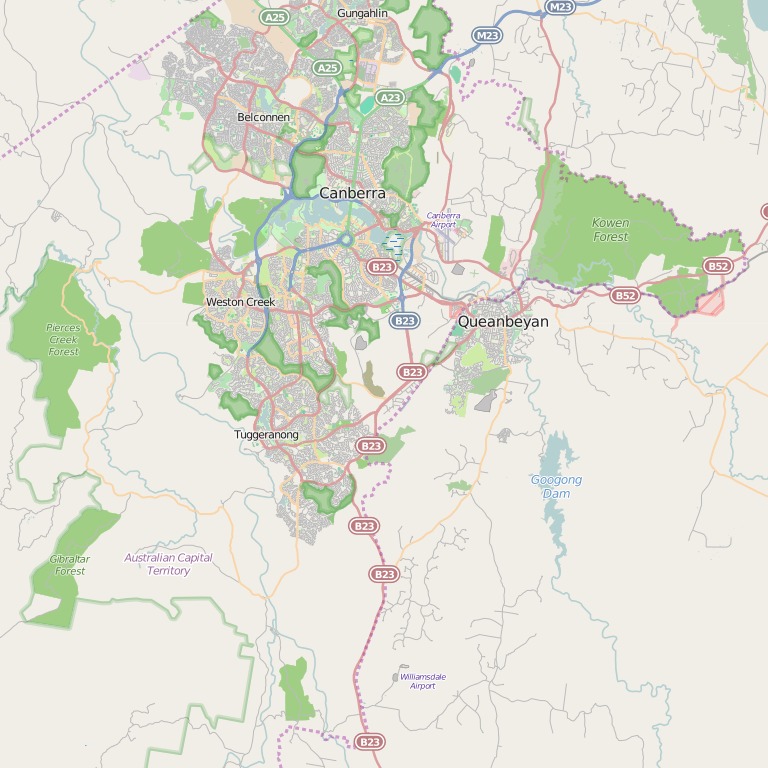
## **Overview**

In this assignment you will be developing a rudimentary system for visualising traffic incidents in the Canberra region. You will do this by parsing tweets from the *ACTPol\_Traffic* twitter account to extract the location (street and suburb) for incidents reported in each tweet. You will then plot these locations on a map of Canberra such as the one shown below. All data and starter code is provided.



## **Tasks**

You are provided with:

* a map of Canberra (and longitude/latitude bounds on the map)
* a file containing street and suburb names and their locations
* a file of tweets obtained from the *ACTPol\_Traffic* twitter account
* and starter Python code and test cases

You should complete the tasks listed below in the Python 3 programming language. You are free to use additional Python libraries to complete the task as long as they are bundled with the Anaconda 3 distribution.

**Part A. Text Processing**

1. parse and normalise tweets

Fix the implementation of the normalize\_message and split\_tweet functions in tweet\_utils.py. Close this issue once both unit tests test\_normalize\_message and test\_split\_tweets in test\_part\_a.py are passing.

1. match tweets against some street names

The unit test test\_match\_streets in test\_part\_a.py is failing. Correct the function match\_streets in tweet\_utils.py.

1. compute histogram of suburb mentions
   * **Note:** the histogram could be slightly inaccurate since suburb can be same as street names

The function suburb\_histogram is supposed to return a dictionary with the number of times each suburb is mentioned in the tweets but it is returning a dictionary containing all zeros. Complete the implementation of the function and close this issue. You can test your function with the test\_suburb\_histogram' unit test in 'test\_part\_a.py.

### **Part B. Visualization**

1. parse street locations from file

The function read\_street\_locations in traffic.py is not complete. You can test the function using the unit test test\_read\_street\_locations in test\_part\_b.py.

1. read and display the map

Modify the function read\_map in traffic.py to correctly read the map's latitude and longitude limits. Make sure your code passed the unit test test\_read\_map in test\_part\_b.py.

1. compute lat/long map coordinates (and test)

Write code in the function latlon\_to\_point to compute (x, y) coordinates for a given latitude and longitude. The coordinates should be computed by interpolating between the minimum and maximum lon/lat for the map where 0 corresponds to the minimum and 1 corresponds to the maximum.

Note this is only an approximation since the actual conversion would need to take into account projection of spherical coordinates onto the flat map.

To close this issue your code should pass the unit test test\_latlon\_to\_point in test\_part\_b.py.

1. animate tweet events on map

There is a small but crucial piece of code missing in the function animate\_tweets in traffic.py. Implement this code and run the traffic.pyscript. You should see the location of traffic events start to appear on the map.