CS1 Lecture 40 Apr. 26, 2017

- Exam scores have been posted:

0-4	5-9	10-14	15-19	20-24	25-29	30-34	35-37
3	5	25	44	52	30	26	12

Median: 22, High: 37

- Scores and grades so far (includes all but HW 7-10, Disc. Sec. 10, and Final Exam). 153 points out of 250
  - "Free" homework and discussion section scores have been added (via 10 points HWfree plus dropping lowest HW score)
  - Grades: 85% A, 70% B, 50% C, 40% D.
- Discussion sections this week – no points. Use discussion section to work on HW 9 Q1
- HW 9 Q2 available. Start working on it!

Review - Global variables

- Previously discussed variable scoping. In functions, local variables are created for 1) formal parameters, and 2) any names that appear on left hand side of assignment
- What if you *want* to assign to top-level variable? Can use "global". Within a function

  ```
  global x
  ```

  tells Python that a local variable should *not* be created for x. Instead, uses of x within the function refer to top-level variable x
- Examples: globalVars.py
- General advice: minimize use of global variables. Functions are easier to understand if they can be understand *locally*, without having to take into account a larger context (perhaps looking through and disentangling many files/functions). *However*, when carefully used they can be very useful.
- See "Global variables" section of Chapter 11 in textbook.

For HW 9 and HW 10 we will use several global variables to make it easy to share GUI- and map-related values across multiple functions.

HW9 Q1

- Make sure to declare appropriate variables ‘global’
- When you update the text in a Label (e.g. statusLabel) using configure.
  - If the new text is shorter than the old text, some of the old text will likely remain displayed on the window. That’s because the Label widget is now smaller and it “left alone” the pixels outside its new extent.
  - One solution: when you create the Label, give it a width property (specified in number of characters) larger than the longest text you’ll ever put in it

HW9 Q2

- HW 9 Q2. Important first steps of program you'll complete in HW 10
  - 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/static-maps/intro#quick_example
- Also uses Google Geocoding API

  https://developers.google.com/maps/documentation/geocoding/intro#Geocoding

Using the map service from Python

1. Create a (sometimes very long) URL string describing the map you want. Use the documentation sites (from prev. slide) to learn the format of the URL
   - see getMapUrl in hw10start.py
   - You will need to modify getMapURL for the pin and maptype parts of HW9 Q2
2. Send the URL to Google and receive the results
   - see retrieveMap

Using the geocoding service from Python

To convert a string for a location to a latitude, longitude pair:

1. Create a URL string describing the geocoding info you want.
   - first three lines of geocodeAddress
2. Send the URL to Google and receive the results
   - Fourth line of geocodeAddress
3. This time, instead of getting back a gif image, Google sends back a JSON-formatted string.
   - JSON is a very commonly used open standard for transmitting data object info as text. http://en.wikipedia.org/wiki/JSON Perhaps most commonly, JSON is used to transmit data in dictionary form – i.e. before transmitting, data is encoded into a long string that "looks" (to human reader) like a dictionary representation. The receiver of this string can then "decode" the string into a dictionary data structure and extract the items of interest out of it.
   - Fifth line decodes the JSON result, yielding a Python dictionary
   - Sixth-eighth lines look at various items in the dictionary to extract the info we need (here, latitude and longitude). *You need to look through the Google Geocoding API web page to see what information is in the dictionary (often, much more than you'll actually use!)*

HW9 Q2 To Do list

- add Entry widget so you can change location
  - Upon button press, callback should read Entry, set mapLocation (global) variable, and call showMap()
- enable zooming
  - Upon use of widget (button or whatever you choose), set zoomLevel (global) variable, call showMap()
- enable changing map type
  - add a new global variable for mapType
  - modify getMapURL to so that string representing request to Google Static maps API specifies desired map type
  - upon use of widget, set mapType variable, and call showMap()
- display pin at map center
  - modify getMapURL so that string representing request to Google Static maps API includes specification of pin at map center

Next time

- Details and important information about HW 10
  - Making calls to Web APIs (and constructing the necessary large strings)
  - How to work with JSON
  - How to access/authenticate with web services that require authorization (e.g. Twitter, Flickr)