MAPNIK

(0.7.0)

XML Schema Reference

for the

Map Definition File

Table of Contents

XML Schema for the Mapnik Map File

Mapnik is a library of capabilities intended for producing Maps. It takes a diverse set of source inputs and produces (renders) an image.

The rendering of the image requires that you define the 'What' and 'How' through the use of a map definition file. This document describes the Mapnik Map file's contents. Note Mapnik provides other methods to generate maps programatically.

You must use an external program to view or generate the actual images. The Mapnik library comes with a very simple viewer that will cause a Map file to be processed and display the result (its called 'viewer' and is located in the Mapnik 'demo' directory.

NOTE: The size of the image and the geographic area for the image are specified external to the Map file.

Map Element

This element of the Mapnik XML definition file is the root element and defines how a map is to be rendered. All other elements must be contained within it, except for the usual xml block comments.

General Form

```
<Map bgcolor="" buffer_size="" srs="">
</Map>
```

Attribute	Description
`bgcolor`	Background color for the map. Default is: transparent There are four styles allowed for specifying the color: ● `#rrggbb` - a set of hexidecimal values representing the Red(rr), Green(gg) and Blue(bb) color values. ● `rgb(rrr,ggg,bbb)` - a set of Decimal values representing the color values. ● `rgba(rrr,ggg,bbb,aaa)` - a set of Decimal values representing the color values and alpha (transparency). ● color name - a standard alphabetic character string name of a color. eg. `"#000000"` is black, `"rgb(255,255,255)"` is white, `"red"` is the color red, `"rgba(255,255,255,128)"` is white at 50% transparent.
`buffer_size`	Extra space, in pixels, surrounding the map size being rendered. Default is: `"0"` This allows you to have text and symbols rendered correctly when they overlap the image boundary. Specifying a buffer_size allows the text / symbols to be placed correctly, even though the final image that is produced will clip them exactly and the images 'defined' limits. Typically a value of 128 provides good results.
`minimum_version`	Used to ensure that the map is rendered with a version of mapnik which supports the features referenced in the xml files. If the associated map is rendered using a lower version of, then an error is generated and processing is stopped. It is a string in the form of: "x.y.z", where x, y, z are numbered corresponding to the mapnik version. eg. "0.7.0"
`paths_from_xml`	Defines how relative paths in the XML files are processed for the 'file' attribute in datasources and symbolizers. Default is `"true"`. ● true, relative to the xml file containing the Map element ● false, relative to the 'script' that invoked the 'load_map' method

`srs`	Spatial Reference System for the map being rendered. Default: `"+proj=latlong +datum=WGS84"` **Note**: This is EPSG:4326 (it really should be `+proj=longlat +ellps=WGS84 +datum=WGS84 +no_defs`) All sources of geometry used in the maps rendering will be re-projected if they are not in the same SRS as the map.

Children

There are currently only five (5) elements which can be contained within the root Map Element:

- Datasource (template variant) Element
- FileSource Element
- FontSet Element
- Layer Element
- Style Element

Example

```
<Map bgcolor="pink" buffer_size="64" srs="+proj=longlat +ellps=WGS84 +datum=WGS84 +no_defs">
    ...and everything else in between...
</Map
```

See the Appendix for a more detailed example and how all the elements are combined.

From this point onward, the Elements are organized alphabetically, so pay particular attention to the sections titled Children to determine which elements are allowed within a given element. Also, some elements are unique to a given element and they will be described following the parent.

Datasource Element (Template Variant)

When a Datasource element is located directly in the root Map element as a child, it is considered to be a Template which can be subsequently referenced in a Layer's Datasource element. It specifically does not contain any Style element(s) and if present, they will be silently ignored and NEVER used.

This form of the Datasource element contains a set of one or more Parameter elements that may be (as a set) referenced by Name within a Layer Datasource element's base attribute. These elements will be used in addition to those defined in the Layer Datasource element.

General Form

```
<Datasource name="">
    <Parameter name="">value</Parameter>   <-- repeated 0 or more times
</Datasource>
```

Attribute	Description
name	This is a string which uniquely identifies the template. Default is: "Unnamed"

Children

There is currently only one (1) element which can be contained within this variation of the Datasource Element:

- Parameter Element

See Datasource Element for a description of the Parameter Element.

Example

```
<Datasource name=”mySource">
    ...and everything else in between...
</Datasource>
```

See the Appendix for a more detailed example and how all the elements are combined.

Datasource Element

This element is used to define a source of data to be rendered. Currently, data sources include:

- ESRI Shape files
- PostgreSQL data bases
- SQLite data bases - (experimental)
- Oracle 10 data bases – (experimental)
- GDAL raster sources
- Raster Image files
- OGR ??
- OSM – Open Street Map
- Kismet

Mapnik refers to these as: Input Plugins.

General Form

```
<Datasource base="">
  <Parameter name=”″>value</Parameter>   <-- repeated 0 or more times
</Datasource>
```

Attribute	Description
`base`	If this attribute is present, then its' value is the Name of a Datasource Template. The Parameters defined in the template are added to the parameters defined in this element. It is used to extends this datasource's Parameters with parameters that are shared (common) to other Datasource elements.

Children

There is one (1) element, Parameter, that may be contained 1 or more times within the Datasource element. Due to the number of data sources that are supported and the wide variety of parameters that are need to support each type, this Parameter element is generalized and uses the 'name' attribute to distinguish the meaning associated with the value. In addition, there are specific parameters required for each type of data source. The following is are each of the parameters grouped by source type

Parameter – 'gdal' type

This set of parameters are for GDAL – Geospatial Data Abstraction Library supported raster images:

Attribute	Description
`name`	This is a string which uniquely identifies the meaning of the 'value'. It may be one of the following: • type • encoding (optional) • band (optional) • file • shared (optional)

Name	Value Description
`type`	Value is: gdal
`band`	Species the band number to be processed. -1 indicates all bands to be processed.
`encoding`	Specifies the character encoding used in the gdal file for attribute data. If omitted, the default is utf-8.
`file`	Value is the location and name of a GDAL file
`shared`	Specifies if the file is to be opened for shared access or exclusive access and may be one of: • true (shared access) • false (exclusive access) default if omitted.

Parameter – 'kismet' type

This set of parameters are for the Kismet server:

Attribute	Description
`name`	This is a string which uniquely identifies the meaning of the 'value'. It may be one of the following: • type • encoding • estimate_extent (optional) • extent (optional) • host • port

Name	Value Description
`type`	Value is: kismet
`encoding`	Specifies the character encoding used. If omitted, the default is utf-8.
`estimate_extent`	If present, it is used to have mapnik to estimate the extents of the layer and may be one of: • true (estimate the extents) • false (do not estimate extents) default if omitted
`extent`	If present, identifies the layer extents and has the following form: minX,minY,maxX,maxY where each comma separated value is a real number (double) and the units are expressed in terms of degrees or meters depending on the layer's SRS.
`host`	Identifies the network name for the host that contains the data base server. Type is string.
`port`	The port number that the postgreSQL server is accessed on. PostgreSQL listens, by default, on port 5432.

Parameter – 'occi' type

This set of parameters are for the Oracle spatial extended data bases (10g versions >= 10.2.0.1):

Attribute	Description
`name`	This is a string which uniquely identifies the meaning of the 'value'. It may be one of the following: • type • encoding (optional) • estimate_extent (optional) • extent (optional) • geometry_field (optional) • host • initial_size (optional) • max_size (optional) • multiple_geometries • password • row_limit (optional) • row_prefetch (optional) • table • use_spatial_index (optional) • user

Name	Value Description
`type`	Value is: occi
`encoding`	Specifies the character encoding used. If omitted, the default is utf-8.
`estimate_extent`	If present, it is used to have mapnik to estimate the extents of the layer and may be one of: • true (estimate the extents) • false (do not estimate extents) default if omitted
`extent`	If present, identifies the layer extents and has the following form: minX,minY,maxX,maxY where each comma separated value is a real number (double) and the units are expressed in terms of degrees or meters depending on the layer's SRS.
`geometry_field`	If present, identifies the name of the geometry field in the associated table. If omitted the default is GEOLOC. Type is string.
`host`	Identifies the network name for the host that contains the data base server. Type is string.
`initial_size`	If present, identifies the initial size ???. If omitted the default is 1. Type is integer.

