a very common authorization scheme called OAuth.
 - After you have a developer account, you need to create/register a Twitter app. Do that here: <https://developer.twitter.com/en/apps>
 - Once the app has been created, copy the four keys/tokens for the app and put them in the appropriate places in twitteraccess.py
 - API key, API secret, Access token, Access token secret
2. Before trying any other functions in twitteraccess.py, execute:
 - authTwitter() to enable access to the Twitter API by sending your authorization information
3. Try various functions in twitteraccess.py
 - Most importantly, learn to use the searchTwitter(...). It is key to retrieving Tweets for HW11
 - searchTwitter(...) constructs a string (like geocodeAddress and getMapURL but now for the Twitter Search API rather than Google APIs) that gets sent to Twitter.
 - Study the Search API docs here: <https://developer.twitter.com/en/docs/tweets/search/api-reference/get-search-tweets> to understand the structure of the string sent to the Twitter Search API
 - The Twitter Search API returns a JSON object that contains a list of JSON Tweet objects as the value of its “statuses” key. The structure of a JSON Tweet object is detailed here: <https://developer.twitter.com/en/docs/tweets/data-dictionary/overview/tweet-object>

HW11: things to consider/keep in mind as you get to various parts

- *recommendation*: use Text widget rather than Label for displaying tweets.
 - in GUI initialization

```
tweetText = tkinter.Text(width=..., height = ...)
tweetText.configure(state=tkinter.DISABLED)
```
 - in code to display tweet:

```
tweetText.configure(state=tkinter.NORMAL)
tweetText.delete(1.0, tkinter.END)
tweetText.insert(...)
tweetText.configure(state=tkinter.DISABLED)
```
- Few tweets have non-null 'coordinates' field, so many of your markers will likely be in the middle of the maps. That's okay.
 - *recommendation*: to get more tweets *with* non-null coordinates field, it seems to help to use small search radius – e.g. 2km rather than 5 or 10.
 - Search in popular location of, say, large city. E.g. "Times Square" and "party"
- Handling Unicode characters in tweets:
 - Printing "raw" text of some tweets in Python shell will cause errors
 - see "printable" function in twitteraccess.py for "hacky" approach to eliminating unprintable characters

Next time

- More HW 11 and twitter
- Intro to some advanced CS topics