# World2Xplane Beta 0.7.0

## Table of Contents

World2Xplane Beta 0.7.0 ........ 1
Table of Contents ........ 1

Introduction ........ 4
Requirements ........ 4
*Recommended System and Performance Tips* ........ *4*
Modes of Operation ........ 4
*Multiprocessing* ........ *5*
*Memory Mode* ........ *5*
*Local Storage Mode* ........ *5*
*Obtaining Data* ........ *5*
*Poly files for exclusions* ........ *6*

Starting the application ........ 7
Assigning Memory ........ 7
*Windows* ........ *7*
*Linux* ........ *8*
*Mac OS X / Command-Line* ........ *8*

Scenery Generation ........ 9
*Node* ........ *9*
*Way* ........ *9*
*Relation* ........ *9*
*3D Tags* ........ *9*
*Filter* ........ *9*
*Operation* ........ *10*

Using the Application ........ 11
Advanced Tab ........ 12
*Log Tab* ........ *14*

The configuration File ........ 15
Default Configuration Files ........ 15
*config.xml* ........ *15*
*config-hd.xml* ........ *15*
Global Config Options ........ 15
Rule Types ........ 18
*Forest Rule* ........ *18*
*Object Rule* ........ *18*
*Facade Rule* ........ *18*
*Line Rule* ........ *18*
*Random Rule* ........ *18*
*Sink Rule* ........ *19*
*Area Tracker Rule* ........ *19*
*Way Tracker Rule* ........ *19*
*3D Object Generation* ........ *19*

Regions ........ 20

ESRI Shapefiles 20
CSV Files 20
Plugins 20
Configuration 21

**Heights and Adaptive Heights 22**

**Filters 24**

*Example 1:* 25
*Example 2:* 25
And Logic Conditions 25
*Example 3:* 25
*Example 4:* 25
Or Logic Conditions 26
*Example 4:* 26
*Example 5:* 26
*Example 6:* 26
*Example 7:* 26

**Rules 28**

Regions 28

**Object Rules 29**

Placing Objects using a footprint 29
Placing Objects using best fit 29
Best fit CSV File 33

**Random Rules 35**

Placing Random Objects 35

**Lines Rules 38**

Placing Objects 38
*Example 1* 38
*Example 2* 38
*Example 3* 39

**Sink Rules 42**

Sinking Objects 42

**Facade Rules 44**

Placing Facades 44
Facade Distribution Rules 45
Options 46

**Polygon Rules 48**

Placing Polygons 48
Options 48

**Forest Rules 49**

Placing Forests 49

**Area Tracker Rules 51**

Tracking Areas 51

**Way Tracker Rules 53**

Tracking Ways 53

**Node Tracker Rules 54**

Tracking Nodes 54

**Command-Line Options** .......... **55**

*Examples* .......... *55*

## Introduction

World2XPlane is a tool used to generate X-Plane scenery using OSM (OpenStreetMap) data. The program takes as input an OSM .PBF file, and outputs a scenery folder which can be placed inside the X-Plane Custom Scenery folder.

**For a quick start guide and tutorials for using the application, please visit tutorials section of the website http://www.world2xplane.com**

### Requirements

A Windows/Linux or Mac OS X system capable of running Java and at least 4GB of available memory. The application will run best on a 64-Bit system, with at least 4GB of RAM and an SSD drive.

You should have a Java runtime installed and correctly configured on your system. Please see:

**(N.B. Always use the 64-bit version of Java if you have a 64-Bit CPU. Although it's possible to run the software on 32-bit computers, this is unsupported and you will need to configure memory and the startup scripts yourself)**

http://www.oracle.com/technetwork/java/javase/downloads/java-se-jre-7-download-432155.html

#### Recommended System and Performance Tips

The application is very memory and CPU intensive, and should be given as much memory as possible. Additionally, the following can also speed up generation and improve performance:

- Placing the OSM file and changing the .tmp file path in the config file to an SSD drive or memory disk will offer a big speed up in performance
- Setting the number of processor cores to whatever your system can spare. Using 2 cores will mean two instances of the generation will run at the same time, thus improving performance. However, this option only works well if used in combination with an SSD or memory disk.

### Modes of Operation

World2XPlane upon beginning generation will build a database and index of information it needs to generate the scenery. This index is needed to generate scenery quickly, as well as support OSM relations and areas.

The index by default is stored in main memory, however it can also be configured to be stored on disk. The differences between the two storage modes are indicated below:

## Multiprocessing

World2XPlane can be configured to run multiple generation instances, so that it will generate more than one tile at a time. However, the more instances that are ran, the more memory required. A good guide is start with the number of cores in your computer minus one, e.g. If you have a 4-core CPU, set this value to 3. If you find generation is hanging or unresponsive, try lowering the value.

## Memory Mode

In memory mode the application will use the memory available on your computer to store the indexes it needs to generate scenery. Using this mode is much faster, but requires a system with a lot of memory. For an OSM file of 512MB, it will require at least 2GB of free ram. If you receive out-of-memory exceptions, then you will need to choose the local storage mode in order to generate scenery.

Using memory mode, the system will need to rebuild the index if you decide to stop and start the generation.

## Local Storage Mode

In local storage mode, the application will create a database for storing indexes it needs to generate the scenery. This index will be stored on the local disk and will be slower than using memory alone. However, local storage mode requires much less memory and also supports resuming (if the index has been created previously). It's advisable to use local storage mode for files over 500MB.
To select local storage mode, check the Use Local Storage tick box in the GUI front-end.

## Obtaining Data

World2XPlane can read OSM PBF files, which is a quick and compressed format for storing OSM data. XML files are not supported, since these are inefficient and very slow to read. PBF files can be obtained from various sources on the Internet:

http://download.bbbike.org/osm/bbbike/ for city extracts
http://geofabrik.de/ for country and continent extracts

You can also convert to PBF from various other formats using the following free tools:
http://wiki.openstreetmap.org/wiki/Osmconvert
http://wiki.openstreetmap.org/wiki/Osmosis

This is useful when debugging small areas, such as an area downloaded inside JOSM.

It's also advisable to strip out data that is not needed to speed up parsing of the file, e.g. Stripping out names and addresses, or source tags can reduce the file size by around 30%. The following tools can be used to filter out unused tags/nodes:
http://wiki.openstreetmap.org/wiki/Osmosis
http://wiki.openstreetmap.org/wiki/Osmconvert

## Poly files for exclusions

Most sites which distribute OSM files also distribute a .poly file which describes the extent of the file. This file should also be obtained if possible, as it can be used to generate more accurate exclusions. Without a poly file, exclusion zones will be limited to one a single rectangle per 1x1 grid, and may block neighboring scenery, e.g. If you generate both France and Belgium as separate files, a large part of North-Eastern France will block out buildings and roads from Western Belgium.

Additionally, even if you enable smart-exclusions, you should still use it in combination with a .poly file, as smart-exclusions cannot be used alongside roads.

## Starting the application

In order to use the application, a Java runtime needs to be installed and configured correctly. Please see the requirements section for more details.

On Windows, launch the application by double-clicking on **World2XPlane.exe**.

On Linux, please use the provided shell scripts from the command-line **W2XP-Linux-64.sh**

The application can also be started from the command-line using a command such as:

**java -jar World2XPlane.jar**

By default, World2XPlane will launch to a GUI. To use via a command-line or script, please see the section “Command-Line options”

On some systems, the World2XPlane jar file can be double-clicked to open it, however you will need to configure the Java runtime to assign the correct amount of memory before doing so.

### Assigning Memory

By default, World2XPlane is only assigned a small amount of memory on some systems, and you may need to configure more memory. To do this, see the operating system specific sections below:

#### Windows

On Windows, memory can be configured using the World2XPlane.ini file. By default, the ini file tells Windows to assign World2XPlane as much memory as possible, however this may not be desirable, and *will also not work correctly if using World2XPlane in combination with a RAM Disk*.

The default .ini file contains the follow entries:

```
main.class=com.world2xplane.GUI.GUI
classpath.1=*.jar
vmarg.1=-d64
process.priority=high
vmarg.2=-XX:+AggressiveHeap
vmarg.3=-XX:+UseParallelGC
vmarg.4=-XX:+UseAdaptiveSizePolicy
```

The vmarg.* options are standard Java options used for configuring memory. To set the exact amount of memory used, you can change the .ini options to the example shown below:

```
main.class=com.world2xplane.GUI.GUI
classpath.1=*.jar
vmarg.1=-d64
process.priority=high
vmarg.2=-Xms8g
vmarg.3=-Xmx8g
vmarg.4=-XX:+UseParallelGC
```

The –Xms8G and –Xmx8g option tells Java to assign World2XPlane 8G. Change the numeric value here to however many gigabytes of RAM you wish to assign to the application.

## Linux

On Linux, you can use the provided bash script **W2XP-Linux-64.sh** to start the application. You will be prompted about your system and the memory will be set.

## Mac OS X / Command-Line

Alternatively, the application can be run from the command-line and the amount of memory configured and passed to Java, e.g.:

**java –d64 –Xms4g –Xmx4g -jar World2XPlane.jar**

Where 4g is 4GB of memory. Increase this to as much memory as possible.

On 64-Bit system, to use more than 4GB of RAM, you must tell the Java runtime to use 64-bit, e.g:

**java  -d64 –Xms6g –Xmx6g -jar World2XPlane.jar**

**For a quick start guide and tutorials for using the application, please visit the tutorials section of the website http://www.world2xplane.com**

## Scenery Generation

Before generating scenery, it is useful to understand how the application works and how it converts OSM data for use in X-Plane.

In OSM, there are three data types: Nodes, Ways and Relations:

### Node

A node is a single point inside OSM. This could be a tree, a church or a traffic light.

