In [1]:
```
# -*- coding: utf-8 -*-
import xml.etree.ElementTree as ET
import pprint

def count_tags(filename):
        tags={}
        for action,elem in ET.iterparse(filename):
            if elem.tag in tags:
                tags[elem.tag]=tags[elem.tag]+1
            else:
                tags[elem.tag]=1
            elem.clear()
        return tags

pprint.pprint(count_tags('jerusalem_israel.osm'))
```

```
{'bounds': 1,
 'member': 3881,
 'nd': 716073,
 'node': 645370,
 'osm': 1,
 'relation': 535,
 'tag': 177384,
 'way': 65316}
```

In [2]:
```
import xml.etree.ElementTree as ET
import pprint
import re


#defining regular expressions for quality check
lower = re.compile(r'^([a-z]|_)*$')
lower_colon = re.compile(r'^([a-z]|_)*:([a-z]|_)*$')
problemchars = re.compile(r'[=\+/&<>;\'"\?%#$@\,\. \t\r\n]')

#going through tag and check against RegEx
def key_type(element, keys):
    if element.tag == "tag":
        value = element.attrib['k']
        if re.search(problemchars,value):
            keys['problemchars']=keys['problemchars']+1
        elif re.search(lower_colon,value):
            keys['lower_colon']=keys['lower_colon']+1
        elif re.search(lower,value):
            keys['lower']=keys['lower']+1
        else:
            keys['other']=keys['other']+1
        pass

    return keys



def process_map(filename):
    keys = {"lower": 0, "lower_colon": 0, "problemchars": 0, "other": 0}
    for _, element in ET.iterparse(filename):
        keys = key_type(element, keys)

    return keys

print process_map('jerusalem_israel.osm')
```

```
{'problemchars': 1, 'lower': 120538, 'other': 13502, 'lower_colon': 4334
3}
```

In [3]:
```
import xml.etree.ElementTree as ET
import pprint
import re


def process_map(filename):
    users = set()
    for _, element in ET.iterparse(filename):
        for e in element:
            if 'uid' in e.attrib:
                users.add(e.attrib['uid'])

    return users
users = process_map('jerusalem_israel.osm')
print len(users), "users in osm file in total"
```

```
513 users in osm file in total
```

In [4]:
```
def find_elem_kval(kvals_to_inspect):
    context = ET.iterparse('jerusalem_israel.osm', events=('start', 'end'))
    _, root = next(context)
    for event, elem in context:
        if event == 'end' and elem.tag in ('node', 'way', 'relation'):
            for tag in elem.findall("tag"):
                if (tag.get("k") in kvals_to_inspect):
                    print elem.tag+" id:"+elem.attrib["id"]
                    for tag in elem.findall("tag"):
                        print "\t"+tag.tag+": k="+tag.get("k"), "v="+tag.get
                    print "*********************************************"
            root.clear()

find_elem_kval(["FIXME"])
print ""
find_elem_kval(["addr:street2","addr2:street"])
print ""
find_elem_kval(["name:be-tarask"])
```