Name	Value Description
max_size	If present, identifies the maximum size ???. If omitted the default is 10. Type is integer.
multiple_geometries	If present, identifies that the geometry retrieved may contain multiple geometries or, more than one object of the same type. It may be one of: • true (multiple geometries present) • false (assume single geometries) default if omitted
password	This is the password required for the specified user to access the data base. It may be omitted if the data base does not use passwords, or there is some other mechanism in place (environment variable) that the underlying data base API uses when accessing a data base. Type is string.
row_limit	Default 0.
row_prefetch	Default 100.
table	This is the name of the table containing the geometries and attributes used on the layer. Type is string. NOTE: You may also use a SQL query in the form of: (query) as table eg. (SELECT * from mickey where “mouse” = 'minnie') AS me
use_spatial_index	If present, indicates that mapnik should use the spatial index features of the data base. If may be one of: • true (use spatial index) • false (do not use spatial index) default if omitted
user	This is the name of the user required to access the data base. It may be omitted if there is some other mechanism in place (environment variable) that the underlying data base API uses when accessing a data base. Type is string.

Parameter – 'ogr' type

This set of parameters are for the OGR supported raster types:

Attribute	Description
`name`	This is a string which uniquely identifies the meaning of the 'value'. It may be one of the following: • type • encoding (optional) • file • layer • multiple_geometries

Name	Value Description
`type`	Value is: ogr
`encoding`	Specifies the character encoding used in the gdal file for attribute data. If omitted, the default is utf-8.
`file`	Value is the location and name of a OGR file
`layer`	Specifies the layer wanted. Type is string.
`multiple_geometries`	If present, identifies that the geometry retrieved may contain multiple geometries or, more than one object of the same type. It may be one of: • true (multiple geometries present) • false (assume single geometries) default if omitted

Parameter – 'osm' type

This set of parameters are for the Open Street Maps (OSM) map definition file:

Attribute	Description
`name`	This is a string which uniquely identifies the meaning of the 'value'. It may be one of the following: • type • bbox • encoding (optional) • file • parser (optional) • url **NOTE:** You may use file or, url and bbox, but not both

Name	Value Description
`type`	Value is: osm
`bbox`	This is a string used to specify the bounding box (bbox) coordinates. It has the form of: minX,minY,maxX,maxY where each comma separated value is a real number (double) and the units are expressed in terms of degrees or meters depending on the layer's SRS.
`encoding`	Specifies the character encoding used in the gdal file for attribute data. If omitted, the default is utf-8.
`file`	Value is the location and name of a OSM file
`parser`	If present, specifies the XML parser to be used to process the file. If omitted the default is libxml2. Type is string.
`url`	Specifies the URL where the OSM file may be obtained. Type is string.

Parameter – 'postgis' type

This set of parameters are for the PostgreSQL spatial extended data base:

Attribute	Description
`name`	This is a string which uniquely identifies the meaning of the 'value'. It may be one of the following: • type • cursor_size (optional) • dbname • estimate_extent (optional) • extent (optional) • extent_from_subquery (optional) • geometry_field (optional) • geometry_table (optional) • host • initial_size (optional) • max_size (optional) • multiple_geometries (optional) • password • persist_connection (optional) • port • row_limit (optional) • srid (optional) • table • user

Name	Value Description
`type`	Value is: postgis
`cursor_size`	It present, . If omitted the default used is 0. Type is integer.
`dbname`	Specifies the name of the data base containing the specified table. Type is string.
`estimate_extent`	If present, it is used to have mapnik to estimate the extents of the layer and may be one of: • true (estimate the extents) • false (do not estimate extents) default if omitted
`extent`	If present, identifies the layer extents and has the following form: minX,minY,maxX,maxY where each comma separated value is a real number (double) and the units are expressed in terms of degrees or meters depending on the layer's SRS.

Name	Value Description
extent_from_subquery	When true (while the 'extent' parameter is not provided and 'estimate_extent' is false) will direct Mapnik to calculate the extent upon the exact table or sql provided in the 'table' parameter. If a sub-select is used for the table parameter then this will, in cases where the subquery limits results, provide a faster and more accurate layer extent. It will have no effect if the 'table' parameter is simply an existing table. If omitted the default is false.
geometry_field	If present, identifies the name of the geometry field in the associated table. If omitted the default is “”. Type is string. If specified these will allow Mapnik to skip several queries to try to determine these values dynamically, and can be helpful to avoid possible query failures during metadata lookup with complex subqueries, but also solvable by specifying the 'geometry_table' parameter.
geometry_table	If present, identifies the name of the geometry table. If omitted the default is “”. Type is string. The 'geometry_table' used by Mapnik to look up metadata in the geometry_columns and calculate extents (when the 'geometry_field' and 'srid' parameters are not supplied). If 'geometry_table' is not specified Mapnik will attempt to determine the name of the table to query based on parsing the 'table' parameter, which may fail for complex queries with more than one 'from' keyword. Using this parameter should allow for existing metadata and table indexes to be used while opening the door to much more complicated subqueries being passed to the 'table' parameter without failing
host	Identifies the network name for the host that contains the data base server. Type is string.
initial_size	If present, identifies the initial size ???. If omitted the default used is 1. Type is integer.
max_size	If present, identifies the maximum size ???. If omitted the default used is 10. Type is integer.
multiple_geometries	If present, identifies that the geometry retrieved may contain multiple geometries or, more than one object of the same type. It may be one of: ● true (multiple geometries present) ● false (assume single geometries) default if omitted
password	This is the password required for the specified user to access the data base. It may be omitted if the data base does not use passwords, or there is some other mechanism in place (environment variable) that the underlying data base API uses when accessing a data base. Type is string.
persist_connection	When false, the connection to the database will be released after each operation. If omitted the default is true.
port	The port number that the postgreSQL server is accessed on. PostgreSQL listens, by default, on port 5432.
row_limit	If present, ???. If omitted the default used is 0. Type is integer.

Name	Value Description
`srid`	If omitted the default is 0. If specified these will allow Mapnik to skip several queries to try to determine these values dynamically, and can be helpful to avoid possible query failures during metadata lookup with complex subqueries, but also solvable by specifying the 'geometry_table' parameter.
`table`	This is the name of the table containing the geometries and attributes used on the layer. Type is string. If a table name contains a special character, then you must surround the table name with quotes. **NOTE**: You may also use a SQL query in the form of: (query) as table `eg. (SELECT * from mickey where “mouse” = 'minnie') AS me`
`user`	This is the name of the user required to access the data base. It may be omitted if there is some other mechanism in place (environment variable) that the underlying data base API uses when accessing a data base. Type is string.

Query Substitution Parameters:

Internal Mapnik tokens which will be substituted when used in SQL statements.

Name	Value Description
`bbox`	(Select ST_Union(geom) as geom from table where ST_Intersects(geometry,!bbox!)) as map
`scale_denominator`	(Select * from table where field_value > !scale_denominator!) as map

Parameter – 'raster' type

This set of parameters are for Raster TIFF or PNG image files:

Attribute	Description
`name`	This is a string which uniquely identifies the meaning of the 'value'. It may be one of the following: • type • base (optional) • file • format • lox • loy • hix • hiy

Name	Value Description
`type`	Value is: raster
`base`	If present, contains the full path (without the trailing /) where the files are located. It is combined with the type for file as follows: file = base + / + file Type is string.
`file`	Value is the location and name of a raster TIFF file (without the extension). Type is string.
`format`	Identifies the type of image file and may be one of: • tiff • png Type is string.
`lox`	Minimum X coordinate for the image. The value is a real number (double) and the units are expressed in terms of degrees or meters depending on the layer's SRS.
`loy`	Minimum Y coordinate for the image. The value is a real number (double) and the units are expressed in terms of degrees or meters depending on the layer's SRS.
`hix` •	Maximum X coordinate for the image. The value is a real number (double) and the units are expressed in terms of degrees or meters depending on the layer's SRS.
`hiy` •	Maximum Y coordinate for the image. The value is a real number (double) and the units are expressed in terms of degrees or meters depending on the layer's SRS.