### Way

A way is a collection of nodes together used to form a line or a shape. A way can be open or closed (i.e. It forms a closed loop). An open way could be a road or railway line, a closed loop could be a building or a forest.

### Relation

A relation is a group of ways, which is used to construct complex shapes or groups of related items. Relations are used for complex buildings, such as those with many parts or inner courtyards, and also forests which are grouped together.

Every item inside OSM consists of a group of tags which describe what the object is, these are a simple key=value pair, some examples are shown below:

| | |
|---|---|
| **building=house,building:levels=2** | For a two storey house. |
| **highway=residential** | For a residential street |
| **amenity=place_of_worship** | For a church |

### 3D Tags

World2XPlane supports building simple 3D objects from the 3D tags inside OSM. Please see here http://wiki.openstreetmap.org/wiki/Simple_3D_Buildings for a list of tags and how to create buildings. This option is enabled via the **enable-osm-3d** configuration file option.

World2XPlane uses these tags to determine what to do with the data using a system of rules. A rule consists of two parts:

### Filter

A filter is a set of conditions to match OSM tags and some geometry information. A rule can consist of a single or multiple complex combinations of filters, e.g. To filter out forests with a particular variety of trees, or a particular height of building. Please see the section Filter Rules for more information and examples of filters.

## Operation

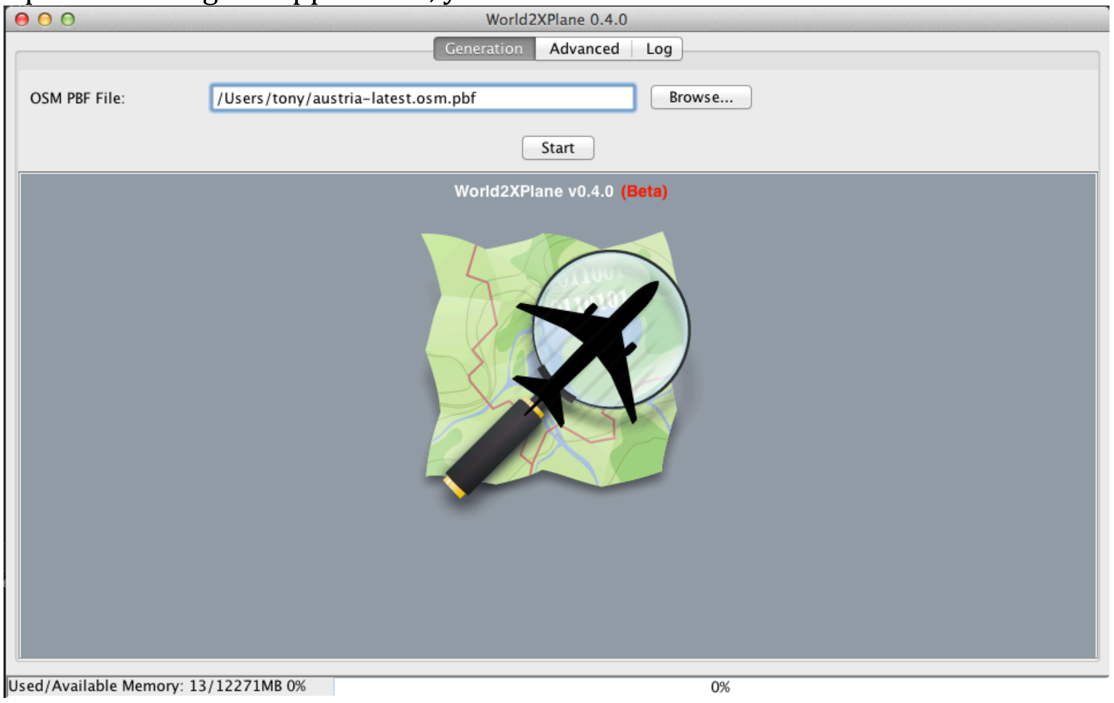
The operation part of a rule determines what to do if the filter matches the OSM object. Please see the section “Rule Types” for the operations that can be performed on objects.

## Using the Application

Before starting the application, make sure you have all required building libraries, the latest list of dependencies is always listed on the scenery page of the website http://www.world2xplane.com

N.B. As of 0.5.1. World2Xplane requires X-Plane to be installed and the world-models library installed.

Upon launching the application, you will see a screen similar to below:



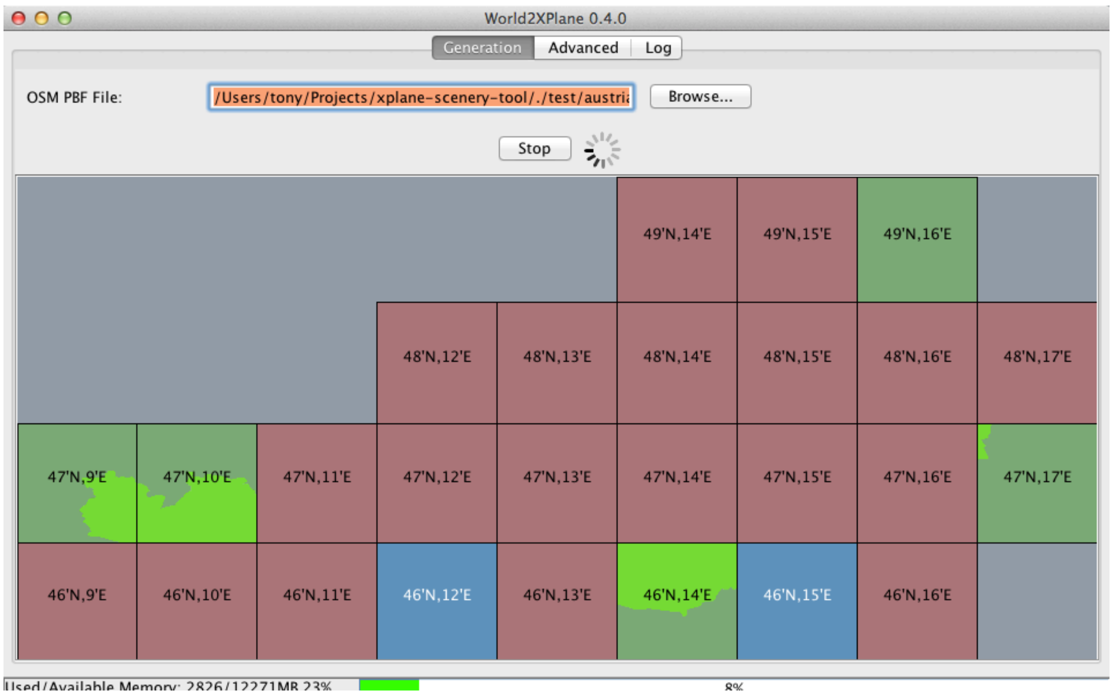
OSM PBF File: /Users/tony/austria-latest.osm.pbf

Used/Available Memory: 13/12271MB 0%

The Generation tab contains the location of the OSM file you wish to generate, and also a status window which will update during processing.

*N.B. World2XPlane at present only supports .osm.pbf files and not xml of o5m files.*

At the most, all you need to do is click **Browse,** locate the OSM PBF file you have obtained, and click on **Start**. Generation will run unattended, and you will be prompted if either an error occurs, or when generation has completed.



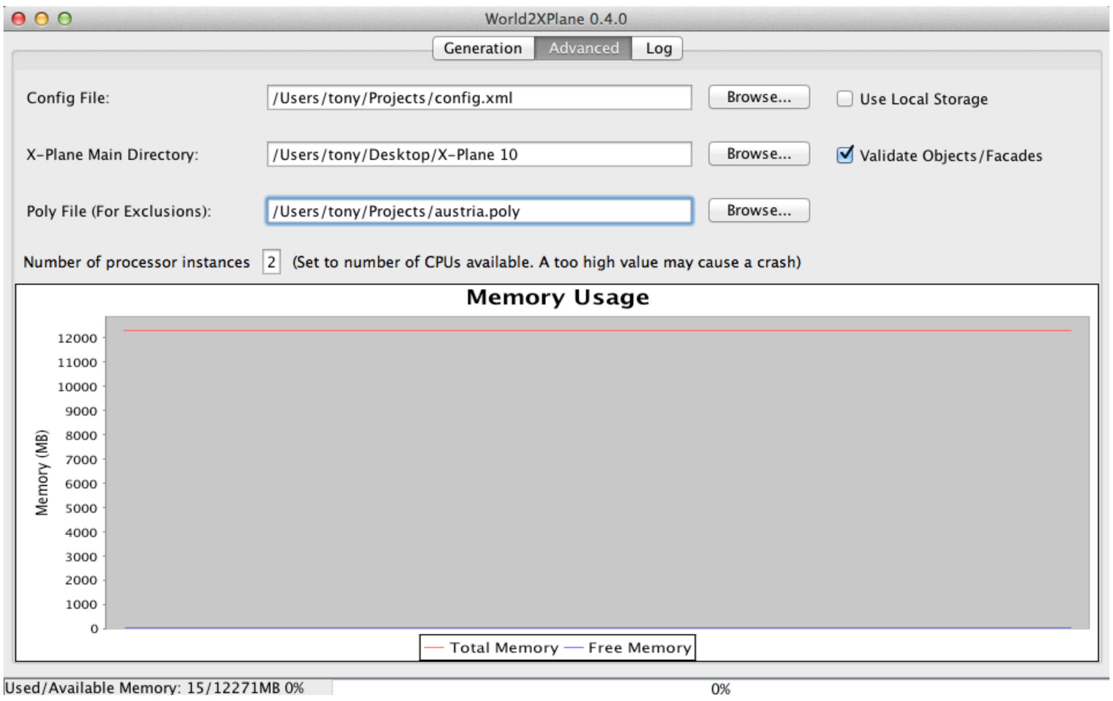
OSM PBF File: /Users/tony/Projects/xplane-scenery-tool/./test/austria

Used/Available Memory: 2826/12271MB 23%    8%

During generation, the progress window will show you a list of tiles and their current generation status. Items marked in red are pending, items in blue are currently being processed, and items marked in green are completed.