In [5]:
```
# -*- coding: utf-8 -*-
import xml.etree.cElementTree as ET
import codecs
from collections import defaultdict
import re
import pprint

OSMFILE = 'jerusalem_israel.osm'
street_type_re = re.compile(r'\b\S+\.?$', re.IGNORECASE)

expected = ["Street", "Avenue", "Boulevard", "Drive", "Court", "Place", "Squ
            "Trail", "Parkway", "Commons", "Lane"]
# UPDATE THIS VARIABLE
mapping = { "St": "Street",
            "St.": "Street",
            'Rd.': 'Road',
            'Ave': 'Avenue',
            'Pkwy': 'Parkway',
            'Dr': 'Drive',
            'Dr.': 'Drive',
            'Exressway': 'Expressway',
            'Expessway': 'Expressway',
            'Trl': 'Trail',
            'Blvd': 'Boulevard',
            'Blvd.': 'Boulevard',
            }
def audit_street_type(street_types, street_name):
    m = street_type_re.search(street_name)
    if m:
        street_type = m.group()
        if street_type not in expected:
            street_types[street_type].add(street_name)

# returns true if an element contains a street value
def is_street_name(elem):
    return (elem.attrib['k'] == "addr:street")


# function that reads osm file line by line and finds street names to audit
def audit(osmfile):
    osm_file = open(osmfile, "r")
    street_types = defaultdict(set)
    for event, elem in ET.iterparse(osm_file, events=("start",)):
        if elem.tag == "node" or elem.tag == "way":
            for tag in elem.iter("tag"):
                if is_street_name(tag):
                    audit_street_type(street_types, tag.attrib['v'])

    return street_types

# function to change incorrect street types to correct street types
def update_name(name, mapping):
    m = street_type_re.search(name)
    if m:
        street_type = m.group()
        if street_type in mapping:
```

```
            name = name.replace(street_type, mapping[street_type])

    return name


def test():
    st_types = audit(OSMFILE)
    pprint.pprint(dict(st_types)) #print out dictonary of potentially incor

    for st_type, ways in st_types.iteritems():
        for name in ways:
            if street_type_re.search(name).group() in mapping:
                better_name = update_name(name, mapping)
                print name, "=>", better_name #print out street names that w

if __name__ == '__main__':
    test()
```

```
{'Agripas': set(['Agripas']),
 'Al-Amid': set(['Umm Al-Amid']),
 'Al-Bayan': set(['Al-Bayan']),
 'Al-Masharif': set(['Al-Masharif']),
 'Al-Mask': set(['Al-Mask']),
 u'Aljoz': set([u'\u0646\u0647\u0627\u064a\u0629 \u0627\u0644\u0645\u0646
\u0637\u0642\u0629 \u0627\u0644\u0635\u0646\u0627\u0639\u064a\u0629  endi
ng of the Industrial zone Wadi Aljoz']),
 'Anata': set(['Anata']),
 'Anatot': set(['Anatot']),
 'Bachar': set(['Nissim Bachar']),
 'Bir-As-Sabil': set(['Bir-As-Sabil']),
 'Botta': set(['Paul Emile Botta']),
 u'Dolorosa': set([u'\u05d5\u05d9\u05d4 \u05d3\u05d5\u05dc\u05d5\u05e8\u0
5d5\u05d6\u05d4 - Via Dolorosa']),
 'El-Sahel': set(['El-Sahel']),
 'Hanina': set(['Beit Hanina']),
 'Hatut': set(['Hatut']),
 u'Het': set([u"\u05d4\u05e2'\u05d7 - HaAin Het"]),
 'Hospital': set(['Located on Mt. Scopus Near Augusta Victoria Hospita
l']),
 u'Israel': set([u'\u05e9\u05d1\u05d8\u05d9 \u05d9\u05e9\u05e8\u05d0\u05d
c - Shivtei Israel']),
 'Jabal': set(["Mu'adh Bin Jabal"]),
 u'Joz': set([u'\u0628\u062f\u0627\u064a\u0629 \u0627\u0644\u0645\u0646\u0
637\u0642\u0629 \u0627\u0644\u0635\u0646\u0627\u0639\u064a\u0629 (\u0648
\u0627\u062f\u064a \u0627\u0644\u062c\u0648\u0632)  begining of the indus
trial zone Wadi al Joz']),
 'Mall': set(['Mamila Mall']),
 'Mamila': set(['Mamila']),
 u'Mountain': set([u'\u0634\u0627\u0631\u0639 \u064a\u0624\u062f\u064a \u
0625\u0644\u0649 \u062c\u0628\u0644 \u0627\u0644\u0631\u0648\u064a\u0633
\u0627\u062a Street leads to Ruwaisat Mountain']),
 'Rd.': set(['Hebron Rd.']),
 'St': set(['Dorot Rishonim St']),
 'St.': set(['Beit Hanina St.', "Hanevi'im St."]),
 'Wakaleh': set(['Al Wakaleh']),
 'Yafo': set(['Yafo']),
 'al-Dur': set(['Shajarat al-Dur']),
 'housing': set(['Al zahra housing']),
 u'land': set([u'\u0637\u0631\u064a\u0642 \u064a\u0624\u062f\u064a \u0625
\u0644\u0649 \u0627\u0644\u0623\u0631\u0636\u064a \u0627\u0644\u0643
2\u0631\u0627\u0639\u064a\u0629 Road leading to the agricultural land']),
 'road': set(['Nablus road']),
 u'school': set([u'\u0634\u0627\u0631\u0639 \u0641\u0631\u0639\u064a / \u
0637\u0631\u064a\u0642 \u0645\u062e\u062a\u0635\u0631 \u0644\u0644\u0645
\u062f\u0631\u0633\u0629 A side street / shortcut to the school']),
 'st': set(['Azzahra st']),
 'street': set(['al makhaniq street'])}
Hanevi'im St. => Hanevi'im Street
Beit Hanina St. => Beit Hanina Street
Hebron Rd. => Hebron Road
Dorot Rishonim St => Dorot Rishonim Street
```