Parameter – 'shape' type

This set of parameters are for ESRI Shapefile sets:

Attribute	Description
`name`	This is a string which uniquely identifies the meaning of the 'value'. It may be one of the following: • type • base (optional) • encoding (optional) • file

Name	Value Description
`type`	Value is: shape
`base`	If present, contains the full path (without the trailing /) where the files are located. It is combined with the type for file as follows: file = base + / + file
`encoding`	Specifies the character encoding used in the shape file set for attribute data. If omitted, the default is utf-8. ESRI Shapefiles are typically “latin1”
`file`	Value is the full path and name of an ESRI Shape file set (without the extension) eg. if the shape file is filename.shp, then use filename as the value string. As an absolute minimum, there are three files in a set: • filename.dbf <-- contains feature attributes • filename.shp <-- contains feature geometry • filename.shx <-- contains indexes to feature geometry and you REALLY should have the file: • filename.prj <-- contains the projection definition for the geometry should you every need to know the spatial reference system for the shapefile geometry

Parameter – 'sqlite' type

This set of parameters are for the SQLite spatial extended data base files:

Attribute	Description
`name`	This is a string which uniquely identifies the meaning of the 'value'. It may be one of the following: • type • base • encoding (optional) • extent (optional) • file • geometry_field (optional) • key_field (optional) • metadata (optional) • multiple_geometries (optional) • row_offset (optional) • row_limit (optional) • table (optional) • wkb_format (optional) • use_spatial_index (optional)

Name	Value Description
`type`	Value is: sqlite
`base`	If present, contains the full path (without the trailing /) where the files are located. It is combined with the type for file as follows: file = base + / + file
`encoding`	Specifies the character encoding used in the shape file set for attribute data. If omitted, the default is utf-8.
`extent`	If present, identifies the layer extents and has the following form: minX,minY,maxX,maxY where each comma separated value is a real number (double) and the units are expressed in terms of degrees or meters depending on the layer's SRS.
`file`	Value is the location and name of a SQLite file (without extension). Type is string.
`geometry_field`	If present, identifies the name of the geometry field in the associated table. If omitted the default is “the_geom”. Type is string.
`key_field`	. If omitted the default is OGC_FID. Type is string.
`metadata`	. Type is string.
`multiple_geometries`	If present, identifies that the geometry retrieved may contain multiple

Name	Value Description
	geometries or, more than one object of the same type. It may be one of: • true (multiple geometries present) • false (assume single geometries) default if omitted
row_offset	If present, ???. If omitted the default used is 0. Type is integer.
row_limit	If present, ???. If omitted the default used is 0. Type is integer.
table	This is the name of the table containing the geometries and attributes used on the layer. Type is string. NOTE: You may also use a SQL query in the form of: (query) as table eg. (SELECT * from mickey where “mouse” = 'minnie') AS me
wkb_format	. If omitted the default is “spatialite”. Type is string.
use_spatial_index	If present, indicates that mapnik should use the spatial index features of the data base. If may be one of: • true (use spatial index) default if omitted. • false (do not use spatial index)

Example

```
<Datasource>
      <Parameter name="type">shape</Parameter>
      <Parameter name="file">/mnt/GIS1/Testing/NRN/roadseg</Parameter>
</Datasource>

<Datasource>
    <Parameter name="type">postgis</Parameter>
    <Parameter name="host">fileserver1</Parameter>
    <Parameter name="port">5432</Parameter>
    <Parameter name="user">micky</Parameter>
    <Parameter name="password">mouse</Parameter>
    <Parameter name="dbname">lethbridge</Parameter>
    <Parameter name="table">nrn_roadseg</Parameter>
</Datasource>
```

See the Appendix for a more detailed example and how all the elements are combined.

FileSource Element

Specifies a full path name to a file or set of files and associates a Name with it. This allows later elements to reference it by Name (See the description for base).

This element, if used, must be located directly in the root Map root element. The following elements can then reference it by supplying the element's name in the `base` attribute:

- LinePatternSymbolizer
- PointSymbolizer
- PolygonPatternSymbolizer
- ShieldSymbolizer

General Form

```
<FileSource name="">path</FileSource>
```

Attribute	Description
`name`	This is a string which uniquely identifies (associated with) the `path`. Default is: `"Unnamed"` The `path` value consists of the full path to a file or set of files without the trailing '/'.

Children

None.

Example

```
<FileSource name="myIcons">/home/david/maps/images</FileSource>
```

See the Appendix for a more detailed example and how all the elements are combined.

FontSet Element

Not all font sets contain all the characters that text being rendered use. This element allows you to create a list of fonts that, if necessary, are searched to locate a character that is missing. The first listed font is the primary one and only if it does not contain a desired character are the remaining ones in the list searched.

General Form

```
<FontSet name="">
    <Font face_name="" />   <-- repeat 0 or more times
</FontSet>
```

Attribute	Description
`name`	This is a string which uniquely identifies (associated with) this set. Default is: `"Unnamed"`

Children

There is currently one (1) element that must be contained within this element and is as follows:

Font Element

Attribute	Description
`face_name`	This is the Style Name of the desired Face within the Font family's different faces. It is NOT the name of the file containing the font, although it may be very similar. eg. Family: DejaVu, Face: DejaVu Sans Book, file: DejaVuSans.ttf

Example

```
<FontSet name="myFonts">
    <Font face_name="DejaVu Sans Book" />
    <Font face_name="DejaVu Serif Book" />
</FontSet>
```

See the Appendix for a more detailed example and how all the elements are combined.

Layer Element

A Layer element combines Styles and Datasources for rendering. You may have more than one Style or Datasource in a given layer. However, Style interaction can result in unexpected results, especially if you the styles are using the same symbolizer.

Layers are rendered in the order in which they appear in the Map definition file; each successive layer definition rendered over top of the preceding ones. Typically referred to as Painter's Order.

General Form

```
<Layer name="" srs="" status="" title="" abstract="" minzoom="" maxzoom =""
       queryable="" clear_label_cache="">
  <StyleName>(name from Styles' name attribute)</StyleName>
  <Datasource base="">
    <Parameter name="">value</Parameter>
  </Datasource>
</Layer>
```

Attribute	Description
`abstract`	This is a string of text describing the layer. Default is: ""
`clear_label_cache`	This is used to clear the label cache, between layers, so that overlapping text will render when using multiple layers where the text of each layer's labels overlap. Default is: “no" or "0" may be one of "yes" or "1", "no" or "0"
`minzoom`	The layer is rendered if the map rendering scale is >= value specified. Default is: “0”. Value range is the same as that for MinScaleDenominator. See http://trac.mapnik.org/wiki/ScaleAndPpi for detailed explanation.
`maxzoom`	The layer is rendered if the map rendering scale is < value specified. Default is: “some very large number”. Value range is the same as that for MaxScaleDenominator. See http://trac.mapnik.org/wiki/ScaleAndPpi for detailed explanation.
`name`	This is a string which uniquely identifies (associated with) this layer. Default is: `"Unnamed"`
`queryable`	This is simply a flag that can be set to indicate whether or not this layer can be queried (from a data base source). Default is: "off" or "0"

Attribute	Description
	may be one of "on" or "1", "off" or "0"
srs	Spatial Reference System for the layer's Datasources. default: is the root Map element srs value All sources of geometry used in the maps rendering will, if they are not in the same SRS as the map, be re-projected.
status	Whether or not this layer is to be rendered. Default is: "on" or "1" may be one of "on" or "1", "off" or "0"
title	This is a character string representing the layers title. Default is: ""

Children

There are two (2) elements which may be contained within this element:

- Datasource Element, one is required and may be repeated zero (0) or more times
- StyleName Element, one is required and may be repeated zero (0) or more times

StyleName Elements are listed in the order they are to be applied to each data source. It should be noted that when you have more than one style and or data source, the precedence of what get rendered first and in what order can lead to unexpected results. Especially if the styles have overlapping rules and symbolizers.

