**OpenStreetMap**

**Table of Contents**

1. **Intro to Importing into OSM 2**
   1. Relevant Links 2
   2. Sample Tagging Info 2
2. **Using JOSM Editor 3**
   1. Downloading Plugins 3
   2. Adding Layers 3
   3. Fixing Tags/Attributes 3
   4. Proper Tagging for Buildings Layer 5
   5. Downloading OSM Data 5
   6. Conflating the Data 6
      1. For Point Data 6
      2. For Polygon Data 8
3. **Scripts 11**
   1. Link to Scripts 11
   2. Descriptions of Scripts 11
   3. Order to Use the Scripts 11

**Intro to Importing into OSM**

<https://wiki.openstreetmap.org/wiki/Import/Guidelines>

<https://wiki.openstreetmap.org/wiki/Import/Catalogue>

<https://wiki.openstreetmap.org/wiki/M-NCPPC_Planning_Department>

**OSM Tags**

<https://wiki.openstreetmap.org/wiki/Map_Features>

<https://wiki.openstreetmap.org/wiki/How_to_map_a>

<https://wiki.openstreetmap.org/wiki/Category:Key_descriptions_by_group>

example: Libraries Point File → OSM Standards

ADDRESS → addr:housenumber + addr:street

CITY → addr:city

STATE → addr:state

ZIPCODE → addr:postcode

NAME → name

TELEPHONE → phone (also change from xxx-xxx-xxxx to +1-xxx-xxx-xxx)

FACILITY\_TYPE → operator

Attributes that need to be added

amenity=library (from Map Features List)

source=M-NCPPC

Specific info on address tags: <https://wiki.openstreetmap.org/wiki/Key:addr>

**Editing field names is easy in the JOSM Editor**

Detailed below

**Editing Telephone is easier in ArcMap**

1. Open Attribute table and create new field (Name = phone, Type = string)
2. Use Field Calculator with following expression

"+1-" + [TELEPHONE]

1. Delete old TELEPHONE field

**I have created a python script that will do this automatically**

**To import large files, such as the building polygon layer, it must be split into much smaller areas and ideally have an address point associated with it whenever possible**

**I have also created a python script to split the buildings layer by election precinct (which is a small enough area) and merge it with the address point layer**

**Using JOSM**

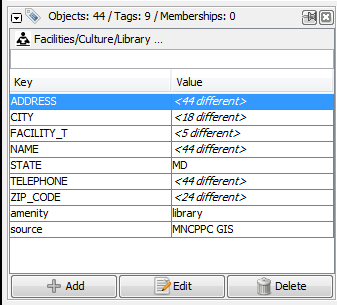
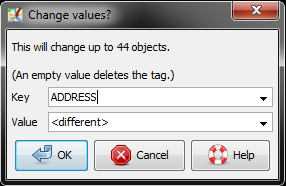
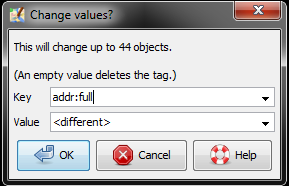
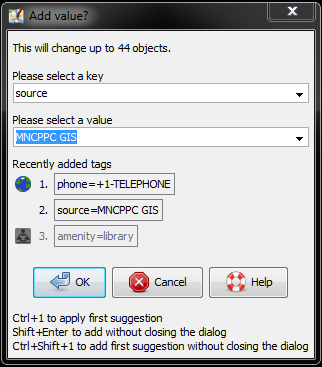
**Downloading Plugins**

1. Download [JOSM](https://josm.openstreetmap.de/)
2. Open, and click edit and preferences
3. Click on the fourth button on the left bar (Plugins)
4. Search for the necessary plugins
   * [Conflation](https://wiki.openstreetmap.org/wiki/JOSM/Plugins/Conflation)
   * [OpenData](https://wiki.openstreetmap.org/wiki/JOSM/Plugins/OpenData)
   * [Reverter](https://wiki.openstreetmap.org/wiki/JOSM/Plugins/Reverter)
5. Check the checkmark button and click download list
6. Click OK and restart JOSM

**Adding Layer**

1. Make sure you have OpenData Plugin installed
2. Click the file icon in the top left  (ctrl+o)
3. Navigate to the proper shapefile and open it
4. Select all the objects and change the necessary tags