In [6]:

```
#!/usr/bin/env python
# -*- coding: utf-8 -*-
import xml.etree.ElementTree as ET
import pprint
import re
import codecs
import json
from pymongo import MongoClient


lower = re.compile(r'^([a-z]|_)*$')
lower_colon = re.compile(r'^([a-z]|_)*:([a-z]|_)*$')
problemchars = re.compile(r'[=\+/&<>;\'"\?%#$@\,\. \t\r\n]')

CREATED = [ "version", "changeset", "timestamp", "user", "uid"]

def is_address(elem):
    if elem.attrib['k'][:5] == "addr:":
        return True



def shape_element(element):
    node = {}

    if element.tag == "node" or element.tag == "way" :

      node['type'] = element.tag

      # Parse attributes
      for a in element.attrib:
        if a in CREATED:
          if 'created' not in node:
            node['created'] = {}
          node['created'][a] = element.attrib[a]

        elif a in ['lat', 'lon']:
          if 'pos' not in node:
            node['pos'] = [None, None]

          if a == 'lat':
            node['pos'][0] = float(element.attrib[a])
          else:
            node['pos'][1] = float(element.attrib[a])

        else:
          node[a] = element.attrib[a]

      # Iterate tag children
      for tag in element.iter("tag"):
        if not problemchars.search(tag.attrib['k']):
          # Tags with single colon
          if lower_colon.search(tag.attrib['k']):

            # Single colon beginning with addr
```

```
            if tag.attrib['k'].find('addr') == 0:
              if 'address' not in node:
                node['address'] = {}

              sub_attr = tag.attrib['k'].split(':', 1)
              node['address'][sub_attr[1]] = tag.attrib['v']

            # All other single colons processed normally
            else:
              node[tag.attrib['k']] = tag.attrib['v']

          # Tags with no colon
          elif tag.attrib['k'].find(':') == -1:
            node[tag.attrib['k']] = tag.attrib['v']

      # Iterate nd children
      for nd in element.iter("nd"):
        if 'node_refs' not in node:
          node['node_refs'] = []
        node['node_refs'].append(nd.attrib['ref'])

      return node
    else:
      return None


def process_map(file_in, pretty = False):
    # You do not need to change this file
    file_out = "{0}.json".format(file_in)
    data = []
    with codecs.open(file_out, "w") as fo:
        for _, element in ET.iterparse(file_in):
            el = shape_element(element)
            if el:
                data.append(el)
                if pretty:
                    fo.write(json.dumps(el, indent=2)+"\n")
                else:
                    fo.write(json.dumps(el) + "\n")
    return data

def test():
    # NOTE: if you are running this code on your computer, with a larger dat
    # call the process_map procedure with pretty=False. The pretty=True opti
    # additional spaces to the output, making it significantly larger.
    data = process_map('jerusalem_israel.osm', False)
    print len(data)
    pprint.pprint(data[10])
    pprint.pprint(data[-10])


if __name__ == "__main__":
    test()
```

```
710686
{'created': {'changeset': '22153763',
             'timestamp': '2014-05-05T19:51:57Z',
             'uid': '385027',
             'user': 'Ori952',
             'version': '5'},
 'id': '29942465',
 'pos': [31.7766745, 35.22721],
 'type': 'node'}
{'building': 'yes',
 'created': {'changeset': '41795805',
             'timestamp': '2016-08-30T09:52:18Z',
             'uid': '189946',
             'user': 'BMM994',
             'version': '1'},
 'id': '439863338',
 'node_refs': ['4375400348',
               '4375400293',
               '4375400279',
               '4375400264',
               '4375400300',
               '4375400262',
               '4375400294',
               '4375400254',
               '4375400249',
               '4375400263',
               '4375400325',
               '4375400353',
               '4375400351',
               '4375400329',
               '4375400350',
               '4375400276',
               '4375400352',
               '4375400290',
               '4375400304',
               '4375400336',
               '4375400334',
               '4375400330',
               '4375400332',
               '4375400348'],
 'type': 'way'}
```

In [17]:
```
#!/usr/bin/env python
# -*- coding: utf-8 -*-
import json
from pymongo import MongoClient
client = MongoClient("mongodb://localhost:27017")
db = client.openstreetmap

def insert_json(infile, db):
    jsonfile=open(infile)
    for dic in jsonfile:
        data=json.loads(dic)
        db.maps.insert_one(data)
insert_json('jerusalem_israel.osm.json',db)
```