Datasource Elements are listed in the order that you want them processed, remembering that all the listed StyleName Elements are processed against a Datasource before moving to the next source.

Example

Simple

```
<Style name="OtherRoad">
  <Rule>
    <Filter>[roadclass] = 'Collector' and [rtnumber1] = 'None'</Filter>
    <MaxScaleDenominator>50000</MaxScaleDenominator>
    <MinScaleDenominator>30000</MinScaleDenominator>
    <LineSymbolizer>
      <CssParameter name="stroke">rgb(255,0,0)</CssParameter>
      <CssParameter name="stroke-width">1</CssParameter>
      <CssParameter name="stroke-linejoin">bevel</CssParameter>
      <CssParameter name="stroke-linecap">round</CssParameter>
    </LineSymbolizer>
  </Rule>
</Style>
```

```
<Layer name="OtherRoad" status="on" srs="+proj=merc +lon_0=0 +k=1 +x_0=0 +y_0=0
       +ellps=WGS84 +datum=WGS84 +units=m +no_defs">
  <StyleName>OtherRoad</StyleName>
  <Datasource>
    <Parameter name="type">postgis</Parameter>
    <Parameter name="host">fileserver1</Parameter>
    <Parameter name="port">5432</Parameter>
    <Parameter name="user">micky</Parameter>
    <Parameter name="password">mouse</Parameter>
    <Parameter name="dbname">lethbridge</Parameter>
    <Parameter name="table">nrn_roadseg</Parameter>
  </Datasource>
</Layer>
```

Using a template

```
 <Datasource name=”commonDB”>
   <Parameter name="type">postgis</Parameter>
   <Parameter name="host">fileserver1</Parameter>
   <Parameter name="port">5432</Parameter>
   <Parameter name="user">micky</Parameter>
   <Parameter name="password">mouse</Parameter>
 </Datasource>

<Layer name="OtherRoad" status="on" srs="+proj=merc +lon_0=0 +k=1 +x_0=0 +y_0=0
      +ellps=WGS84 +datum=WGS84 +units=m +no_defs">
  <StyleName>OtherRoad</StyleName>
  <Datasource base=”commonDB”>
    <Parameter name="dbname">lethbridge</Parameter>
    <Parameter name="table">nrn_roadseg</Parameter>
  </Datasource>
</Layer>
```

See the Appendix for a more detailed example and how all the elements are combined.

Rule Element

General Form

```
<Rule name="" title="">
</Rule>
```

Attribute	Description
`name`	This is a string which uniquely identifies (associated with) this style. This name value is subsequently used in a StyleName Element within a Layer Element. It is a required attribute. Type is string.
`title`	This is a string value that is used by external programs. Type is string.

Children

There is currently fourteen (14) elements that may be contained within this element:

- Filter
- ElseFilter
- MinScaleDenominator
- MaxScaleDenominator
- BuildingSymbolizer
- LineSymbolizer
- LinePatternSymbolizer
- MarkersSymbolizer
- PointSymbolizer
- PolygonSymbolizer
- PolygonPatternSymbolizer
- RasterSymbolizer
- ShieldSymbolizer
- TextSymbolizer

The following provide a detailed explanation of each element.

Filter

This element is used to select data from a data source for rendering. If data are actually selected, then the elements which follow it are used to render the selected data. All subsequent elements are evaluated and used up to the end of the current Rule.

The Filter and ElseFilter Elements can NOT be used in the same Rule.

The expression that is contained in the

General Form

```
<Filter>
     <expression>
</Filter>
```

where <expression> has one of the following forms:

```
[Field] or,
[Field] operator 'value' or,
[Field] operator 'value' operator ...
```

Description

`[Field]`	Is the name of the field or attribute in a data source whose corresponding value is used in the evaluation of the overall expression. The case of the characters between the [and] are critical and dependant on the data source. ESRI Shape file usually have all UPPER case with no spaces. PostreSQL tables built from shape files are all LOWER case.
`value`	The value enclosed between two single quotes (') is dependant on the operator being used; numeric, boolean or string types.
`Operator`	The available operators are:

Symbol	Precedence	Description
( )		Enclose a sub-expression
[ ]		Enclose a Field/Attribute Name
*		multiply
/		divide
+		add
-		subtract
%		modulo
=		equal to or equivalent to
<> !=		<> not equal not equal
&lt;		< less than
<=		<= less than or equal
&gt;		> greater than
>=		>= greater than or equal
and		and
or		or

Symbol	Precedence	Description
not		unary not
.match(')		exact match regular expression within single quotes. The regular expression must be constructed to match the entire value, not just a sub-string within the value.
min		minimum value given
max		maximum value
sqrt		square root of value
sin		sine of value
cos		cosine of value
Equals		Spatial Equals
Disjoint		Spatial Disjoint
Touches		Spatial Touches
Within		Spatial Within
Overlaps		Spatial Overlaps
Crosses		Spatial Crosses
Intersects		Spatial Intersect
Contains		Spatial contains
DWithin		Spatial DWithin
Beyond		Spatial Beyond
BBOX		Spatial Bounding Box
true		boolean constant true
false		boolean constant false

Children

None.

ElseFilter

This element is used to rendered data that was NOT selected by the **preceding Rule**. If there is data not selected in the preceding Rule, then the elements which follow the ElseFilter are used to render the remaining data. All subsequent elements are evaluated and used up to the end of the current Rule.

The Filter and ElseFilter Elements can NOT be used in the same Rule. However, multiple ElseFilter rules can follow a Filter or ElseFilter rule. This is particularly useful if the ElseFilter rule contains the MinScaleDenominator and / or the MaxScaleDenominator elements.

General Form

```
<ElseFilter/>
```

Children

None.

MinScaleDenominator

This element is used to specify the minimum scale denominator. The current Mapnik scale must be greater than or equal (>=) to this value before the current Rule is used to render selected data.

General Form

```
<MinScaleDenominator>value</MinScaleDenominator>
```

The value is a numeric string. This link, http://trac.mapnik.org/wiki/ScaleAndPpi, provides an explanation of how scaling is determined in Mapnik. Further, Appendix A illustrates the various Denominator values that are appropriate if you were creating Google Maps tiles.

Children

None.

MaxScaleDenominator

This element is used to specify the maximum scale denominator. The current Mapnik scale must be less than (<) to this value before the current Rule is used to render selected data.

General Form

```
<MaxScaleDenominator>value</MaxScaleDenominator>
```

The value is a numeric string. This link, http://trac.mapnik.org/wiki/ScaleAndPpi, provides an explanation of how scaling is determined in Mapnik. Further, Appendix A illustrates the various Denominator values that are appropriate if you were creating Google Maps tiles.

Children

None.

BuildingSymbolizer

This element is used to create a pseudo 3D effect on polygons.

General Form

```
<BuildingSymbolizer>
      <CssParameter name="">value</CssParameter>
</BuildingSymbolizer>
```

Children

There is one element that has 3 variations as follows:

CssParameter

This element is repeated three (3) times, with the name attribute changing, to completely describe the rendering.

Attribute	Description
`name`	This is a string which uniquely identifies the meaning of the 'value'. It may be one of the following: • fill • fill-opacity • height

Name	Value Description
`fill`	Value specifies the color to fill the interior of the polygon with. If omitted the default is #808080 or gray (or grey) There are three possible styles specifying a color: • `#rrggbb` - a set of hexidecimal values representing the Red(rr), Green(gg) and Blue(bb) color values. • `rgb(rrr,ggg,bbb)` - a set of Decimal values representing the color values. • color name - a standard alphabetic character string name of a color. See Appendix B for some more common values and actual color swatches.
`fill-opacity`	Value specifies the opacity ratio as a number between 0.0 and 1.0. A value of 0.0 is completely transparent and 1.0 is completely opaque . If omitted the default is 1.0.
`height`	Value is the number of pixels to use simulating the 3D effect. If omitted the default is 0. Info: the shaded portion (3D) has a color which is 80% of the fill r, g, b values. (darker)

LineSymbolizer

This element is used to define how lines (linear geometry) are rendered using a simple line stroke. A line may be a discrete segment or a polyline containing multiple segments ordered end to end.