The status bar at the bottom of the page will also show you how much memory you are currently using, and the overall progress of generation.

## Advanced Tab



Config File: /Users/tony/Projects/config.xml

X-Plane Main Directory: /Users/tony/Desktop/X-Plane 10

Poly File (For Exclusions): /Users/tony/Projects/austria.poly

☐ Use Local Storage

☑ Validate Objects/Facades

Number of processor instances 2 (Set to number of CPUs available. A too high value may cause a crash)

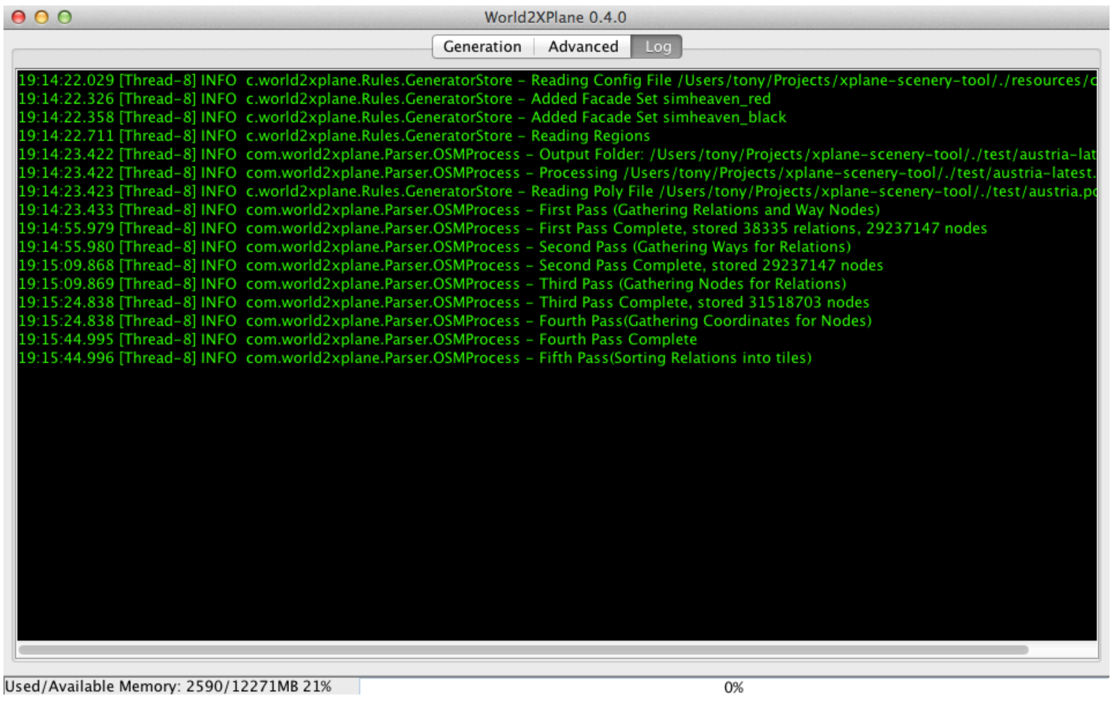
Used/Available Memory: 15/12271MB 0%    0%

The Advanced Tab contains some extra options and a memory graph, which can be used to change various options related to generation.

| | |
|---|---|
| **Config File** | This points to the location of the World2XPlane config file. You can change this to a different file if you have multiple config files available. By default, the config file provided with the application is in the resources folder. |
| **Use Local Storage** | This option will use your disk to store the index files used during generation, and should be used if the OSM file is larger than approx. 500MB, or if you receive out of memory errors. Using this option is generally slower than main memory. |
| **X-Plane Main Directory** | This option is optional and used to validate the scenery and configuration file you are generating against your X-Plane installation. |
| **Validate Objects/Facades** | This option, when used in combination with the **X-Plane Main Directory** folder will validate that all objects and facades referenced in your scenery actually exist inside your X-Plane directory. |
| **Poly File (For Exclusions)** | By default, World2XPlane will create an exclusion zone for the entire 1x1 degree tile which may produce undesirable effects when used with other scenery. E.g. If you generate France and later Belgium, the 1x1 western tile from Belgium may be blocked by a tile in France. If you choose a .poly file here (Normally available alongside the PBF file from various websites), then more accurate exclusions will be created. Using this option is highly recommended. Additionally, smart exclusions do not work with roads, so you will need a .poly file if you wish to generate exclusion zones for roads. |
| **Number of Processor Instances** | This option will set the number of tile generation processors than will run at the same time. This option can speed up generation, but will also use more memory and may cause out of memory errors. Try setting this value initially to number of cores – 1 you have, and see what works best. If you receive hangs, try decreasing the value. |

Most configuration for the application is currently done through the resources/config.xml file, and is covered in more detail in later sections:

## Log Tab



```
19:14:22.029 [Thread-8] INFO  c.world2xplane.Rules.GeneratorStore - Reading Config File /Users/tony/Projects/xplane-scenery-tool/./resources/c
19:14:22.326 [Thread-8] INFO  c.world2xplane.Rules.GeneratorStore - Added Facade Set simheaven_red
19:14:22.358 [Thread-8] INFO  c.world2xplane.Rules.GeneratorStore - Added Facade Set simheaven_black
19:14:22.711 [Thread-8] INFO  c.world2xplane.Rules.GeneratorStore - Reading Regions
19:14:23.422 [Thread-8] INFO  com.world2xplane.Parser.OSMProcess - Output Folder: /Users/tony/Projects/xplane-scenery-tool/./test/austria-lat
19:14:23.422 [Thread-8] INFO  com.world2xplane.Parser.OSMProcess - Processing /Users/tony/Projects/xplane-scenery-tool/./test/austria-latest.
19:14:23.423 [Thread-8] INFO  c.world2xplane.Rules.GeneratorStore - Reading Poly File /Users/tony/Projects/xplane-scenery-tool/./test/austria.pc
19:14:23.433 [Thread-8] INFO  com.world2xplane.Parser.OSMProcess - First Pass (Gathering Relations and Way Nodes)
19:14:55.979 [Thread-8] INFO  com.world2xplane.Parser.OSMProcess - First Pass Complete, stored 38335 relations, 29237147 nodes
19:14:55.980 [Thread-8] INFO  com.world2xplane.Parser.OSMProcess - Second Pass (Gathering Ways for Relations)
19:15:09.868 [Thread-8] INFO  com.world2xplane.Parser.OSMProcess - Second Pass Complete, stored 29237147 nodes
19:15:09.869 [Thread-8] INFO  com.world2xplane.Parser.OSMProcess - Third Pass (Gathering Nodes for Relations)
19:15:24.838 [Thread-8] INFO  com.world2xplane.Parser.OSMProcess - Third Pass Complete, stored 31518703 nodes
19:15:24.838 [Thread-8] INFO  com.world2xplane.Parser.OSMProcess - Fourth Pass(Gathering Coordinates for Nodes)
19:15:44.995 [Thread-8] INFO  com.world2xplane.Parser.OSMProcess - Fourth Pass Complete
19:15:44.996 [Thread-8] INFO  com.world2xplane.Parser.OSMProcess - Fifth Pass(Sorting Relations into tiles)
```

Used/Available Memory: 2590/12271MB 21%    0%

The log tab contains a running log of generation and should be monitored for errors and progress reports. Also, if you wish to report any bugs or issues, please copy and paste the content of this window and send it along with your bug report.

## The configuration File

The configuration file consists of a list of rules, which are evaluated in priority order, i.e. If multiple rules match an object in OSM, the rule which is highest on the list will be evaluated. This file is called *config.xml* and is available in the resources folder.

A rule consists of a rule type, a list of filters and some specific rule options. A rule can be specified as *pass-through*, which means that once it has evaluated, it won't sink the object and pass it down to the next matching rule.

### Default Configuration Files

World2XPlane comes with 2 default config files, the main differences are shown below:

#### config.xml

This is the default configuration file, as contains recommended settings and shouls be enough for most users.

#### config-hd.xml

This config file is a heavy-weight config file and includes lots of extras not set in the default, such as extra street-lighting, fire-hydrants, more trees in residential areas and fences. This file will produce nicer scenery, but can be quite intensive on slower machines, and should be used only if you have the hardware to run it.

