**External Data Sources - Crime, News Feeds**

Note: Local APIs and Mashups

<http://blog.programmableweb.com/2012/06/12/88-local-apis-yahoo-local-yelp-and-zvents/>

1. **Web based – Local News**

Pros:

* Available on public websites
* information rich, categorized
* Trusted
* Most of them can be tagged for places from the content

Cons:

* No API based access
* No location enabled data

1. ***News Gazzette – Local News - Sorted by time***

<http://www.news-gazette.com/news/local>

Defined Categories

* University of Illinois – Tags: #News, #Events
* Courts, Police and Fire – Tags: #Safety, #Crime

1. ***Daily Illini – (Student News Paper at University of Illinois)***

<http://www.dailyillini.com/news/>

Defined Categories

* Campus – Tags: #News, #Events
* Administration – Tags: #News
* Crime (Police Blotter by date) - Tags: #Safety, #Crime

Types: Burglary, Theft, Robbery, Homicide, Illegal Possession etc

1. **Web based– API based Local News**
2. ***YourStreet API***

The YourStreet API lets you provide latitude and longitude coordinates and returns geocoded local news articles for your area. You can specify your location and the time span you would like to receive articles from

1. ***Fwix*** is a local news site using technology designed to filter and select news stories that are locally relevant. The Wire API uses RESTful architecture to allow developers and media outlets to use Fwix's data platform to incorporate local news and information into their web properties. Currently, fetching information from the Wire is unrestricted, but publication requires pre-approval. Responses are formatted in XML and JSON.

<http://www.programmableweb.com/api/fwix-wire>

1. **Web based National – API based**
2. ***New York Times Community API***

<http://developer.nytimes.com/docs/community_api#h3-requests>

* + [Recent Comments](http://developer.nytimes.com/docs/community_api#commentsRecent)
  + [Random Comments](http://developer.nytimes.com/docs/community_api#commentsRandom)
  + [Comments by Date](http://developer.nytimes.com/docs/community_api#commentsDate)
  + [Comments by User ID](http://developer.nytimes.com/docs/community_api#commentsUser)
  + [Comments by URL](http://developer.nytimes.com/docs/community_api#commentsURL)

1. **Social Media - API based streaming data**

Pros:

* API Based
* Filtered by keywords, location
* real time streaming data

Cons:

* Untrusted
* Noisy
* Unclassified

1. ***Twitter Streaming APIs***

<https://dev.twitter.com/docs/streaming-apis/streams/public#locations>

A Walk through

<http://bcomposes.wordpress.com/2013/01/25/a-walk-through-for-the-twitter-streaming-api/>

* + Filter by keywords
  + Filter by location
  + Filter by Users

1. ***Facebook Open Streaming APIs***

<https://developers.facebook.com/docs/reference/api/realtime/>

<https://blog.facebook.com/blog.php?post=57822962130>

<http://mashable.com/2009/04/27/facebook-open-stream-api-the-next-huge-platform/>

* + Filter by Event Type
  + Filter by User
  + Filter by Keywords
  + Filter by Delta

**Appendix:**

1. Twitter Streaming API example – Filter by keywords

Eg.

*/\*\**

*\* Copyright 2013 Twitter, Inc.*

*\* Licensed under the Apache License, Version 2.0 (the "License");*

*\* you may not use this file except in compliance with the License.*

*\* You may obtain a copy of the License at*

*\* http://www.apache.org/licenses/LICENSE-2.0*

*\* Unless required by applicable law or agreed to in writing, software*

*\* distributed under the License is distributed on an "AS IS" BASIS,*

*\* WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.*

*\* See the License for the specific language governing permissions and*

*\* limitations under the License.*

*\*\*/*

**package** com**.**twitter**.**hbc**.**example**;**

**import** com.google.common.collect.Lists**;**

**import** com.twitter.hbc.ClientBuilder**;**

**import** com.twitter.hbc.core.Client**;**

**import** com.twitter.hbc.core.Constants**;**

**import** com.twitter.hbc.core.endpoint.StatusesFilterEndpoint**;**

**import** com.twitter.hbc.core.processor.StringDelimitedProcessor**;**

**import** com.twitter.hbc.httpclient.auth.Authentication**;**

**import** com.twitter.hbc.httpclient.auth.OAuth1**;**

**import** java.util.concurrent.BlockingQueue**;**

**import** java.util.concurrent.LinkedBlockingQueue**;**

**public** **class** **FilterStreamExample** **{**

**public** **void** **oauth(**String consumerKey**,** String consumerSecret**,** String token**,** String secret**)** **throws** InterruptedException **{**

    BlockingQueue**<**String**>** queue **=** **new** LinkedBlockingQueue**<**String**>(**10000**);**

    StatusesFilterEndpoint endpoint **=** **new** StatusesFilterEndpoint**();**

*// add some track terms*

    endpoint**.**trackTerms**(**Lists**.**newArrayList**(**"twitterapi"**,** "#yolo"**));**

    Authentication auth **=** **new** OAuth1**(**consumerKey**,** consumerSecret**,** token**,** secret**);**

*// Authentication auth = new BasicAuth(username, password);*

*// Create a new BasicClient. By default gzip is enabled.*

    Client client **=** **new** ClientBuilder**()**

**.**hosts**(**Constants**.**STREAM\_HOST**)**

**.**endpoint**(**endpoint**)**

**.**authentication**(**auth**)**

**.**processor**(new** StringDelimitedProcessor**(**queue**))**

**.**build**();**

*// Establish a connection*

    client**.**connect**();**

*// Do whatever needs to be done with messages*

**for** **(int** msgRead **=** 0**;** msgRead **<** 1000**;** msgRead**++)** **{**

      String msg **=** queue**.**take**();**

      System**.**out**.**println**(**msg**);**

**}**

    client**.**stop**();**

**}**

**}**