General Form

```
<LineSymbolizer>
      <CssParameter name="">value</CssParameter>
</LineSymbolizer>
```

Children

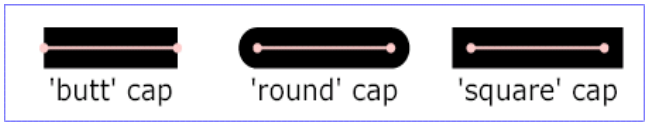
There is one element and it has 6 variations as follows:

CssParameter

This element may be repeated six (6) times, with the name attribute changing, to completely describe the rendering.

Attribute	Description
`name`	This is a string which uniquely identifies the meaning of the 'value'. It may be one of the following: ● stroke ● stroke-width ● stroke-opacity ● stroke-linejoin ● stroke-linecap ● stroke-dasharray

Name	Value Description
`stroke`	Value specifies the color to fill the width of the segment(s) with. If omitted the default ois #000000 or black. There are three possible styles specifying a color: ● `#rrggbb` - a set of hexidecimal values representing the Red(rr), Green(gg) and Blue(bb) color values. ● `rgb(rrr,ggg,bbb)` - a set of Decimal values representing the color values. ● color name - a standard alphabetic character string name of a color. See Appendix B for some more common values and actual color swatches.
`stroke-width`	Value specifies the number of pixels that the segment width is to be rendered at. This is real number allowing you to specify a fraction of a pixel. If omitted, the default is 1.0 pixels.

Name	Value Description
	eg. Use 2.5 if you want the line to be approximately 2.5 pixels wide. Mapnik uses shading in adjacent pixels to effect the fractional portion.
`stroke-opacity`	Value specifies the opacity ratio as a number between 0.0 and 1.0. A value of 0.0 is completely transparent and 1.0 is completely opaque . If omitted the default is 1.0.
`stroke-linejoin`	Value specifies the method to use when distinct line features intersect. If may be one of: • miter • miter_revert • round • bevel If omitted the default is miter. 
`stroke-linecap`	Value specifies how the end of lines are rendered and may be one of: • round • butt • square If omitted the default is butt 
`stroke-dasharray`	If this CssParameter is present, it will change a Solid line into a 'dashed' line. The dash style is defined with the Value containing a series of comma (,) separated number pairs. The first number is the dash length and the second number is the gap length. If there are an odd number of values, the whole list is repeated to produce an even number of pairs. eg. 20 --> 20,20 10,5 --> 10,5 10,5,3 --> 10,5,3,10,5,3

LinePatternSymbolizer

This element is used to define how lines (linear geometry) are rendered using a graphic element rather than a simple line stroke.. A line may be a discrete segment or a polyline containing multiple segments ordered end to end.

General Form

```
<LinePatternSymbolizer
      base="" file="" height="" type="" width=""
/>
```

Children

There are no additional elements.

Attributes

Name	Value Description
`base`	If present, contains the name of a FileSource Element definition (see FileSource Element). The value associated with the named FileSource is combined with the value for file as follows: file = base + / + file
`file`	Value specified the file name containing the graphic to be used and may be in one of two forms: ● if the base parameter is present, then it is generally just the file name ● If the base parameter is not present, then it specifies the complete path and file name
`height`	Value specifies the the height, in pixels, of the graphic
`type`	Value specifies the graphic format type and may be one of: ● tiff ● png
`width`	Value specifies the width, in pixels. Of the graphic

MarkersSymbolizer

This element is used to define how lines (linear geometry) are rendered using a graphic element rather than a simple line stroke.. A line may be a discrete segment or a polyline containing multiple segments ordered end to end.

General Form

```
<MarkersSymbolizer
      allow_overlap=””
/>
```

Children

There are no additional elements.

Attributes

Name	Value Description
`allow_overlap`	If present, it is used to allow or disallow Marker symbols to over lap. May be one of: ● “yes” or “true” or “1” - allow over lap ● “no” or “false” or “0” - disallow over lap Default is “no” or “false” or “0”.

PointSymbolizer

This element is used to define how lines (linear geometry) are rendered using a graphic element rather than a simple line stroke.. A line may be a discrete segment or a polyline containing multiple segments ordered end to end.

General Form

```
<PointSymbolizer
      allow_overlap="" base="" file="" height="" type="" width=""
/>
```

Children

There are no additional elements.

Attributes

Name	Value Description
`allow_overlap`	If present, it is used to allow or disallow the text to be over lapped. May be one of: ● "yes" or "true" or "1" - allow over lap ● "no" or "false" or "0" - disallow over lap Default is "no" or "false" or "0".
`base`	If present, contains the name of a FileSource Element definition (see FileSource Element). The value associated with the named FileSource is combined with the value for file as follows: file = base + / + file
`file`	Value specified the file name containing the graphic to be used and may be in one of two forms: ● if the base parameter is present, then it is generally just the file name ● If the base parameter is not present, then it specifies the complete path and file name
`height`	Value specifies the the height, in pixels, of the graphic
`opacity`	If present, the value specifies the opacity ratio as a number between 0.0 and 1.0. A value of 0.0 is completely transparent and 1.0 is completely opaque . If omitted the default is 1.0.
`type`	Value specifies the graphic format type and may be one of: ● tiff ● png
`width`	Value specifies the width, in pixels, of the graphic

PolygonSymbolizer

This element is used to define how polygons are rendered. Specifically, how the area defined by the polygon boundary is filled with color

General Form

```
<PolygonSymbolizer>
      <CssParameter name=””>value</CssParameter>
</PolygonSymbolizer>
```

Children

There is one element that has 2 variations as follows:

CssParameter

This element is repeated three (2) times, with the name attribute changing, to completely describe the rendering.

Attribute	Description
`name`	This is a string which uniquely identifies the meaning of the 'value'. It may be one of the following: • fill • fill-opacity

Name	Value Description
`fill`	Value specifies the color to fill the interior of the polygon with. If omitted the default is #808080 or gray (or grey) There are three possible styles specifying a color: • `#rrggbb` - a set of hexidecimal values representing the Red(rr), Green(gg) and Blue(bb) color values. • `rgb(rrr,ggg,bbb)` - a set of Decimal values representing the color values. • color name - a standard alphabetic character string name of a color. See Appendix B for some more common values and actual color swatches.
`fill-opacity`	Value specifies the opacity ratio as a number between 0.0 and 1.0. A value of 0.0 is completely transparent and 1.0 is completely opaque . If omitted the default is 1.0.
`gamma`	Value specifies the gamma setting as a number between 0.0 and 1.0. This may be a a solution for gap artifacts or "ghost lines" between adjacent polygons and allows for slight sharpening of the edges of non overlapping polygons. Depends on background color and alpha values but in test cases .5 to .7 seems to work great to remove rendering artifacts while

Name	Value Description
	retaining AA look and good detail. If omitted, the default is 1.0.

PolygonPatternSymbolizer

This element is used to define how polygons are rendered using a graphic element rather than a simple area fill.

General Form

```
<PolygonPatternSymbolizer
      base="" file="" height="" type="" width=""
/>
```

Children

There are no additional elements.

Attributes

Name	Value Description
`base`	If present, contains the name of a FileSource Element definition (see FileSource Element). The value associated with the named FileSource is combined with the value for file as follows: file = base + / + file
`file`	Value specified the file name containing the graphic to be used and may be in one of two forms: ● if the base parameter is present, then it is generally just the file name ● If the base parameter is not present, then it specifies the complete path and file name
`height`	Value specifies the the height, in pixels, of the graphic
`type`	Value specifies the graphic format type and may be one of: ● tiff ● png
`width`	Value specifies the width, in pixels. Of the graphic

RasterSymbolizer

This element is used to define how Raster Images are rendered.

NOTE: The raster image must be geo-referenced using the same projection as the Map declaration.