### Global Config Options

The following options in the configuration file effect the scenery generation as a whole:

| Option | Meaning/Values | Example |
|---|---|---|
| facadeSet | A facade set to use for generating facades. This currently uses OSM2XP facade sets, so point it at the set descriptor file.<br>You can use multiple facade sets and reference them in different rules by using an identifier, as shown in the example | <facadeSet identifier="simheaven_red"><br>./resources/facades/<br>osm2xpFacadeSetDescriptor.xml<br></facadeSet> |
| region-enabled | Whether to enable regions. With this option enabled, the generator will pass the country code by default to object rules. When disabled, regional rules are ignored | true |

| | | |
|---|---|---|
| world-models-folder | Location of world models folder. Only relevant when using the command-line. | |
| smart-exclusions | Whether to enable smart exclusions. Smart exclusions only apply X-Plane exclusions around areas where there are objects and forests. Where there is no scenery, any scenery below will come through. | true/false<br>true/false |
| building-exclusion-range | Division in metres to use for calculating exclusion zones for buildins. Using a low value will improve accuracy, but greatly effect generation time. A value of 200 is recommended | 200 |
| buildings-per-grid | Number of buildings per grid (In the building-exclusion-range) to use to determine if an exclusion zone is needed. E.g. A value of 10 will indicate that at least 10 houses have to be clumped together in the exclusion range before an exclusion is created | 10 |
| clip-airports | Clip airports out of forests to prevent trees on the runway | true/false |
| clip-roads | Clip forests and fields with roads passing through them or along the perimeter to prevent trees on the roads | true/false |
| forest-exclusion-range | Division in metres to use for calculating exclusion zones for forests. Using a low value will improve accuracy, but greatly effect generation time. A value of 1000 is recommended | 1000 |
| obstacle-lights | With this option enabled, buildings taller than 50m will have an obstacle light placed on the top of them. Buildings over 100m will have a flashing light. | true/false<br>true/false |
| generate-roof-wall-colors | With this option enabled, the system will try and generate a texture for 3D objects inside OSM, i.e. Those with building:colour, roof:colour, building:material and roof:material tags.<br>**DISABLE THIS IF X-PLANE RANDOMALLY CRASHES (there is a known problem with heights)** | true/false |
| enable-osm-3d | Create objects instead of facades for objects with 3D information inside OSM. | true/false |
| compress-textures | Will attempt to compress textures used from the facade and world library into DDS. If a DDS texture is available, X-Plane will use this instead of a PNG, which decreases memory usage and improves loading time. | true/false |

| | | |
|---|---|---|
| | Although this option is slow, it’s recommended you use it. | |
| exclude-objects | Whether to exclude objects from the scenery below | true/false |
| exclude-forests | Whether to exclude forests from the scenery below | true/false |
| exclude-facades | Whether to exclude facades from the scenery below | true/false |
| exclude-network | Whether to exclude roads/rail and power lines from the scenery below | true/false |
| exclude-beach | Whether to exclude X-Plane’s default beaches | true/false |
| | | |
| create-road-network | Generate roads from OSM data | true/false |
| rules | The list of rules to use. See the next section on rules below | See rules section |
| regions | A list of region providers. | See regions section |
| temporary-file-path | Location to store temporary files on the disk. Recommend to set this to a fast SSD drive. | ./tmp |

## Rule Types

The following types of rules are supported:

### Forest Rule

A forest rule is used to attach X-Plane forest .for files around a closed way in OSM.

### Object Rule

An object rule is used to insert an X-Plane .obj file either at a single node's position, or when used in combination with object footprint information, will attempt to find a matching object in OSM, and rotate the object to fit inside. Some examples of object rules are:

1. Placing a tree whenever a node is found with natural=tree
2. Placing a smoke chimney whenever landmark=chimney is found
3. Placing a house that is 10m by 10m, into a matching way in OSM.
4. Placing a gas tank object into any object in OSM that is round and has a specific radius.

### Facade Rule

A facade rule will attach an X-Plane facade file around a closed way in OSM. Facade rules also support custom texture generation using 3D information from OSM. If a building has height, colour, material or roof information inside OSM, a custom texture will be created for that object.

Additionally, obstacle lights will be placed on top of buildings that are over 50 meters in height, so that they can be seen in the dark.

Some examples of facade rules are:

1. Wrapping a building=house closed way with a facade
2. Placing a facade fence around a closed way with the tag barrier=fence
3. Creating a custom texture for an object in OSM which has the following tags: *building=yes,building:colour=green,building:material=glass,building:levels=3*

### Line Rule

A line rule will place objects at a density specified along a way, e.g. Placing trees along the perimeter of a field. N.B. The way doesn't need to be closed.

### Random Rule

A random rule will randomly place X-Plane .obj files inside a closed OSM way, e.g. Filling a field with random cattle, or placing cranes on a building site.

### Sink Rule
A sink rule will sink a particular OSM object, basically excluding/ignoring it. E.g. You could add a sink rule to sink all objects with an area less than 3m$^2$.

### Area Tracker Rule
An area tracker rule will record the location of a closed OSM way using the rules specified. You can then refer to the area tracker rule in other rules as a filter, e.g. You could track all landuse=residential areas, and then make sure only houses of a particular type appear in residential areas. *N.B. Area tracker rules will greatly slow down the speed of generation.*

### Way Tracker Rule
A way tracker rule will record which way a node is a member of. This can be used to rotate objects based on their parent way. E.g. You can create a way tracker to track cable car lines, and then use it to place the cable car towers rotated correctly.

### 3D Object Generation
World2XPlane can be configured to automatically generate a 3D object for a building, and texture/colour it using 3D data from OSM. The buildings will look better than using facades, as the objects can be offset from the ground and have complex shapes, such as domed/gabled roofs. Generally, 3D object generation is used with building:part=* tags inside OSM.

Each rule will be covered in greater detail later on in this manual.

## Regions

World2XPlane can be used to generate regional scenery, e.g. Place different buildings or trees based on the country they are in. Regions should not be confused with areas (which are taken from OSM and used to identify residential and industrial zones).

In order to identify a region, World2XPlane uses a region-provider. The region providers are user configurable and extendable. At present, three types of region providers are support:

### ESRI Shapefiles

By far the most common way to describe a region is using a Shapefile (http://wiki.openstreetmap.org/wiki/Shapefiles). Many of which are available for free on the Internet. By default, World2XPlane uses a Shapefile from www.naturalearthdata.com to identify countries. However, if needed additional Shapefiles can be placed into the application, e.g. To split up a country into counties, etc.

In order to use a Shapefile, an identifier is needed for the region. e.g. This could be the standard ISO country code. This value is used in rules using the region. In order to find out the name of this identifier, you can either consult the documentation from the Shapefile provider, or open the Shapefile up in an external application and find the variable name.

*Using too many Shapefiles, or high resolution complex shapes can slow down generation, so try and use the lowest resolution shapefile possible.*

### CSV Files

World2XPlane can also read a basic CSV file which contains rectangular regions. This is the quickest and easiest way to quickly add a new region, and also is less demanding on resources as tests on rectangles are much quicker than polygons.

The format for the CSV file is shown below:

```
north,west,south,east,identifier
81.00,-31.26,27.63,39.969,CE
```

All coordinates are specified in WGS84 decimal degrees, and identify a rectangular area.

### Plugins

Developers can write their own region identifier and plug it into the generator pipeline. The plugin must implement three simple methods. *Work in Progress (Plugins aren't enabled yet)*

## Configuration

Region providers are configured via the regions area in the configuration file, some examples are given below:

```
<regions>
   <region type="shapefile">
       <identifier>iso_a2</identifier>
       <file>ne_50m_admin_0_countries.shp</file>
   </region>
   <region type="csv">
       <file>./resources/regions.csv</file>
   </region>
</regions>
```

This option defines two regional providers, one using a Shapefile, and the other a CSV file. The shapefile uses the variable **iso_a2** to identify the region.

The configuration options are detailed below:

| Option | Meaning/Values | Example/Values |
|---|---|---|
| type | Defines the type of region identifier | shapefile<br>csv |
| identifier | When using shapefiles, this is variable name used to identify the name of the region. | iso_a2 |
| file | When using shapefile or CSV regions, specify the path to the file here. | ./resources/regions.csv |

## Heights and Adaptive Heights

Most buildings inside OSM outside of cities don't have height information associated with them, therefore choosing a suitable random height for buildings can be difficult. An adaptive height rule can be created to use an area centred around a node to create random heights.

This example adaptive-height option will use the village, town and city node-trackers (which are tagged as place=city, place=town, and place=village) inside OSM, and create a height rule around them. In a village, buildings within 2km of the centre will be either 1 or 2 storey buildings, in a town between 1 and 3 levels and in a city anywhere from 2 to 6.

```
<adaptive-height identifier="city_heights">
    <node identifier="village" min_levels="1" max_levels="2" range="2" />
    <node identifier="town" min_levels="1" max_levels="3" range="2" />
    <node identifier="city" min_levels="2" max_levels="6" range="2" />
</adaptive-height>
```

This adapative-height can then be used inside facade and object rules to choose the height of the building. N.B. If no nearby town, village or city is found, then you should also provide a fallback height, as shown in the examples below:

```
<rule type="object">
   <filter type="key-value">
             building=residential
             building=house
             building=terrace
             building=apartments
             building=detached
             building=garage
             building=yes,area=residential
             building=yes,area=none
    </filter>
    <circular>false</circular>
    <tolerance>1.5</tolerance>
    <restrict-height>true</restrict-height>
    <default-height>city_heights</default-height>
    <min-random-height>3</min-random-height>
    <max-random-height>6</max-random-height>
    <best-fit>./resources/residential.csv</best-fit>
 </rule>

<rule type="facade">
    <filter type="key-value">
        building=*,area=residential
    </filter>
    <simple>true</simple>
    <default-height>city_heights</default-height>
    <min-random-height>4</min-random-height>
    <max-random-height>9</max-random-height>
    <max-area>15</max-area>
    <area-type>sloped-residential</area-type>
    <simplify>0.5</simplify>
</rule>
```

The configuration options for nodes are shown below

| Option | Meaning/Values | Example/Values |
|---|---|---|
| identifier | The identifier of the node tracker | city |
| min_levels | Minimum random range of building levels (stories) | 1 |
| max_levels | Maximum random range of building levels (stories) | 3 |
| range | Range in KM from the node, e.g. 4km around the city centre | 4 |

## Filters

A filter is passed the list of tags from the OSM object, and if the conditions in the filter match, the filter returns true and the rule is valid.

A filter rule can be a basic rule consisting of one simple filter, or many filters together to form a complex set of conditions.