In [22]:
```
client = MongoClient('localhost:27017')
db = client.peru
collection = db.map_data

with open('jerusalem_israel.osm.json', 'r') as f:
    for row in f:
        data = json.loads(row)
        db.map_data.insert(data)
```

```
/Users/jaydenyuen/anaconda/envs/DAND/lib/python2.7/site-packages/ipykerne
l/__main__.py:8: DeprecationWarning: insert is deprecated. Use insert_one
 or insert_many instead.
```

In [62]:
```
database_stats = db.command("dbstats")
userids = collection.distinct("created.uid")

# find number of elements associated with each user and outputs the latest t
user_contribution = collection.aggregate([{"$group": {"_id": "$created.user"
                      {"$sort": {"entries":-1}}])

# data for users that have contributed less than 10 elements to the data set
userslt10_contribution = collection.aggregate([{"$group":
                                                {"_id": "$created.user",
                                                 "max":{"$max":"$created
                      {"$match": {"entries":{"$lt":10}}},
                      {"$sort":{"max":-1}}])

# WAY AND NODE ELEMENT ANALYSIS
# outputs distinct element types in data set: only should output way and nod
element_types = collection.distinct("type")

# total documents in the database
element_count = collection.count()

# counts number of elements created by year
timestamps = collection.aggregate([{"$group": {"_id": {"$substr": ["$created
                                  {"$sort": {"_id":1}}])


# AMENITY ANALYSIS: SCHOOLS, CHURCHES, RESTAURANTS, ETC.
# list of all amenities included in the data base
distinct_amenity = collection.distinct("amenity")

# counts of entries for each amenity type
amenity_count = collection.aggregate([{"$match": {"amenity":{"$exists":1}}},
                      {"$group": {"_id": "$amenity", "count":{"$sum":1}}},
                      {"$sort": {"count":-1}}])
```

In [52]:
```
print database_stats
```

```
{u'storageSize': 51752960.0, u'ok': 1.0, u'avgObjSize': 235.2108554270099
6, u'views': 0, u'db': u'peru', u'indexes': 1, u'objects': 710686, u'coll
ections': 1, u'numExtents': 0, u'dataSize': 167161062.0, u'indexSize': 62
66880.0}
```

In [53]:
```
print(list(user_contribution))
```

In [37]:
```
print element_types
```

```
[u'node', u'palm', u'Waypoint', u'way', u'Old ecomuseum, Restaurant', u'B
asic school for Boys', u'Boulangerie', u'Macon', u'Village Council Buildi
ng', u'Electricien', u'Mason', u'Secondary School for Boys', u'Garage',
 u'Forgeron', u'Basic school for Girls', u'Cyber Cafe', u'0ld Mosque',
 u'Epicerie', u'Secondary school for Girls']
```

In [38]:
```
print element_count
```

```
710686
```

In [41]:
```
print (list(timestamps))
```

```
[{u'count': 710, u'_id': u'2007'}, {u'count': 419, u'_id': u'2008'}, {u'c
ount': 12236, u'_id': u'2009'}, {u'count': 4372, u'_id': u'2010'}, {u'cou
nt': 47366, u'_id': u'2011'}, {u'count': 65255, u'_id': u'2012'}, {u'coun
t': 93660, u'_id': u'2013'}, {u'count': 183260, u'_id': u'2014'}, {u'coun
t': 93935, u'_id': u'2015'}, {u'count': 209473, u'_id': u'2016'}]
```

In [44]:
```
print distinct_amenity
```

```
[u'place_of_worship', u'embassy', u'restaurant', u'atm', u'bus_station',
 u'fuel', u'police', u'bench', u'school', u'hospital', u'parking', u'toil
ets', u'kindergarten', u'post_office', u'public_building', u'bank', u'tow
nhall', u'arts_centre', u'cafe', u'nightclub', u'car_rental', u'pharmac
y', u'shop', u'library', u'doctors', u'college', u'car_wash', u'taxi',
 u'bureau_de_change', u'car_sharing', u'recycling', u'university', u'fire
_station', u'theatre', u'fast_food', u'pub', u'telephone', u'grave_yard',
 u'post_box', u'cinema', u'studio', u'waste_basket', u'waste_disposal',
 u'fountain', u'courthouse', u'monastery', u"children's home", u'0', u'dr
inking_water', u'swimming_pool', u'bbq', u'dentist', u'youth_centre', u'd
riving_school', u'vending_machine', u'community_centre', u'NGO', u'clini
c', u'bar', u'veterinary', u'marketplace', u'parking_entrance', u'social_
facility', u'insitute', u'childcare', u'internet_cafe', u'social_centre',
 u'shelter', u'events_venue', u'Archive', u'amphitheater', u'retirement_h
ome', u'ritual_bath', u'\u05e8\u05de\u05ea \u05d4\u05d7\u05d9\u05e0\u05d5
\u05da', u'parking_space']
```

