Analyzing Social Graph Connections

An official Python SDK for the Graph API is a community fork of that repository previously maintained by Facebook and can be installed per the standard protocol with pip via pip install facebook-sdk. This package contains a few useful convenience methods that allow you to interact with Facebook in a number of ways, including the ability to make FQL queries and post statuses or photos. However, there are really just a few key methods from the GraphAPI class (defined in the facebook.py source file) that you need to know about in order to use the Graph API to fetch data as shown next, so you could just as easily opt to query over HTTP directly with requests if you prefer. The methods are:

get_object(self, id, **args)
Example usage: get_object("me", metadata=1)

get_objects(self, id, **args)
Example usage: get_objects(["me", "some_other_id"], metadata=1)

get_connections(self, id, connection_name, **args)
Example usage: get_connections("me", "friends", metadata=1)

request(self, path, args=None, post_args=None)
Example usage: request("search", {"q" : "social web", "type" : "page"})

The most common (and often, the only) keyword argument you'll probably use is metadata=1, in order to get back the connections associated with an object in addition to just the object details themselves. This example also introduces a helper function called pp that is used throughout the remainder of this chapter for pretty-printing results as nicely formatted JSON to save some typing.

```
{
 "paging": {
  "next": "https://graph.facebook.com/search?q=social+web&type=page...",
 },
 "data": [
  {
   "category": "Book",
   "name": "Mining the Social Web",
   "id": "146803958708175"
  },
  {
   "category": "Internet/software",
   "name": "Social & Web Marketing",
   "id": "172427156148334"
  },
  {
   "category": "Internet/software",
   "name": "Social Web Alliance",
   "id": "160477007390933"
  },
```