Possible filter values can be:

| | | |
|---|---|---|
| *tag=value* | Validates a tag = a value.<br>Accepts wildcards | e.g.<br>**building=residential**<br>**building=*** |
| *tag!=value* | Validates a tag is not equal to a value<br>Accepts wildcards | e.g.<br>**building!=industrial** |
| *tag<value* | Validates a tag's numeric value is less than the specified value. | e.g.<br>**height<30** |
| *tag>value* | Validates a tag's numeric value is greater than the specified value. | e.g.<br>**height>30** |
| !tag | Validates a tag is not present | e.g.<br>**!building:color** will validate that the tag *building:color* does not exist |
| *area=identifier* | Validates that a particular object is in an area specified by an area-tracker rule.<br>For any area, use the identifier value of **none** | e.g.<br>**area=residential**<br>Will evaluate true if the object is in an area tracked by the residential area tracker |
| *node=identifier* | Validates that the item has a specified node inside it from a node-tracker rule, e.g. node=place_of_worship will evaluate to true if the node is inside the specified object. | **node=place_of_worship, node=fast_food** |
| *tag~value* | Tag contains the string value | e.g.<br>**name~lidl**<br>Will evaluate true if the name contains the word lidl |
| *tag!~value* | Tag doesn't contain the string value | e.g.<br>**note!~light**<br>will evaluate true if the note doesn't contain the word light |

### Example 1:

```
<filter type="key-value">
    building=house
</filter>
```

This filter will match all tags from OSM where building=house.

### Example 2:

```
<filter type="key-value">
    building=*
</filter>
```

This filter will match all tags from OSM which have a tag of building, e.g. building=house, building=yes or building=residential.

## And Logic Conditions

To evaluate more than one set of conditions using **AND**, join the rules together separated by a comma on the same line, e.g.

### Example 3:

```
<filter type="key-value">
    building=supermarket,name=lidl
</filter>
```

This filter will match all objects from OSM where *building=supermarket* AND *name=lidl*.

### Example 4:

```
<filter type="key-value">
    building=*,height>10,height<30,building:colour!=red
</filter>
```

This filter will match all buildings (building=*) from OSM which have a height between 10 and 30 meters, and don't have the *building:colour=red* tag

## Or Logic Conditions

To evaluate more than one set of conditions using OR, separate the rules by a new line, e.g.

### Example 4:

```
<filter type="key-value">
    natural=wood
    landuse=forest
</filter>
```

This filter will match either natural=wood, or landuse=forest rules

### Example 5:

```
<filter type="key-value">
   building=*,area=residential,!height
   building=house
   building=residential
</filter>
```

This filter will match either
1) All buildings in residential areas (see area-trackers), which don’t have any height information.
2) Anything matching *building=house*
3) Anything matching *building=residential*

### Example 6:

```
<filter type="key-value">
   man_made=chimney,height>200,height<225
   landmark=chimney,height>200,height<225
</filter>
```

This filter will match all objects with *man_made=chimney* or *landmark=chimney*, and a height between 200 and 225 meters.

### Example 7:

```
<filter type="key-value">
   amenity=place_of_worship
   building=*,node=church
</filter>
```

This filter will match all objects which are either a amenity=place_of_worship, or any buildings which have a node “church” inside them from the node-tracker “church”.

## Rules

A rule consists of two parts, a filter and an operation. If the conditions inside the filter are met, the rule will run against the OSM object. A few important notes about rules are shown below:

- The first rule that accepts the object will sink it (unless pass-through is set). This means that the order of the rules in the configuration file is very important.
- The more rules added, the slower the generation will be. If you don't need a particular rule, consider commenting it out (or using regions).
- Certain rules can only accept specific types of objects. E.g. A forest or facade rule can only accept a closed way. An object rule can accept both ways and individual nodes

Each rule is covered in detail in the following sections. However, there are few settings which are common for all rules

### Regions

Rules can be configured to be regional. For example, a rule can be used to change the trees in Spain to more regional variations, or change an object based on the region it is inside.

To use a region in a rule, use the <regions> option, as shown below:

```
<regions>
   <region>ES</region>
   <region>IT</region>
</regions>
```

This will limit the rule to Spain and Italy only.

It's also possible to tell a rule not to apply to a particular region by placing ! before the identifier, e.g.

```
<regions>
   <region>!ES</region>
</regions>
```
This will prevent the rule from being used in Spain.

## Object Rules

An object rule will place an X-Plane .obj file at either the position of the node, or at the centroid of a closed way.

An object rule can also take a simple width/length footprint, and if it finds a matching closed way inside OSM, it will rotate the object to fit inside the footprint. This can be used to place house models into OSM buildings which match in size.

### Placing Objects using a footprint

```
<rule type="object">
   <filter type="key-value">
      building=residential
      building=house
      building=terrace
      building=*,area=residential
   </filter>
   <min-area>4.96x9.82</min-area>
   <max-area>4.96x9.82</max-area>
   <area>UK</area>
   <objects>
       <object>objects/UK_1.obj</object>
       <object>objects/UK_2.obj</object>
   </objects>
</rule>
```

This rule will look for any residential buildings inside OSM which have a footprint of **4.96x9.82** and insert one of the objects specified.

### Placing Objects using best fit

If you have a large list of objects, e.g. Residential house objects, it's much better to use the best fit object rule. The best fit object rule will take a CSV list of buildings, and if it finds an object in OSM which matches the rules, it will look through the CSV file for the tightest fitting object. If it finds more than one object with the same dimensions, it will return one at random.

Since most rules are evaluated on a first found basis, best fit rules allow a large list of objects to be evaluated in a single rule. This not only speeds up the parser, but also it is easier to maintain and produces more varied scenery.

```
<rule type="object">
   <filter type="key-value">
      building=residential
      building=house
      building=terrace
      building=*,area=residential
   </filter>
   <best-fit>./residential.csv</best-fit>
</rule>
```

This rule will match residential buildings inside OSM, and find the best fitting object inside the passed in CSV file. If no matching object is found, the filter will not evaluate and the next rule will be evaluated.

```
<rule type="object">
   <filter type="key-value">
      man_made=storage_tank
      building=storage_tank
   </filter>
   <circular>true</circular>
   <min-radius>5</min-radius>
   <max-radius>7</max-radius>
   <objects>
     <object>objects/storage_tank.obj<object>
   </objects>
</rule>
```

This rule will find all objects tagged with either **man_made=storage_tank** or **building=storage_tank**, that are **circular** and have a radius between **5** and **7** metres. When found, it will place the object **objects/storage_tank.obj** at the centroid of the closed way, or at the node position.

```
<rule type="object">
    <filter type="key-value"
           aerialway=pylon
    </filter>
    <angle-from-way base="90">chair-lift</angle-from-way>
    <objects>
      <object>objects/obstacles/Pylon-Lift.obj</object>
    </objects>
 </rule>
```

This rule will find all pylons used in ski-lifts/gondolas and will rotate it along its parent way using the way tracker **chair-lift.** The base parameter specifies that the object will firstly be rotated 90 degrees.

| Option | Meaning/Values | Example |
|---|---|---|
| min-area | Minimum area of polygon footprint to use in m/m. | 10x10, 20x20 |
| max-area | Maximum area of polygon footprint to use in m/m | 10x10 20x0 |
| closed-override | When set, the rule will be applied to OSM ways as well as nodes, even if the footprint doesn't match. | yes no |
| circular | Only applies the OSM way is circular. | yes no |
| min-radius | For circular rules, the minimum radius in metres to accept | 5 |
| max-radius | For circular rules, the maximum radius in metres to accept | 10 |
| best-fit | The name of a CSV file containing a list of objects to use. The rule will, if the tags match, will use best-fit to find a matching model. If a model is found, then the rule evaluates, if no match is found, the object is passed on. | ./residential.csv |
| restrict-height | When enabled, the min-height, max-height values in the CSV are respected. | true false |
| objects | Contains a list of x-plane object files to use (can't be used with best-fit) | <object>objects/house.obj</object> |
| angle | The angle to place the object at. For footprints this value is ignored. If absent, a random angle is used. | 30 |
| pass-through | Whether the rule sinks the object or passes it on to the next rule. Can be used to chain rules together. | true false |
| tolerance | When using best-fit, the tolerance specifies how much bigger or smaller the object can be in metres. This can drastically increase or decrease the number of objects used, but also effects accuracy, since a higher tolerance will mean buildings maybe too big to fit into the place and overflow. | 1 |
| default-height | The default height limit to use for objects which have no height information when using a best-fit rule. Lower this in areas where there are no high rises, or increase it in cities for taller buildings. This option can also be given an adaptive height identifier | 9 city |
| min-random-height | In the absence of height information, the minimum height in metres to use for random generation | 3 |
| max-random-height | In the absence of height information, the maximum height in metres to use for random generation. | 6 |
| street-facing | Whether to try and choose a building which is facing the nearest street (Using the X axis of the object). | true/false |

| | | |
|---|---|---|
| | Enabling this will produce nicer looking scenery, but will really slow down generation | |
| reject-multi-polygon | Whether to reject items which are part of multi-polygons | true/false |
| | | |
| closed | Whether the rule should only apply to closed ways. If set to true, only closed ways (such as buildings) will validate for this rule, if set to false only unclosed ways (such as roads as piers) will validate. | true/false |
| required-level | The minimum level to always show objects/facades of this rule, e.g. Setting to 6 will mean the object is guaranteed only to be visible on the highest density setting, setting to 1 will mean it will always be visible. | (0 to 6), e.g. 1 to always show up the lowest density setting |
| min-height | The minimum height for this rule to apply to. If the height is below the minimum height, the rule will be skipped | 3 (for 3 meters) |
| max-height | The maximum height for this rule to apply to. If the height is above the maximum height, the rule will be skipped | 20 (for 20 meters) |
| angle-from-road | Try and get the angle of the object from the nearest road, e.g. Make a church or bus-stop face the nearest road | true/false |

## Best fit CSV File

The best fit CSV file contains a list of X-Plane objects and basic information about them. The format of the file is shown below:

```
width,length,type,region,model,height-min,height-max,stories
10.0,10.0,residential,DE;NL,german_house.obj,0,13,2
```

| Column | Meaning/Values | Example |
|---|---|---|
| width | The width of the building in metres | 10.0 |
| length | The length of the building in metres | 10.0 |
| type | The type or description for the object (Not used by the generator) | House in a village |
| region | A list of region identifiers that applies to this object. The object will only be placed if it fulls within the list of regions | **DE** – For Germany<br>**DE;NL;UK** For Germany, Holland and the UK |
| height-min | The minimum height restriction of the building in metres. If the height of the object is below this value, then the building won't be placed | 0 |
| height-max | The maximum height restriction of the building in metres. If the height of the object is avoe this value, then the building won't be placed. | 10 |

| | | |
|---|---|---|
| best-fit | The name of a CSV file containing a list of objects to use. The rule will, if the tags match, will use best-fit to find a matching model. If a model is found, then the rule evaluates, if no match is found, the object is passed on. | ./residential.csv |
| stories | When enabled, the min-height, max-height values in the CSV are respected. | true<br>false |
| objects | Contains a list of x-plane object files to use (can't be used with best-fit) | <object>objects/house.obj</object> |
| angle | The angle to place the object at.<br>For footprints this value is ignored.<br>If absent, a random angle is used. | 30 |
| stories | The number of stories the building has (for reference only) | 2 |

*Regions and height restrictions are only applied if*
*a) region tracking is enabled in the config file i.e. region-enabled is true*
*b) restrict-height is enabled in the rule.*

## Random Rules

A random rule will place random X-Plane object files inside a closed way, e.g. Placing trees in a residential area or cows in a field.

