Twitter

Some companies will actually release language specific API wrappers for their API, while others will just publish an API and an independent group will wrap their API in their programming language of choice. In Twitter's case, they just published an API and several different individuals have wrapped it with Python. One of the most popular Python wrappers for the Twitter API is called **tweepy**. Let's install it and give tweepy a try!

```
pip install tweepy
```

Now that you have it installed, you will need to register with Twitter to get an authentication key and secret that you can use with their OAuth implementation. Twitter will also provide you with a unique access token and secret for accessing their API. Once you have those in hand we can write some code to extract the tweets from your account. Here's a simple example:

```
import tweepy

key = 'random_key'
secret = 'random_secret'
access_token = 'access_token'
access_secret = 'super_secret'

auth = tweepy.OAuthHandler(key, secret)
auth.set_access_token(access_token, access_secret)
api = tweepy.API(auth)

my_tweets = api.user_timeline()

for tweet in my_tweets:
    print(tweet.text)
```

The first few lines of code are all setup. We need to import tweepy and set up our various keys and secrets. Then we can create our authentication handler

set our access token. Once that's done, we create a Twitter API object and use

that grab our tweets via the **user_timeline** method. You can pass the user_timeline a userid or screen name parameter to get a specific user's tweets. When you don't pass anything, you will get the authenticated user's timeline tweets. Also note that this only returns the 20 latest tweets by default. We can tell it to grab more than that and we can also enable pagination.

In the last chapter, we were also grabbing the date of the tweet with BeautifulSoup. You can do that quite easily by accessing your **tweet** object's **create_at** attribute. If you'd like to know what other items you can access, just call Python's built-in **dir** method on a tweet object:

```
dir(tweet)
#['__class__', '__delattr__', '__dict__', '__dir__', '__doc__', '__eq__', '__format__', '__ge
```

You can actually update your status via the api object we created:

```
api.update_status('I just tweeted using Python')
api.update_with_media(filename, 'I can tweet files with Python')
```

Basically you can do just about anything with tweepy that you can do in a browser on Twitter. I highly recommend checking out their documentation as there is a lot more to their API than what I can cover in this chapter.