**Fixing Tags/Attributes**

1. Open the shapefile (ctrl+o)
2. Select all (ctrl+a)
3. On the right is an area where the tags are located and can be edited 
4. Double click a tag to edit and a new window opens 
5. Change the Key field (it will autocomplete with relevant field) 
6. Do not change the Value field and hit OK
7. To add a field click the + Add button on bottom left of the Tag Window and a new window will open
8. Type in Key name (ex: Source) and Value name (ex: MNCPPC GIS) and hit OK

**Proper Tagging for the Buildings Layer**

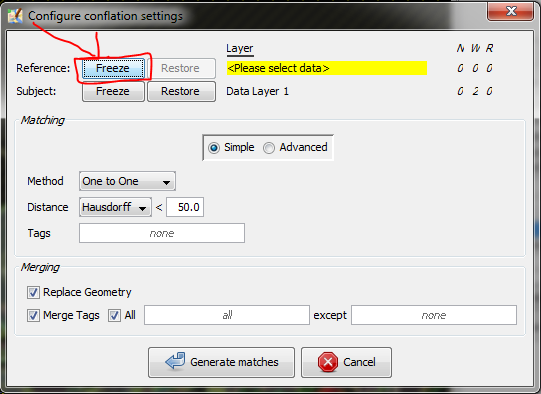
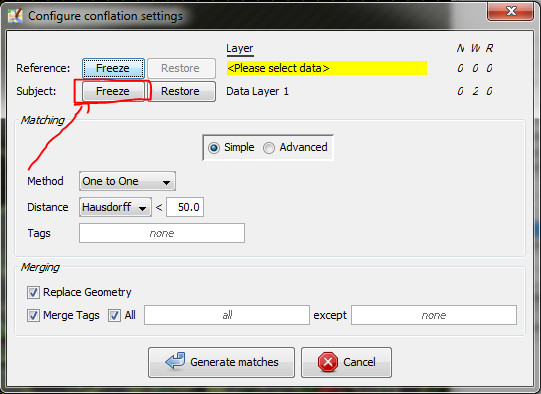
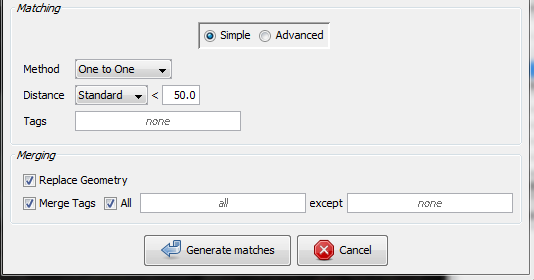
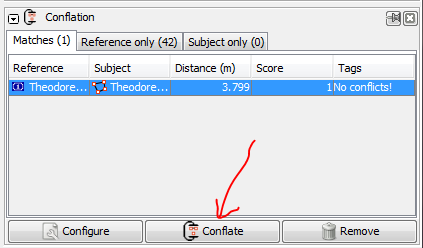
|  |  |  |
| --- | --- | --- |
| Buildings Merged w/ Address |  | OMS Standards |
| COMPLETE\_A | → | addr:housenumber |
| COMPLETE\_S | → | addr:street |
| COMPLETE\_1 | → | addr:flats |
| PLACE\_NAME | → | addr:city |
| STATE\_NAME | → | addr:state |
| ZIP\_CODE | → | addr:postcode |

**Downloading OSM Data**

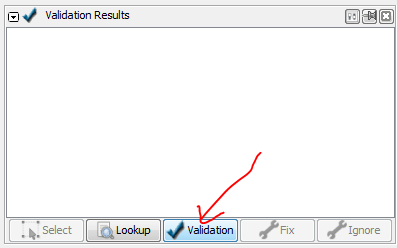
1. Click the download button  (ctrl+shift+down)
2. This will open a new window to download the OSM data
   1. JOSM will highlight the area you have zoomed to (or you can manually select the area)
   2. If the file has a small enough area, it can be downloaded in one chunk
      1. Else, you will have to download multiple chunks
      2. If uploading a file such as libraries or DMV locations that have a few points but spread out over a large area, zoom in close onto each point/group of points and then click the download button to download that small area. Repeat this for each point
3. If working with a point file, check the box to download as a new layer. Else, uncheck that box
4. Click Download