Random rules can also place forests using <forest> tags if the area is empty. This can improve performance when placing trees, as collision detection is not needed and forests perform better then individual items.

A random rule can either use a **density** value, which will place the requested number of objects per meter, or a **count** value, which will fill the closed way with the requested number of objects

### Placing Random Objects

```
<rule type="random">
   <filter type="key-value">
      landuse=farmland
   </filter>
   <density>5</density>
   <objects>
       <object>objects/cow.obj</object>
       <object>objects/sheep.obj</object>
   </objects>
</rule>
```

This rule will look for farmland landuse tags and place random cattle at a density of 1 every 5 metres.

```
<rule type="random">
   <filter type="key-value">
      landuse=residential
   </filter>
   <density>2</density>
   <area-densities>
     <area-density min="0" max="5000">2.0</area-density>
     <area-density min="5000" max="6000">2.5</area-density>
     <area-density min="6000" max="7000">3.0</area-density>
     <area-density min="7000" max="9000">7.0</area-density>
     <area-density min="9000" max="10000">12.0</area-density>
     <area-density min="10000">25.0</area-density>
   </area-densities>
   <forests-on-empty>true</forests-on-empty>
   <collision-test>true</collision-test>
   <pass-through>true</pass-through>
   <objects>
       <object>tree1.obj</object>
       <object>tree2.obj</object>
   </objects>
```

```
    <forests>
        <forest>for1.for</forest>
        <forest>for2.for</forest>
    </forests>

</rule>
```

This rule will match all residential areas, and place random trees at a density specified in the area-density tags based on the area in m2. It will also do a collision test to make sure the tree won't be inside a building, and will also pass the rule through so another rule can evaluate it. If the area is empty, it will instead place a forest from one of the forest tags

| Option | Meaning/Values | Example |
|---|---|---|
| density | How many items to place per meter | 2 |
| count | How many items to place in the area (Instead of using density) | 3 (to place 3 objects) |
| angle | The angle to place the objects at. If not set, a random angle is used. If value is "determine", the angle of the exterior way is used, i.e. e.g. Lining up cars in a car park based on the car park's longest wall. | 5<br>determine |
| collision-test | Will ensure the object isn't placed inside a building (Warning, this slows down generation) | true |
| circular | Only applies if the OSM way is circular. | yes<br>no |
| min-radius | For circular rules, the minimum radius in metres to accept | 5 |
| max-radius | For circular rules, the maximum radius in metres to accept | 10 |
| objects | Contains a list of x-plane object files to use. If more than one item, one will be chosen at random. | <object>objects/house.obj</object> |
| pass-through | Whether the rule sinks the object or passes it on to the next rule. Can be used to chain rules together. | true<br>false |
| min-area | Mininum area in metres squared for the random rule. | 2000 |
| forests-on-empty | Will use forests if the area is empty for performance | true/false |
| max-area | Maximum area in metres squared for the random rule | 5000 |

| | | |
|---|---|---|
| area-density | Changes the density based on the area of the way in m2. Takes a min/max value, or just a min. See example above | <area-density min="0" max="5000">2.0</area-density> |
| buffer | Make an object thicker/thinner by specified metres. Can also be used to convert a line into a polygon | 1.5 (for 1.5 meters)<br>-1.0 (Shrink object by 1.0 metre) |
| closed | Whether the rule should only apply to closed ways. If set to true, only closed ways (such as buildings) will validate for this rule, if set to false only unclosed ways (such as roads as piers) will validate. | true/false |
| required-level | The minimum level to always show objects/facades of this rule, e.g. Setting to 6 will mean the object is guaranteed only to be visible on the highest density setting, setting to 1 will mean it will always be visible. | (0 to 6), e.g. 1 to always show up the lowest density setting |
| min-height | The minimum height for this rule to apply to. If the height is below the minimum height, the rule will be skipped | 3 (for 3 meters) |
| max-height | The maximum height for this rule to apply to. If the height is above the maximum height, the rule will be skipped | 20 (for 20 meters) |

## Lines Rules

A line rule will place X-Plane object files along a way, e.g. Placing trees along a river bank or ditch, or lining a road with houses

### Placing Objects

#### Example 1

```
<rule type="line">
   <filter type="key-value">
      waterway=ditch
   </filter>
   <min-density>10</min-density>
   <max-density>11</max-density>
   <objects>
       <object>objects/tree1.obj</object>
       <object>objects/tree2.obj</object>
   </objects>
</rule>
```

This rule will look for all waterway=ditch ways and will place random trees along the line at a random density of one every 10 to 11 metres.

#### Example 2

```
<rule type="line">
   <filter type="key-value">
        highway=primary,area=residential
        highway=secondary,area=residential
        highway=residential
        highway=tertiary,area=residential
    </filter>
    <min-density>40</min-density>
    <max-density>40</max-density>
    <offset>4</offset>
    <pass-through>true</pass-through>
    <offset-angles>90;270</offset-angles>
    <objects>
         <object>light.obj</object>
    </objects>
</rule>
```

This rule will find all residential roads, and all primary, secondary and tertiary roads inside residential zones (using the residential area-tracker). It will place street lights (light.obj) along the at a spacing of 40 metres. The street lights will be offset from the road at angles 90, and 270 (so one will be one side of the street, the second on the other side). The objects will be offset from the centre of the road by 4 metres

## Example 3

```
<rule type="line">
   <filter type="key-value">
        highway=primary,area=residential
        highway=residential
    </filter>
    <offset>14</offset>
    <offset-angles>90;270</offset-angles>
    <pass-through>true</pass-through>
   <spacing-division>true</spacing-division>
    <objects>
       <object group=”1” spacing=”1” width=”4”>house1.obj</object>
       <object group=”1” spacing=”1” width=”5”>house2.obj</object>

       <object group=”2” spacing=”0” width=”4”>house3.obj</object>
       <object group=”2” spacing=”0” width=”4”>house4.obj</object>

    </objects>
</rule>
```

This rule will find all residential roads, and all primary roads inside residential zones (using the residential area-tracker). It will place random houses along the road at an offset of 14 metres from the road centre line. It will place houses on both sides of the road, i.e 90 degrees and 270 degrees.  The houses will be placed based on group, e.g. One road would have only group 1 houses, another would have group 2 houses. The spacing and width for each object determines how much space is needed for each house (This is used when no density is given). This rule could be used to place autogen houses in areas which don’t have much data in OSM.

The spacing-division option will divide the distance travelled along the line by the number of offset-angles. E.g. If you were placing houses along both sides of a road, this would ensure that as each house is placed, each offset angle would keep it’s own running distance instead of stepping (As in the case of placing street lights).

| Option | Meaning/Values | Example |
|---|---|---|
| min-density | Minimum random value for placing objects in metres.<br>Used in combination with max-density to add variety to the object placement. | 5 (for every 5 metres) |
| max-density | Maximum random value for placing objects in metres.<br>Used in combination with min-density to add variety to the object placement | 6 (for every 6 metres). |
| circular | Only applies if the OSM way is circular. | yes<br>no |
| min-radius | For circular rules, the minimum radius in metres to accept | 5 |
| max-radius | For circular rules, the maximum radius in metres to accept | 10 |
| objects | Contains a list of x-plane object files to use. If more than one item, one will be chosen at random. For additional options see the objects table below | <object>objects/house.obj</object> |
| pass-through | Whether the rule sinks the object or passes it on to the next rule. Can be used to chain rules together. | true<br>false |
| offset | A value in metres to offset the item from the line. Must be used in conjunction with offset-angles | 4 |
| offset-angles | A list of angles separated by a semi-colon. Each angle is used to offset the object and is used in conjunction with the offset options | 90 (Offset 90 degrees)<br>90;270 (Offset 90, then 270, then 90…) |
| buffer | Make an object thicker/thinner by specified metres. Can also be used to convert a line into a polygon | 1.5 (for 1.5 meters)<br>-1.0 (Shrink object by 1.0 metre) |
| closed | Whether the rule should only apply to closed ways. If set to true, only closed ways (such as buildings) will validate for this rule, if set to false only unclosed ways (such as roads as piers) will validate. | true/false |
| required-level | The minimum level to always show objects/facades of this rule, e.g. Setting to 6 will mean the object is guaranteed only to be visible on the highest density setting, setting to 1 will mean it will always be visible. | (0 to 6), e.g. 1 to always show up the lowest density setting |
| min-height | The minimum height for this rule to apply to. If the height is below the minimum height, the rule will be skipped | 3 (for 3 meters) |
| max-height | The maximum height for this rule to apply to. If the height is above the maximum height, the rule will be skipped | 20 (for 20 meters) |
| collision-test | Whether to check for collisions with buildings and roads. | true/false |
| track | Whether to track the placed object as a building for collision detection. | true/false |

| | | |
|---|---|---|
| count | if set, restrict the number of items randomally placed along the line by the set number, e.g. Place a single tractor somewhere along a farmtrack | 1 |
| spacing-division | Divide the distance travelled along the line by the number of offset angles. Useful if you want to equally line both sides of a road instead of stepped placement. | true/false |
| clip-to-area | If used with an area tracker in the filter, the line will be clipped so it fits inside the area. This can be used to place caravans along the roads only inside a caravan park, or street lights only inside residential zones and not just along the entire road. | true/false |

