# OpenStreetMap Data Case Study

## Map Area

Suginami,Tokyo,Japan

- mapzen tokyo

# Problems Encountered in the Map

After initially downloading a small sample size of the Tokyo area and running it against a provisional data.py file, I noticed five main problems with the data, which I will discuss in the following order:

- The same thing, but various names and abbreviations are used.

`ex`

```
<tag k="name" v="有明ジャンクション"/>
<tag k="highway" v="motorway_junction"/>

<tag k="name" v="芝浦JCT"/>
<tag k="highway" v="motorway_junction"/>

<tag k="name" v="成田ＪＣＴ"/>
```

`ex2`

```
<tag k="wikipedia" v="ja:川口東インターチェンジ"/>
<tag k="name" v="千葉北ＩＣ"/>
<tag k="name" v="湾岸習志野IC"/>
```

- Japanese is used for Japanese data set. Therefore, it may be necessary to perform processing specific to Japanese.(Russian and many other languages are included.
  )

`ex`

```
<tag k="name" v="下高井戸敬老会館"/>

 <node id="31236676" lat="35.634834" lon="139.7687577" version="3" timestamp="2014-12-07T22
```

- High way where class is not registered

`ex`

```
<tag k="highway" v="unclassified"/>
```

- Mysterious note

`ex`

```
<tag k="note" v="estimated"/>
```

- abbreviation

`ex`

```
<tag k="name" v="府中駅北口"/>
<tag k="highway" v="traffic_signals"/>
<tag k="name:en" v="Fuchu Station North Entrance"/>

<tag k="name" v="西府駅入口"/>
<tag k="name:en" v="Nishifu Sta. Ent."/>

<tag k="name" v="日野橋"/>
<tag k="name:en" v="Hino brdg."/>
```

# Data Audit

First of all, I decided to audit the XML file.
TODO

`data.py`

## Unique Tags

As the beginning of the audit, I measured unique tags.

```
{'node': 864177, 'nd': 1055982, 'bounds': 1, 'member': 15818, 'tag': 560767, 'relation': 79
```

## Patterns in the Tags

Then I examined the type of tag.

```
{'problemchars': 0, 'lower': 498542, 'other': 9043, 'lower_colon': 53182}
```

## Change to the correct name

`python3-code/audit.py`

```
udagawacho → '宇田川町'
takadanobaba → '高田馬場'
sakuragaoka-cho →  '桜ヶ丘町'
dogenzaka → '道玄坂'
jingumae →  '神宮前'
shinsen → '神泉'
Matsubara →  '松原'
Nerima → '練馬'
Omotesando → '表参道'
```

# Process your Dataset

## data to csv

Converting data from XML to CSV format. Then import the cleaned .csv files into a SQL database

`data_to_csv.py`

## create database

`create_database.py`

# Explore your Database

`query.py`

- size of the file
- number of unique users
- number of nodes and ways
- number of chosen type of nodes, like cafes, shops etc.

## size of the file

## number of unique users

`query.py`

```
sqlite> SELECT COUNT(DISTINCT(e.uid))
FROM (SELECT uid FROM nodes UNION ALL SELECT uid FROM ways) e;
```

```
Number of unique users:  15
```

## number of nodes and ways

```
sqlite> SELECT COUNT(*) FROM nodes
```

```
sqlite> SELECT COUNT(*) FROM ways
```

```
Number of nodes:  815
Number of ways:  184
```

## number of chosen type of nodes, like cafes, shops etc.

`Top contributing users`

```
sqlite> SELECT e.user, COUNT(*) as num
FROM (SELECT user FROM nodes UNION ALL SELECT user FROM ways) e
GROUP BY e.user
ORDER BY num DESC
LIMIT 10;
```

```
Top contributing users:  [(u'roguish', 233), (u'ribbon', 196), (u'Nahainec', 186), (u'yoshi
```

# Ideas for additional improvements

I will describe two additional problems related to improvement.

`The value of input varies depending on the person.`

We should prepare some conventions for those who register. For example IC, ic,インターチェンジ unified in IC.

`Even the same thing means that the number of input information is different.`

Likewise, Should be suggested to some extent what value should be entered.We can not remember this, so I think that it is better to encourage using the system.For example, when you enter a bridge, items to be entered are displayed.

# Files