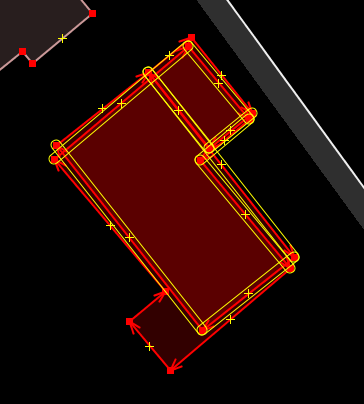
**Conflating the Data**

For Point Data

1. Add the point data layer and download the OSM data in JOSM
   * Be sure to check the box to download as new layer
2. With the OSM data layer selected either select all (ctrl+a) or go to the presets tab along the top bar and select the Search For Objects by Preset Tool (shift+F3)
3. Search for the type of data you are importing (ex: Library)
4. Select the conflate tool from the left-hand bar to open the Conflation window
5. Click the configure button
6. Next to the Reference tab click the Freeze button
7. Leaving the Configure window open, select the point data you wish to import and select all (ctrl+a)
8. Back in the configure window, select the Freeze button next to the Subject tab
9. Set the rest of the window with Method: “One to One” and Distance: “Standard < 50” 
10. Click generate matches
11. In the Conflation window, conflicts will be generated
12. Select the conflicts and click the conflate button
13. If needed, choose what data for the fields you want
14. Select both layers and right click and merge the layers into one layer
15. Upload the data