### Object Options

By default, World2Xplane will chose an object at random from the list of objects, and place the object using the min/max density options. However, if you would like more control over the objects and their spacing, then you can specify the width and spacing required for each object (See example 3 above). Additionally, objects can be grouped into groups. If grouping is used, then a group will be chosen at random for each line, and then only objects from that group will be placed.

| Option | Meaning/Values | Example |
|---|---|---|
| width | The width along the line the object would take. This can be used instead of the min and max density options | 5 (for 5 metres) |
| spacing | Additional spacing needed when placing the object in metres | 1 (for 1 meter) |
| group | An integer group ID for grouping similar objects, e.g. Similar houses. | 1 |

## Sink Rules

A sink rule will swallow a particular object, and won't place anything inside X-Plane. Sink rules can used to exclude objects from generation, such as small buildings or airport buildings.

### Sinking Objects

```
<rule type="sink">
   <filter type="key-value">
      building=*
   </filter>
   <max-area>3</max-area>
</rule>
```

This rule will look for any buildings which have an area under 3m$^2$. If found, the rule will sink the object.

*It's important to place sink rules at the top of the rules list, as the rules are evaluated in priority order. If the rule is placed after another rule that may accept the object, then the object will not be sinked.*

| Option | Meaning/Values | Example |
|---|---|---|
| min-area | Minimum area in metres squared for the object | 3 for 3m$^2$ |
| max-area | Maximum area in metres squared for the object | 1000m$^2$ |
| circular | Only applies if the OSM way is | yes |

|  |  |  |
|---|---|---|
|  | circular. | no |
| min-radius | For circular rules, the minimum radius in metres to accept | 5 |
| max-radius | For circular rules, the maximum radius in metres to accept | 10 |
| closed | Whether the rule should only apply to closed ways. If set to true, only closed ways (such as buildings) will validate for this rule, if set to false only unclosed ways (such as roads as piers) will validate. | true/false |
| min-height | The minimum height for this rule to apply to. If the height is below the minimum height, the rule will be skipped | 3 (for 3 meters) |
| max-height | The maximum height for this rule to apply to. If the height is above the maximum height, the rule will be skipped | 20 (for 20 meters) |

## Facade Rules

A facade rule is used to wrap an X-Plane facade around a closed-way. These can be used to generate buildings or fences. Facade rules can also take a list of facades using OSM2XP facade packages and choose a facade based on a set of rules:

### Placing Facades

```
<rule type="facade">
   <filter type="key-value">
        building=garage
   </filter>
   <min-random-height>2</min-random-height>
   <max-random-height>3</max-random-height>
   <facades>
       <facade>buildingGeneric6.fac</facade>
       <facade>buildingGeneric7.fac</facade>
   </facades>
</rule>
```

This rule will find all building=garage ways inside the OSM file, and wrap one of two facades around it, and in the absence of height information, will choose a random height between 2 and 3 metres.

```
<rule type="facade">
   <filter type="key-value">
                building=house,
                building=residential,
                building=apartments,
                building=terrace,
                building=*,area=residential
   </filter>
   <simple>true</simple>
   <min-random-height>6</min-random-height>
   <max-random-height>20</max-random-height>
   <area-type>sloped-residential</area-type>
</rule>
```

This rule will match any residential buildings which are simple (i.e. Basic squares) and will assign a random sloped-residential facade from the facade set, and in the absence of height information, a random height between 6 and 20 metres.

```
<rule type="facade">
    <filter type="key-value">
          building=*,area=commercial
    </filter>
    <min-random-height>8</min-random-height>
    <max-random-height>20</max-random-height>
    <min-area>18</min-area>
    <area-type>commercial</area-type>
</rule>
```
This rule will match any building inside a commercial area which has a minimum area of 18m$^2$. It will choose a random commercial facade from the facade set, and in the absence of any height information, assign a random height between 8 and 20 metres

## Facade Distribution Rules

If you have more than one facade set, you can configure the rule to use a random item from the facade set using a given frequency. e.g. You could create two facade sets, one with red roofs, and one with dark, and set up a rule to show 40% Red 60% black roofs in the UK, and 80% red, 20% black in France.

The below rule shows an example of how to set this up.

```
<rule type="facade">
 <filter type="key-value">
       building=house
       building=residential
       building=apartments
       building=terrace
       building=*,area=residential
  </filter>
  <simple>true</simple>
  <min-random-height>4</min-random-height>
  <max-random-height>9</max-random-height>
  <max-area>15</max-area>
  <area-type>sloped-residential</area-type>
  <regions>
      <region>DE</region>
      <region>NL</region>
      <region>FR</region>
      <region>GB</region>
  </regions>
  <distributions>
     <distribution identifier="red" percentage="40" />
     <distribution identifier="black" percentage="60” />
```

```
    </distributions>
</rule>
```

This rule will assign a random residential sloped house in Germany, the Netherlands, France or the UK. If will use a random house 40% of the time from the red set, and 60% of the time from the black set.

## Options

| Option | Meaning/Values | Example |
|---|---|---|
| min-random-height | In the absence of height information, the minimum height in metres to use for random generation | 3 |
| roof-height | The roof height in metres. This height will be added onto the value inside OSM or from the random value, and is used to when creating facades to add extra height to generate the roof slope. | 3 |
| max-random-height | In the absence of height information, the maximum height in metres to use for random generation. | 15 |
| min-area | The minimum area in metres square. If an object's area is below the minimum area, then rule will not be evaluated | 10 |
| max-area | The maximum area in metres square. If an object's area is above the maximum area, then rule will not be evaluated | 100 |
| circular | Only applies if the OSM way is circular. | yes<br>no |
| min-radius | For circular rules, the minimum radius in metres to accept | 5 |
| max-radius | For circular rules, the maximum radius in metres to accept | 10 |
| facades | Contains a list of x-plane facade files to use. If more than one item is available, one will be chosen at random.<br><br>To use a facade from the facade set, specify the name without a path, e.g. slopedhouse1a.fac | <facade>facades/house.fac</facade> |
| area-type | Use a random facade from the facade set from the specified type group. | residential<br>industrial<br>commercial<br>sloped-residential<br>random |
| pass-through | Whether the rule sinks the object or passes it on to the next rule. Can be used to chain rules together. | true<br>false |
| distribution | How to distribute facade sets when | <distribution |

| | | |
|---|---|---|
| | more than one is available. The option takes the identifier of the facade set and a percentage | `identifier="red" percentage="40" />` |
| buffer | Make an object thicker/thinner by specified metres. Can also be used to convert a line into a polygon | 1.5 (for 1.5 meters)<br>-1.0 (Shrink object by 1.0 metre) |
| default-height | The adaptive height identifier to use when trying to determine a height (See the adaptive height section). This should also be used in combination with min-random-height, max-random-height values | city |
| allow-unclosed | Whether to allow this facade to wrap the object if it isn't closed, i.e. forms a loop. Use this option if you want to wrap fences (or facades with no ring) around OSM ways which aren't closed | true/false |
| closed | Whether the rule should only apply to closed ways. If set to true, only closed ways (such as buildings) will validate for this rule, if set to false only unclosed ways (such as roads as piers) will validate. | true/false |
| skip-last-point | By default, X-Plane doesn't need to draw the final point on a closed facade, e.g. For a building, as it will close the building itself.  This however, isn't desirable for fences or unclosed ways. Set this value to false to show render all points.<br>The default behavior is true. | true/false |
| required-level | The minimum level to always show objects/facades of this rule, e.g. Setting to 6 will mean the object is guaranteed only to be visible on the highest density setting, setting to 1 will mean it will always be visible. | (0 to 6), e.g. 1 to always show up the lowest density setting |
| min-height | The minimum height for this rule to apply to. If the height is below the minimum height, the rule will be skipped | 3 (for 3 meters) |
| max-height | The maximum height for this rule to apply to. If the height is above the maximum height, the rule will be skipped | 20 (for 20 meters) |

N.B. If the object inside OSM contains height, colour or material information, World2XPlane will generate unique facade for it and ignore the facade or area-type area.

## Polygon Rules

A polygon rule is used to wrap an X-Plane polygon around a closed-way. These can be used to cover areas with textures, e.g. beaches or grass.