In [48]:
```
print (list(amenity_count))
```

```
[{u'count': 606, u'_id': u'school'}, {u'count': 566, u'_id': u'place_of_w
orship'}, {u'count': 457, u'_id': u'parking'}, {u'count': 132, u'_id':
 u'restaurant'}, {u'count': 81, u'_id': u'public_building'}, {u'count': 7
2, u'_id': u'fuel'}, {u'count': 66, u'_id': u'cafe'}, {u'count': 61, u'_i
d': u'doctors'}, {u'count': 61, u'_id': u'bank'}, {u'count': 58, u'_id':
 u'college'}, {u'count': 54, u'_id': u'hospital'}, {u'count': 48, u'_id':
 u'pharmacy'}, {u'count': 46, u'_id': u'drinking_water'}, {u'count': 45,
 u'_id': u'kindergarten'}, {u'count': 41, u'_id': u'toilets'}, {u'count':
 41, u'_id': u'townhall'}, {u'count': 39, u'_id': u'fast_food'}, {u'coun
t': 28, u'_id': u'community_centre'}, {u'count': 23, u'_id': u'clinic'},
 {u'count': 23, u'_id': u'library'}, {u'count': 22, u'_id': u'waste_baske
t'}, {u'count': 21, u'_id': u'post_office'}, {u'count': 21, u'_id': u'rec
ycling'}, {u'count': 21, u'_id': u'police'}, {u'count': 21, u'_id': u'ben
ch'}, {u'count': 18, u'_id': u'atm'}, {u'count': 16, u'_id': u'theatre'},
 {u'count': 15, u'_id': u'university'}, {u'count': 15, u'_id': u'embass
y'}, {u'count': 15, u'_id': u'bus_station'}, {u'count': 14, u'_id': u'fou
ntain'}, {u'count': 13, u'_id': u'pub'}, {u'count': 13, u'_id': u'taxi'},
 {u'count': 13, u'_id': u'arts_centre'}, {u'count': 11, u'_id': u'car_was
h'}, {u'count': 10, u'_id': u'post_box'}, {u'count': 10, u'_id': u'grave_
yard'}, {u'count': 9, u'_id': u'swimming_pool'}, {u'count': 8, u'_id':
 u'cinema'}, {u'count': 8, u'_id': u'car_rental'}, {u'count': 8, u'_id':
 u'bureau_de_change'}, {u'count': 7, u'_id': u'dentist'}, {u'count': 6,
 u'_id': u'waste_disposal'}, {u'count': 5, u'_id': u'shelter'}, {u'coun
t': 5, u'_id': u'parking_entrance'}, {u'count': 5, u'_id': u'driving_scho
ol'}, {u'count': 5, u'_id': u'childcare'}, {u'count': 5, u'_id': u'courth
ouse'}, {u'count': 4, u'_id': u'bar'}, {u'count': 4, u'_id': u'telephon
e'}, {u'count': 4, u'_id': u'veterinary'}, {u'count': 4, u'_id': u'fire_s
tation'}, {u'count': 3, u'_id': u'internet_cafe'}, {u'count': 3, u'_id':
 u'marketplace'}, {u'count': 3, u'_id': u'studio'}, {u'count': 2, u'_id':
 u'events_venue'}, {u'count': 2, u'_id': u'shop'}, {u'count': 2, u'_id':
 u'bbq'}, {u'count': 2, u'_id': u'nightclub'}, {u'count': 2, u'_id': u'mo
nastery'}, {u'count': 1, u'_id': u'parking_space'}, {u'count': 1, u'_id':
 u'ritual_bath'}, {u'count': 1, u'_id': u'amphitheater'}, {u'count': 1,
 u'_id': u'insitute'}, {u'count': 1, u'_id': u'vending_machine'}, {u'coun
t': 1, u'_id': u'youth_centre'}, {u'count': 1, u'_id': u'0'}, {u'count':
 1, u'_id': u'Archive'}, {u'count': 1, u'_id': u"children's home"}, {u'co
unt': 1, u'_id': u'car_sharing'}, {u'count': 1, u'_id': u'retirement_hom
e'}, {u'count': 1, u'_id': u'\u05e8\u05de\u05ea \u05d4\u05d7\u05d9\u05e0
\u05d5\u05da'}, {u'count': 1, u'_id': u'social_centre'}, {u'count': 1,
 u'_id': u'social_facility'}, {u'count': 1, u'_id': u'NGO'}]
```