General Form

```
<RasterSymbolizer>
      <CssParameter name="">value</CssParameter>
</RasterSymbolizer>
```

Children

There are no additional elements.

Attributes

Name	Value Description
`mode`	Specifies over lapping images are to be combined. It may be one of: • grain_merge • grain_merge2 • multiply • multiply2 • divide • divide2 • screen • hard_light • normal (default) Compositing / Merging effects with image below raster level (?). The formula for combining foreground (raster) and background are: grain_merge: bg + fg - 0.5, grain_merge2: bg + 2 * fg - 1.0, multiply: fg * bg, multiply2: 2 * fg * bg, divide: bg / fg, divide2: 2 * bg / fg, screen: (1 - fg) * (1 - bg), hardlight: see http://docs.gimp.org/en/gimp-concepts-layer-modes.html#id2834930
`opacity`	Value specifies the opacity ratio as a number between 0.0 and 1.0. A value of 0.0 is completely transparent and 1.0 is completely opaque . If omitted the default is 1.0.
`scaling`	Specifies how scaling of the image is to be performed. It may be one of: • bilinear, uses bilinear interpolation for all 4 channels (RGBA) • bilinear8, like bilinear, but only one channel assumed • fast, uses nearest neighbour

ShieldSymbolizer

This element is used to define how polygons are rendered using a graphic element rather than a simple area fill.

General Form

```
<ShieldSymbolizer
      base="" file="" height="" type="" width="" ...
/>
```

Children

There are no additional elements.

Attributes

Common

Name	Value Description
allow_overlap	Allow text and shield to over lap other text / shield. Maybe one of “true” or “false”. If omitted, the default is “false”.
avoid_edges	Avoid placing text / shield near edges of the map. Maybe one of “true” or “false”. If omitted, the default is “false”.
unlock_image	Allows the text to move but the image is centred over the point. Maybe one of “true” or “false”. Default is “false”.
opacity	Value specifies the opacity ratio, for both text and shield, as a number between 0.0 and 1.0. A value of 0.0 is completely transparent and 1.0 is completely opaque . If omitted, the default is “1.0”.

Image Specific

Name	Value Description
base	If present, contains the name of a FileSource Element definition (see FileSource Element). The value associated with the named FileSource is combined with the value for file as follows: file = base + / + file
file	Value specifies the file name containing the graphic to be used and may be in one of two forms: ● if the base parameter is present, then it is generally just the file name ● If the base parameter is not present, then it specifies the complete path and file name
height	Value specifies the the height, in pixels, of the image.
type	Value specifies the graphic format type and may be one of: ● tiff ● png

Name	Value Description
`width`	Value specifies the width, in pixels. Of the image.

Text Specific

Name	Value Description
`character_spacing`	Specify the number of additional pixels inserted between characters. If omitted, the default is “0”.
`dx`	Displace the text by a fixed amount, in pixels, +/- along x-axis. A positive value will shift the text right. **NOTE**: Only the text is affected.
`dy`	Displace text by a fixed amount, in pixels, +/- along y-axis. A positive value will shift the text down. **NOTE**: Only the text is affected.
`face_name`	Identify the Style Name of the desired Face within the Font family's different faces. It is NOT the name of the file containing the font, although it may be very similar. eg. Family: DejaVu, Face: DejaVu Sans Book, file: DejaVuSans.ttf **NOTE**: Use either face_name or fontset_name, but not both.
`fontset_name`	If present is the name of the Font Set defined using the FontSet Element (see FontSet Element for description). **NOTE**: Use either face_name or fontset_name, but not both.
`fill`	Specifies the color for the text. There are three possible styles specifying a color: ● `#rrggbb` - a set of hexidecimal values representing the Red(rr), Green(gg) and Blue(bb) color values. ● `rgb(rrr,ggg,bbb)` - a set of Decimal values representing the color values. ● color name - a standard alphabetic character string name of a color. See Appendix B for some more common values and actual color swatches. If omitted, the default is “#000000” or “rgb(0,0,0)” or “black”.
`halo_fill`	Specifies the color of the halo around the text. There are three possible styles specifying a color: ● `#rrggbb` - a set of hexidecimal values representing the Red(rr), Green(gg) and Blue(bb) color values. ● `rgb(rrr,ggg,bbb)` - a set of Decimal values representing the color values. ● color name - a standard alphabetic character string name of a color. See Appendix B for some more common values and actual color swatches. If omitted, the default is “#FFFFFF” or “rgb(255,255,255)” or “white”.
`halo_radius`	Specify the radius of the halo in pixels. Must be an integer value. If omitted, the default is “0” (no halo)

Name	Value Description
horizontal_alignment	Specifies text horizontal alignment relative to the reference point. It may be one of: • left, all text is to the left of the reference. • middle, all text is horizontally centered on the the reference. • right, all text is to the right of the reference. If omitted, the default is “bottom”. **NOTE:** If “unlock_image” is “false”, then both shield and text are affected. Otherwise only the text is affected
justify_alignment	Specifies line text justification relative to the horizontal alignment. It is only applicable when there is more than one line of text. It may be one of: • left, all lines are left justified. • middle, all lines are center justified. • Right, all lines are right justified. If omitted, the default is “middle”. This only affects the text.
line_spacing	Specifies the number of additional pixels inserted between successive lines of text. If omitted, the default is “0”.
min_distance	Specifies the minimum distance, in pixels, between adjacent text / shields. If omitted, the default is “0”.
name	The field name (from the data source), whose value is to be rendered as text. **WARNING:** This value is case sensitive and must be of the same case as the data source.
no_text	Specifies whether or not the source text is used or if a 'space' character (' ') is used instead. It may be one of the following: • false, source text is used • true, a single space character is substituted for all text When “true”, the text that would normally be rendered with the image is discarded and replaced with a single space character. This effectively allows the image to be rendered but with no apparent text. However, the 'space' character is subject to all text related attributes.
placement	Specifies how the text / shield is placed. It may be one of the following: • point, at the point (or first point of a line) • line, at the mid-point of a line, or repeats at 'spacing' intervals • vertex, every vertex of a line If omitted, the default is “point”.
size	Specifies the text size, in pixels. Default is “10”.
spacing	Specifies the distance, in pixels, along a line between repeated text / shields. If omitted, the default is “0” - only a single instance of the text/shield are rendered. **NOTE:** The first shield is placed ½ the spacing from the beginning of a line.
text_convert	Specifies how the text character's case is to be adjusted before rendering. It

Name	Value Description
	may be one of: ● none, No change to character case ● toupper, characters all converted to Upper Case ● tolower, characters all converted to Lower Case If omitted, the default is “none”.
`vertical_alignment`	Specifies text vertical alignment relative to the reference point. It may be one of: ● top, all text is above the reference. ● middle, all text is vertically centered on the the reference. ● Bottom, all text is below the reference. If omitted, the default is “middle” unless **NOTE:** If “unlock_image” is “false”, then both shield and text are affected. Otherwise only the text is affected
`wrap_before`	Specifies that the line wrap point is to occur at the “wrap_char” preceding the current word rather than after the current word. May be one of “true” or “false”. If omitted, the default is “false”.
`wrap_character`	Specifies the character that is to be used to define the point in the text characters where a line wrap is to occur. The wrap character, at the wrap point, is removed from the text actually rendered. If omitted, the default is the space character (' ').
`wrap_width`	Specifies the width, in pixels, before wrapping text to a new line. Text wrapping occurs at the first 'wrap_char' after the 'wrap_width. If omitted, the default is “0” (at every 'wrap_char').

TextSymbolizer

This element is used to define how text is rendered.

General Form

```
<TextSymbolizer
      base="" file="" height="" type="" width=""
/>
```

Children

There are no additional elements.