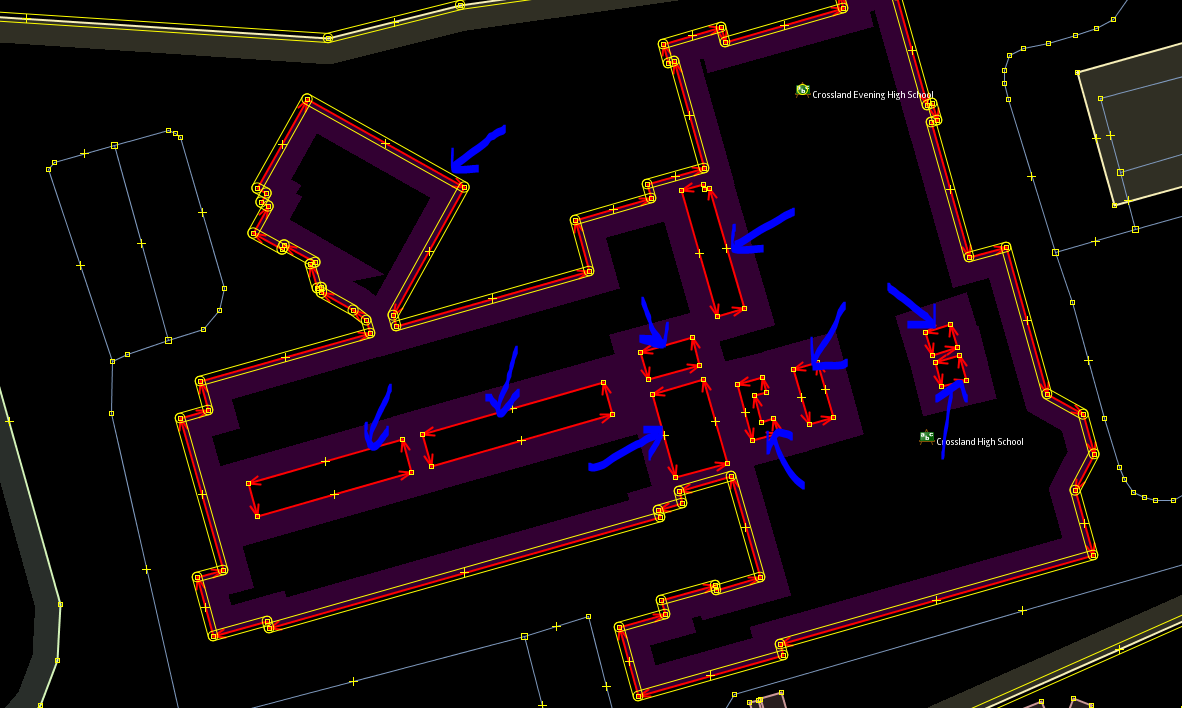
For Polygon Data

1. First add the new buildings layer, validate it, and import it as there should be no overlap
2. Add the intersecting layer and download the OSM data in JOSM
3. Make sure the data is on one single layer
4. Click the blue checkmark  in the left-hand bar to open the Validator window on the right-hand bar
5. In the Validation window click the validation button 
6. After a few moments the window will be populated with validation errors and warnings
7. If there are any “Building inside Building” or “Crossing Building” warnings, then double click the first one and use the Replace Geometry Tool (ctrl+shift+G)
   * If a window appears about conflicting tags, choose either “All” or choose the best fit and click the Apply button
   * If you come across an area of multiple overlaps, you will only be able to fix one using the Replace Geometry Tool (ctrl+shift+G). In order to fix the rest just delete all of the old polygons that still overlap the first fix.



There are three polygons here. Only two will be able to be merged using the Replace Geometry Tool (ctrl+shift+G). First, we use that tool where we can and then when it doesn’t work anymore we have to select the remaining old polygons and delete them.

* + To cycle through these quickly if there are a lot use the arrow down key and hit enter to select the next warning in the list and then click (ctrl+shift+G)
  + There is potentially a way to automate this process, but it requires installing a software that will auto create keystrokes (such as [AutoHotkey](https://autohotkey.com/))

1. If you encounter a complex polygon that will not be fixed by the Replace Geometry Tool you, such as a multipolygon
   * Select only the two outside polygons and then use the Replace Geometry Tool (ctrl+shift+G)
   * This will likely bring up a window. Select the Keep option under Decision
   * Click apply and the button of the window
   * Select all of the polygons in the multipolygon
   * Use the Update Multipolygon Tool (ctrl+shift+B)
   * Run the Validator again to ensure the issue is fixed
2. After fixing these warnings you should be able to upload freely

**Scripts**

<https://github.com/GregoryMu/M-NCPPC-Planning-Department>

**Descriptions of Scripts**

1. [buildingMergeByWSSCGrid.py](https://github.com/GregoryMu/M-NCPPC-Planning-Department/blob/master/buildingMergeByPrecinct.py)
   * This script allows you to input a building, address point, and existing OSM buildings file and will clip them to smaller sizes using the WSSC Grid
   * Next it joins the buildings and address point files and fixes most fields to OSM standards
   * Lastly it will split the individual buildings files into intersecting (buildings that already have OSM data) and new (buildings that are entirely new) building files.
2. [calcPhone.py](https://github.com/GregoryMu/M-NCPPC-Planning-Department/blob/master/calcPhone.py)
   * This script will convert a TELEPHONE field of a shapefile from xxx-xxx-xxxx format to +1-xxx-xxx-xxxx format for importing to OSM
3. [clipBuildings.py](https://github.com/GregoryMu/M-NCPPC-Planning-Department/blob/master/clipBuildings.py)
   * This script will clip the downloaded OSM buildings file to Prince George’s County
4. [download.py](https://github.com/GregoryMu/M-NCPPC-Planning-Department/blob/master/download.py)
   * This script will download the current OSM data for the State of Maryland and extracts that data to a folder
5. reproject.py
   * This script will reproject all the shapefiles in the selected workspace to the WGS 1984 format
6. [splitAddr.py](https://github.com/GregoryMu/M-NCPPC-Planning-Department/blob/master/splitAddr.py)
   * This script will take an address field from an ArcMap shapefile and split it into two fields, House Number and Street Name

**Order to Use the Scripts**

1. download.py
   * Use this first to download new data from OSM
   * This will take about 5-10 minutes
2. clipBuildings.py
   * Use this to clip the recently downloaded OSM data
3. reproject.py
   * Use this to ensure that all of the data is projected properly and up to date
   * This will take about 5-8 minutes
4. splitAddr.py and calcPhone.py
   * Use this to fix any attributes (of already projected files) that are possible
   * If needed, change the path name to fix any shapefiles needed
5. buildingMergeByWSSCGrid.py
   * Use this last in order to keep all of the data up to date
   * This script takes a long time to fully complete so if there is a certain range of data, change the parameters in the for loop to get the desired shapefiles. (ex: you only want data from grid 5-10, there are 658 grids)