### Placing Polygons

This rule will find all natural=beach ways inside the OSM file, and place one of two polygon textures on it,.

```
<rule type="polygon">
   <filter type="key-value">
      natural=beach
   </filter>
     <polygons>
          <polygon>polys/beach1.pol</polygon>
          <polygon>polys/beach2.pol</polygon>
     <polygons>
</rule>
```

### Options

| Option | Meaning/Values | Example |
|---|---|---|
| min-area | The minimum area in metres square. If an object's area is below the minimum area, then rule will not be evaluated | 3 |
| max-area | The maximum area in metres square. If an object's area is above the maximum area, then rule will not be evaluated | 3 |
| circular | Only applies if the OSM way is circular. | true/false |
| min-radius | For circular rules, the minimum radius in metres to accept | 10 |
| max-radius | For circular rules, the maximum radius in metres to accept | 100 |
| polygons | Contains a list of x-plane polygon files to use. If more than one item is available, one will be chosen at random. | <polygon>beach.pol</polygon> |
| pass-through | Whether the rule sinks the object or passes it on to the next rule. Can be used to chain rules together. | true/false |
| buffer | Make an object thicker/thinner by specified metres. Can also be used to convert a line into a polygon | 1 |

## Forest Rules

A forest rule will wrap an X-Plane .for forest file around a closed way inside X-Plane. The forest can be configured to either fill the area, or surround the perimeter only (e.g. Surround a field with trees)

### Placing Forests

```
<rule type="forest">
   <filter type="key-value">
      natural=wood
      landuse=forest
   </filter>
   <density>255</density>
   <forests>
       <forest>conifer.for</forest>
       <forest>broad.for</forest>
   </objects>
</rule>
```

This rule will look all natural=wood and landuse=forest tags and place at random either a conifer or broad forest. The density value specifies a density, which can be between 0 and 255. 255 being maximum density.

```
<rule type="forest">
   <filter type="key-value">
      landuse=farmland
   </filter>
   <density>200</density>
   <perimeter-only>true</perimeter-only>
   <pass-through>true</pass-through>
   <forests>
       <forest>shrubs.for</forest>
       <forest>mixed.for</forest>
   </objects>
</rule>
```

This rule will match landuse=farmland areas and place at random either shrubs or mixed trees around the perimeter of the area. This rule would surround farmland fields with small trees. The rule is also marked as pass-through, so once it has finished evaluating, it will pass the shape onto the next rule.

| Option | Meaning/Values | Example/Values |
|---|---|---|
| density | The forest density in the range of 0-255. 255 being maximum density | 255 |
| perimeter-only | When true, the forest will be placed around the perimeter of the area only. | true<br>false |
| collision-test | Will ensure the object isn't placed inside a building (Warning, this slows down generation) | true |
| circular | Only applies if the OSM way is circular. | yes<br>no |
| min-radius | For circular rules, the minimum radius in metres to accept | 5 |
| max-radius | For circular rules, the maximum radius in metres to accept | 10 |
| forests | Contains a list of x-plane .for files to use. If more than one item is given, one will be chosen at random. | <forest>shrub.for</forest> |
| pass-through | Whether the rule sinks the object or passes it on to the next rule. Can be used to chain rules together. | true<br>false |
| buffer | Make an object thicker/thinner by specified metres. Can also be used to convert a line into a polygon | 1.5 (for 1.5 meters)<br>-1.0 (Shrink object by 1.0 metre) |
| closed | Whether the rule should only apply to closed ways. If set to true, only closed ways (such as buildings) will validate for this rule, if set to false only unclosed ways (such as roads as piers) will validate. | true/false |
| required-level | The minimum level to always show objects/facades of this rule, e.g. Setting to 6 will mean the object is guaranteed only to be visible on the highest density setting, setting to 1 will mean it will always be visible. | (0 to 6), e.g. 1 to always show up the lowest density setting |
| min-height | The minimum height for this rule to apply to. If the height is below the minimum height, the rule will be skipped | 3 (for 3 meters) |
| max-height | The maximum height for this rule to apply to. If the height is above the maximum height, the rule will be skipped | 20 (for 20 meters) |

## Area Tracker Rules

An area tracker rule is used to track areas and does not directly place any objects. Area trackers can be used to record residential or industrial areas, and then used in other rules, e.g. To place unknown buildings as houses in residential areas, and unknown buildings as factories in industrial.

Area tracker rules should be used with caution, as they can slow down generation and require more memory.

### Tracking Areas

```
<rule type="area-track">
   <filter type="key-value">
      landuse=residential
   </filter>
   <identifier>residential</identifier>
   <pass-through>yes</pass-through>
</rule>
```

This rule will identify all residential areas inside OSM and store them with the identifier **residential**. The identifier can then be used in other rules using the **area=residential** filter option. The rule is set as pass-through, so that another rule can use **landuse=residential** areas as well, e.g. A random rule to place trees inside residential zones.

```
<rule type="area-track">
   <filter type="key-value">
      aeroway=aerodrome
      aeroway=runway
      aeroway=taxiway
   </filter>
   <identifier>aeroway</identifier>
  <pass-through>yes</pass-through>
</rule>
```

This rule will identify airports inside OSM, and can be used in other rules to either change the building types (e.g. To hangars), or to exclude placing buildings, so that buildings or trees don't appear on runways.

| Option | Meaning/Values | Example/Values |
|---|---|---|
| identifier | The identifier used when the area is found. This identifier is then used by other rules. | residential<br>industrial<br>airport |
| pass-through | Whether the rule sinks the object or passes it on to the next rule. Can be used to chain rules together. | true<br>false |

## Way Tracker Rules

A way tracker rule is used to track ways and does not directly place any objects. Way trackers can be used to determine which way a node is part of, e.g. You may wish to know that a power=tower node is part of a power line, and know the orientation of the tower.

### Tracking Ways

```
<rule type="way-track">
   <filter type="key-value">
      power=line
   </filter>
   <identifier>power_line</identifier>
   <pass-through>yes</pass-through>
</rule>
```

This rule will identify all power lines inside OSM and store them with the identifier **power_line**. The identifier can then be used in objct rules using the **angle-from-way** option. The rule is set as pass-through, so that another rule can use **power=line** ways as well

```
<rule type="object">
   <filter type="key-value">
      power=tower
   </filter>
   <angle-from-way base="90">power_line</angle-from-way>
   <objects>
        <object>objects/obstacles/pylon.obj</object>
   </objects>
</rule>
```

This rule will identify power pylons inside OSM, and will place a pylon each time it finds the tag **power=tower**. It will look at the parent way **power_line** from the way tracker above, and rotate the pylon using the vectors from the parent. It will also add 90 degrees to each rotation.

| Option | Meaning/Values | Example/Values |
|---|---|---|
| identifier | The identifier used when the way is found. This identifier is then used by other rules. | power_line |
| pass-through | Whether the rule sinks the object or passes it on to the next rule. Can be used to chain rules together. | true<br>false |

## Node Tracker Rules

A node tracker rule is used to track nodes and does not directly place any objects. Node trackers can be used to determine if another object contains the node inside it, e.g. Many buildings in OSM are tagged as building=*, but have a node inside them called amenity=place_of_worship

### Tracking Nodes

```
<rule type="node-track">
   <filter type="key-value">
      amenity=place_of_worship
   </filter>
   <identifier>church</identifier>
    <pass-through>yes</pass-through>
</rule>
```

This rule will identify all nodes inside OSM tagged as **amenity=place_of_worship** and record their locations. The identifier can then be used in other rules as a filter, e.g.

```
<rule type="object">
   <filter type="key-value">
      amenity=place_of_worship
       building=*,node=church
   </filter>
   <objects>
        <object>objects/church.obj</object>
   </objects>
</rule>
```

This rule will identify churches inside OSM, and will place a church object each time it finds the tag **amenity=place_of_worship tag, or a building with the tracked node inside it**.

| Option | Meaning/Values | Example/Values |
|---|---|---|
| identifier | The identifier used when the way is found. This identifier is then used by other rules. | church, petrol_station |
| pass-through | Whether the rule sinks the object or passes it on to the next rule. Can be used to chain rules together. | true false |

## Command-Line Options

As well as providing a simple GUI to generate scenery, World2XPlane can also be started from a command-line, and can be used inside scripts to automate the generation of scenery.

To launch the application from the command-line, invoke Java with the path to the World2XPlane.jar file, and provide it with the following commands:

```
java  <java_options> jar World2XPlane <config_file> <input_file>
<output_folder> {<xplane-folder> validate useDatabase resume}
```

The parameters must be in the order specified above. The X-Plane folder, validate, useDatabase and resume parameters are all optional.

| Parameter | Meaning/Values | Example/Values |
|---|---|---|
| 1 | Path to config file | resources/config.xml |
| 2 | Input File | france-latest.osm.pbf |
| 3 | Output Folder | test/france |
| 4 (Optional) | X-Plane Folder | "/Users/Tony/X-Plane 10" |
| 5 (Optional) | validate | Validate models exist in X-Plane |
| 6 (Optional) | useDatabase | Whether to use file based storage for indexes. Use this for generating larger files |
| 7 (Optional) | resume | Whether to resume an existing generation |
| 8 (Optional) | polyfile=?.poly | Path to a poly file to use for the exclusions |

### Examples

1.

```
java -d64 -Xms6g -Xmx8g –jar World2XPlane.jar ./resources/config.xml
./test/france-latest.osm.pbf ./test/france "/Users/tony/Desktop/X-Plane 10"
validate useDatabase resume
```

This example will generate France from a file "./test/france-latest.osm.pbf" and output the resulting scenery to "./test/france". It will use the config file "./resources/config.xml", and the X-Plane directory "/Users/tony/Desktop/X-Plane 10". It will also validate the X-Plane directory, use local storage and resume if the scenery already exists. The processor will run in 64-bit mode, with a minimum of 6GB of Memory, and maximum of 8GB.

2.

```
java -d64 –Xms10g –Xmx12g –jar World2XPlane.jar ./resources/config-
roads.xml ./test/france-latest.osm.pbf ./test/france
polyfile=./test/france.poly
```

This example will generate France from a file "./test/france-latest.osm.pbf" and output the resulting scenery to "./test/france". It will use the config file "./resources/config-roads.xml". The scenery will all be generated in memory, and the objects will not be validated if they exist.  The processor will run in 64-bit mode, with a minimum of 10GB of Memory, and maximum of 12GB. A polyfile france.poly will be used to generate the exclusion zones.