Attributes

Name	Value Description
`avoid_edges`	Avoid placing a text near map edges. May be one of "true" or "false". If omitted, the default is "false"
`allow_overlap`	Allow text to over lap other text. May be one of "true" or "false". If omitted, the default is "false".
`character_spacing`	Specify the number of additional pixels inserted between characters. If omitted, the default is "0".
`dx`	Displace text by fixed amount, in pixels, +/- along the X axis. A positive value will shift the text right. If omitted, the default is "0".
`dy`	Displace text by fixed amount, in pixels, +/- along the Y axis. A positive value will shift the text down. If omitted, the default is "0". **NOTE**: If `vertical_alignment` is not specified and `dy` is, then the default `vertical_alignment` may change as follows: if `dy > 0`, then "bottom" if `dy < 0`, then "top"
`face_name`	Identify the Style Name of the desired Face within the Font family's different faces. It is NOT the name of the file containing the font, although it may be very similar. eg. Family: DejaVu, Face: DejaVu Sans Book, file: DejaVuSans.ttf **NOTE:** Use either face_name or fontset_name, but not both.
`fontset_name`	Name of the Font Set defined using the FontSet Element (see FontSet Element for description). **NOTE:** Use either face_name or fontset_name, but not both.
`fill`	Specifies the color for the text. There are three possible styles specifying a color: • `#rrggbb` - a set of hexidecimal values representing the

Name	Value Description
	Red(rr), Green(gg) and Blue(bb) color values. • `rgb(rrr,ggg,bbb)` - a set of Decimal values representing the color values. • color name - an alphabetic character string name of a color. See Appendix B for some more common values and actual color swatches. If omitted, the default is “#000000” or “rgb(0,0,0)” or “black”.
`force_odd_labels`	Have every other label, along a line, rendered. May be one of “true” or “false”. If omitted, the default is “false”. **NOTE**: NOT available using the xml file.
`halo_fill`	Specifies the color of the halo around the text. There are three possible styles specifying a color: • `#rrggbb` - a set of hexidecimal values representing the Red(rr), Green(gg) and Blue(bb) color values. • `rgb(rrr,ggg,bbb)` - a set of Decimal values representing the color values. • color name - an alphabetic character string name of a color. See Appendix B for some more common values and actual color swatches. If omitted, the default is “#FFFFFF” or “rgb(255,255,255)” or “white”.
`halo_radius`	Specify the radius of the halo in pixels. Must be an integer value. If omitted, the default is “0” (no halo).
`horizontal_alignment`	Specifies text horizontal alignment relative to the reference point. It may be one of: • left, all text is to the left of the reference. • middle, all text is horizontally centered on the the reference. • right, all text is to the right of the reference. If omitted, the default is “bottom”.
`justify_alignment`	Specifies line text justification relative to the horizontal alignment. It is only applicable when there is more than one line of text. It may be one of: • left, all lines are left justified. • middle, all lines are center justified. • Right, all lines are right justified. If omitted, the default is “middle”.
`label_position_tolerance`	Allows a label to be moved within the specified number of pixels of the defining position. If omitted, the default is “0”. **NOTE**: NOT available using the xml file.
`line_spacing`	Specifies the number of additional pixels inserted between successive lines of text. If omitted the default is “0”.
`max_char_angle_delta`	If present, the maximum angle change, in degrees, allowed between adjacent characters in a label. This will stop label placement around

Name	Value Description
	sharp corners.
`min_distance`	Specifies the minimum distance, in pixels, between adjacent text. If omitted, the default is “0”.
`name`	The field name (from the data source), whose value is to be rendered. **WARNING:** This value is case sensitive and must be of the same case as the data source.
`opacity`	Specifies the opacity ratio between the text and the background as a number between 0.0 and 1.0. A value of 0.0 is completely transparent and 1.0 is completely opaque . If omitted, the default is “1.0”.
`placement`	Specifies how text is to be placed, follow the angle of a line or horizontally at a point. It may be one of: ● point, horizontally at the point (or first point of a line) ● line, at the mid-point of a line, following the angle of the line If omitted, the default is “point”.
`size`	Specifies the text size, in pixels. If omitted, the default is “10”.
`spacing`	Specifies the spacing, in pixels, between repeated labels along a line. If omitted, the default is “0” - only a single instance of the text is rendered.
`text_convert`	Specifies how the text character's case is to be adjusted before rendering. It may be one of: ● none, No change to character case ● toupper, characters all converted to Upper Case ● tolower, characters all converted to Lower Case If omitted, the default is “none”.
`text_ratio`	Define the amount of text (of the total) present on successive lines when wrapping occurs. **NOTE**: NOT available using the xml file.
`vertical_alignment`	Specifies text vertical alignment relative to the reference point. It may be one of: ● top, all text is above the reference. ● middle, all text is vertically centered on the the reference. ● bottom, all text is below the reference. If omitted, the default is “middle”. **NOTE**: If `vertical_alignment` is not specified and `dy` is, then the default may change as follows: if `dy > 0`, then “bottom” if `dy < 0`, then “top”

Name	Value Description
wrap_before	Specifies that the line wrap point is to occur at the “wrap_char” preceding the current word rather than after the current word. May be one of “true” or “false”. If omitted, the default is “false”.
wrap_character	Specifies the character that is to be used to define the point in the text characters where a line wrap is to occur. The wrap character, at the wrap point, is removed from the text actually rendered. If omitted, the default is the space character (“ “).
wrap_width	Specifies the width, in pixels, before wrapping text to a new line. Text wrapping occurs at the first 'wrap_char' after the 'wrap_width. If omitted, the default is “0” (at every 'wrap_char').

Example

```
<Style name="Boundary">
```

Style Element

This element is used to define how a data source may be rendered. It is a collection of one (1) or more rules that are used to render data of the data source. There may be many Styles defined within the Map definition file, but they have to be referenced in a Layer Element before they are effective.

Rules are style elements which allow the data source to be filtered selectively and describe the manner in which the filtered data are to be rendered (symbolized).

General Form

```
<Style name="">
      <Rule name="" title="">
            ...
      </Rule>
</Style>
```

Attribute	Description
name	This is a string which uniquely identifies (associated with) this style. This name value is subsequently used in a StyleName Element within a Layer Element. It is a required attribute. Type is string.

Children

There is currently one (1) element that must be contained within this element:

- Rule

See description of Rule Element for details.

Example

```
<Style name="Boundary">
      <Rule>
            <PolygonSymbolizer>
                  <CssParameter name="fill">lightcyan</CssParameter>
            </PolygonSymbolizer>
            <LineSymbolizer>
                  <CssParameter name="stroke">black</CssParameter>
                  <CssParameter name="stroke-width">.1</CssParameter>
            </LineSymbolizer>
      </Rule>
</Style>
```

See the Appendix for a more detailed example and how all the elements are combined.

Appendix A Google Maps Tile Scale Denominators

Zoom Level	Scale Denominator	MaxScaleDenominator	MinScaleDenominator
0	559,082,264	750 000 000	540 000 000
1	279,541,132	500 000 000	270 000 000
2	139,770,566	250 000 000	130 000 000
3	69,885,283	100 000 000	60 000 000
4	34,942,642	50 000 000	30 000 000
5	17,471,321	25 000 000	15 000 000
6	8,735,660	10 000 000	8 000 000
7	4,367,830	5 000 000	4 000 000
8	2,183,915	2 500 000	2 000 000
9	1,091,958	1 500 000	1 000 000
10	545,979	750 000	540 000
11	272,989	500 000	270 000
12	136,495	250 000	130 000
13	68,247	100 000	60 000
14	34,124	50 000	30 000
15	17,062	25 000	15 000
16	8,531	10 000	8 000
17	4,265	5 000	4 000
18	2,133	2 500	2 000
19	1,066	1 500	1 000
20	533	750	520

Note: The Scale Denominator above is the 'computed value' for the corresponding zoom level. You may use any value between the desired level minimum value and the next higher levels value. The values in the remaining two columns are a bit arbitrary.

eg. Using:

```
<MaxScaleDemoninator>25000000</MaxScaleDenominator>
<MinScaleDenominator>8000000</MinScaleDenominator>
```

will result in the style Rule being rendered at Zoom level 5 and 6 tiles only

Appendix B Color Reference Chart

The following chart contains the Web safe HTML color codes:

