CS 116x Winter 2015
Craig S. Kaplan

Assignment 09: Data Processing

Due date: Monday (Friday?), 06 April, 5:00pm

Question 1: Kiosk

Congratulations! Based on your performance in CS 116x, the University of Waterloo has asked you to design a new information kiosk to be installed in multiple locations on campus. The kiosk will access local sources of Waterloo-related information in real time and display them on a monitor. The display will rotate through a few different information sources, changing at fixed intervals of time, so that passerby can casually watch the monitor for a minute and catch something of interest (you might compare it to a billboard cycling through a few ads, or the annoying parade of ads and trivia that runs before the lights go down in a movie theatre). A lot of the software infrastructure for the kiosk is already in place. Your job is to decide what content to show, and to format it in a visually appealing way on the monitor, i.e., in a sketch window.

Download the kiosk starter code from the course web page. You will find that the sketch is divided into two tabs. The main tab, `Kiosk`, is the one you'll edit to create your content. It has the usual `setup()` and `draw()` functions, together with a few other helper functions and local variables. The other tab defines a new class called `KioskModeInfo`, which represents one channel of information from the outside world (and will map onto one of the screens in the kiosk's rotation). You'll use that class to write your sketch, but don't change anything in it.

You must modify the starter code to create some interesting visualizations of live data. Here are the minimum requirements for a complete implementation:

- You must support at least two different "modes", i.e., screenfuls of information. You can create more if you want, but it's not required.
- Your modes can't draw all their information from the same information sources (this will be explained in more detail shortly).
- Each mode can display information using a mixture of text and graphics (including unchanging clip-art images or vector illustrations if you want), but you can't use text exclusively: there must be at least one mode that has a non-zero amount of graphics on it.
- At least one piece of graphics must *vary based on the data received from the internet*. A simple example would be a live virtual thermometer in which the mercury rises and falls with the real outside temperature.

The means of talking to the outside world is encapsulated in the `KioskModeInfo` class. An instance of this class stores the code for a web API request, and retrieves and stores a `JSONObject` that you can ask for later. There are four kinds of `KioskInfoMode` instances you can create:

- `new KioskInfoMode( "TEST", "" )`

  Build a test instance. This special case doesn't actually retrieve any information from the internet—it just makes up a `JSONObject` on request and gives it back to you. This feature is included because it makes it easier to test your ability to access parts of a `JSONObject`, and to demonstrate a working prototype in the starter code.

- `new KioskInfoMode( "UW", "a path to a UW service" )`

  Query the University of Waterloo's Open Data API. The University publishes a long list of live data services that you can query. Please see the complete list by visiting `api.uwaterloo.ca` and clicking on "Documentation". You should replace the second argument to the constructor with one of the paths in that list. For example, passing in `"/foodservices/menu"` will cause the query to send you back the current menu for food service locations on campus. Each possible query has its own more detailed documentation. A good starting point for this course would be `"weather/current"`.

  Note that calls to the UW API require an access key. I have obtained a group CS 116x access key that we will all share, which is embedded in the `KioskInfoMode` class. Please don't abuse this key by making too many requests—in the worst case, the whole CS 106 class may get locked out of the service!

- `new KioskInfoMode( "GRT", "a stop number", "a route number" )`

  Query the upcoming bus arrivals for a particular route at a particular stop. For example, 3699 is the iXpress stop outside Davis Centre, so you could pass in `"3699"` and `"200"`.

- `new KioskInfoMode( "RSS", "a feed URL" )`

  Obtain a JSON representation of an RSS feed (e.g., a blog or other web page that publishes information in instalments). For example, passing in

  `"http://www.cbc.ca/cmlink/rss-topstories"`

  as the second argument returns a feed of the current CBC headlines. Of course, there are countless other potential RSS feeds to draw information from (I can help you determine the URL for an RSS feed if it's not obvious for a given website).

The screens you implement must come from distinct queries (for example, don't just use the weather service twice). One should absolutely come from anywhere within the UW API. The second (and any others your choose to create) can come from a different part of the UW API, or from the GRT or RSS sources.

Rather than adding pages of text here, I suggest you proceed to read the comments in the starter code, and begin experimenting with data you retrieve from the internet. I just have a few other important tips to include here:

- When using RSS mode, you'll often get back information as raw HTML source instead of plain text. There isn't much you can do about that—Processing doesn't include a function to render HTML. Just display it as text and ignore any HTML tags, that's good enough for the purposes of this assignment. (You could try to use regular expressions to filter out HTML tags, but it will require a fair amount of work.)
- At some point while debugging it could be helpful to print out the `JSONObject` instances containing the data you're interested in. It should look like the examples on the documentation pages for the UW API. Looking at the printed object should give you a sense of how to drill down to the pieces of information you want to render.
- A kiosk screen need not display every piece of information in a `JSONObject`. Feel free to pick out one or two pieces of information that are easy or fun to display. For example, if you were grabbing the JSON for top CBC headlines, you could pick just one headline and show it.

What to submit: On LEARN, you should submit the sketch `A10Kiosk`. Submit the entire sketch folder.