#EEEEEE		#DDDDDD		#CCCCCC		#BBBBBB		#AAAAAA		#999999	
#888888		#777777		#666666		#555555		#444444		#333333	
#000000		#000033		#000066		#000099		#0000CC		#0000FF	
#330000		#330033		#330066		#330099		#3300CC		#3300FF	
#660000		#660033		#660066		#660099		#6600CC		#6600FF	
#990000		#990033		#990066		#990099		#9900CC		#9900FF	
#CC0000		#CC0033		#CC0066		#CC0099		#CC00CC		#CC00FF	
#FF0000		#FF0033		#FF0066		#FF0099		#FF00CC		#FF00FF	
#003300		#003333		#003366		#003399		#0033CC		#0033FF	
#333300		#333333		#333366		#333399		#3333CC		#3333FF	
#663300		#663333		#663366		#663399		#6633CC		#6633FF	
#993300		#993333		#993366		#993399		#9933CC		#9933FF	
#CC3300		#CC3333		#CC3366		#CC3399		#CC33CC		#CC33FF	
#FF3300		#FF3333		#FF3366		#FF3399		#FF33CC		#FF33FF	
#006600		#006633		#006666		#006699		#0066CC		#0066FF	
#336600		#336633		#336666		#336699		#3366CC		#3366FF	
#666600		#666633		#666666		#666699		#6666CC		#6666FF	
#996600		#996633		#996666		#996699		#9966CC		#9966FF	
#CC6600		#CC6633		#CC6666		#CC3399		#CC66CC		#CC66FF	
#FF6600		#FF6633		#FF6666		#FF6699		#FF66CC		#FF66FF	
#009900		#009933		#009966		#009999		#0099CC		#0099FF	
#339900		#339933		#339966		#339999		#3399CC		#3399FF	
#669900		#669933		#669966		#669999		#6699CC		#6699FF	
#999900		#999933		#999966		#999999		#9999CC		#9999FF	
#CC9900		#CC9933		#CC9966		#CC9999		#CC99CC		#CC99FF	
#FF9900		#FF9933		#FF9966		#FF9999		#FF99CC		#FF99FF	
#00CC00		#00CC33		#00CC66		#00CC99		#00CCCC		#00CCFF	
#33CC00		#33CC33		#33CC66		#33CC99		#33CCCC		#33CCFF	
#66CC00		#66CC33		#66CC66		#66CC99		#66CCCC		#66CCFF	
#99CC00		#99CC33		#99CC66		#99CC99		#99CCCC		#99CCFF	
#CCCC00		#CCCC33		#CCCC66		#CCCC99		#CCCCCC		#CCCCFF	
#FFCC00		#FFCC33		#FFCC66		#FFCC99		#FFCCCC		#FFCCFF	
#00FF00		#00FF33		#00FF66		#00FF99		#00FFCC		#00FFFF	
#33FF00		#33FF33		#33FF66		#33FF99		#33FFCC		#33FFFF	
#66FF00		#66FF33		#66FF66		#66FF99		#66FFCC		#66FFFF	
#99FF00		#99FF33		#99FF66		#99FF99		#99FFCC		#99FFFF	
#CCFF00		#CCFF33		#CCFF66		#CCFF99		#CCFFCC		#CCFFFF	
#FFFF00		#FFFF33		#FFFF66		#FFFF99		#FFFFCC		#FFFFFF	

The following chart shows the HTML named colors in alphabetical order.

aliceblue #F0F8FF	antiquewhite #FAEBD7	aqua #00FFFF	
aquamarine #7FFFD4	azure #F0FFFF	beige #F5F5DC	
bisque #FFE4C4	black #000000	blanchedalmond #FFEBCD	
blue #0000FF	blueviolet #8A2BE2	brown #A52A2A	
burlywood #DEB887	cadetblue #5F9EA0	chartreuse #7FFF00	
chocolate #D2691E	coral #FF7F50	cornflowerblue #6495ED	
cornsilk #FFF8DC	crimson #DC143C	cyan #00FFFF	
darkblue #00008B	darkcyan #008B8B	darkgoldenrod #B8860B	
darkgray #A9A9A9	darkgreen #006400	darkkhaki #BDB76B	
darkmagenta #8B008B	darkolivegreen #556B2F	darkorange #FF8C00	
darkorchid #9932CC	darkred #8B0000	darksalmon #E9967A	
darkseagreen #8FBC8F	darkslateblue #483D8B	darkslategray #2F4F4F	
darkturquoise #00CED1	darkviolet #9400D3	deeppink #FF1493	
deepskyblue #00BFFF	dimgray #696969	dodgerblue #1E90FF	
firebrick #B22222	floralwhite #FFFAF0	forestgreen #228B22	
fuchsia #FF00FF	gainsboro #DCDCDC	ghostwhite #F8F8FF	
gold #FFD700	goldenrod #DAA520	gray #808080	
green #008000	greenyellow #ADFF2F	honeydew #F0FFF0	
hotpink #FF69B4	indianred #CD5C5C	indigo #4B0082	
ivory #FFFFF0	khaki #F0E68C	lavender #E6E6FA	
lavenderblush #FFF0F5	lawngreen #7CFC00	lemonchiffon #FFFACD	
lightblue #ADD8E6	lightcoral #F08080	lightcyan #E0FFFF	
lightgoldenrodyellow #FAFAD2	lightgreen #90EE90	lightgrey #D3D3D3	

lightpink #FFB6C1	lightsalmon #FFA07A	lightseagreen #20B2AA	
lightskyblue #87CEFA	lightslategray #778899	lightsteelblue #B0C4DE	
lightyellow #FFFFE0	lime #00FF00	limegreen #32CD32	
linen #FAF0E6	magenta #FF00FF	maroon #800000	
mmediumaquamarine #66CDAA	mediumblue #0000CD	mediumorchid #BA55D3	
mediumpurple #9370D8	mediumseagreen #3CB371	mediumslateblue #7B68EE	
mediumspringgreen #00FA9A	mediumturquoise #48D1CC	mediumvioletred #C71585	
midnightblue #191970	mintcream #F5FFFA	mistyrose #FFE4E1	
moccasin #FFE4B5	navajowhite #FFDEAD	navy #000080	
oldlace #FDF5E6	olive #808000	olivedrab #688E23	
orange #FFA500	orangered #FF4500	orchid #DA70D6	
palegoldenrod #EEE8AA	palegreen #98FB98	paleturquoise #AFEEEE	
palevioletred #D87093	papayawhip #FFEFD5	peachpuff #FFDAB9	
peru #CD853F	pink #FFC0CB	plum #DDA0DD	
powderblue #B0E0E6	purple #800080	red #FF0000	
rosybrown #BC8F8F	royalblue #4169E1	saddlebrown #8B4513	
salmon #FA8072	sandybrown #F4A460	seagreen #2E8B57	
seashell #FFF5EE	sienna #A0522D	silver #C0C0C0	
skyblue #87CEEB	slateblue #6A5ACD	slategray #708090	
snow #FFFAFA	springgreen #00FF7F	steelblue #4682B4	
tan #D2B48C	teal #008080	thistle #D8BFD8	
tomato #FF6347	turquoise #40E0D0	violet #EE82EE	
wheat #F5DEB3	white #FFFFFF	whitesmoke #F5F5F5	
yellow #FFFF00	yellowGreen #9ACD32		

Appendix C DejaVu Font Style

Mapnik supplies a default set of fonts with the installation. They are placed in a mapnik specific location dependant on the system architecture that you are using:

For Linux/Unix the default is: /usr/local/lib or /usr/local/lib64
For Windows the default is:

The full DejaVu fonts family contains following **styles**:

- Sans: Book, Bold, Oblique, Bold Oblique, Condensed, Condensed Bold, Condensed Oblique and Condensed Bold Oblique, Extralight
- Serif: Book, Bold, Italic, Bold Italic, Condensed, Condensed Bold, Condensed Italic and Condensed Bold Italic
- Mono: Book, Bold, Oblique, Bold Oblique

See http://dejavu-fonts.org for a full description.

Examples:

For Sans Bold, use “DejaVu Sans Bold” for the 'face_name'
For Serif Bold, use “DejaVu Serif Bold” for the 'face_name'
For Mono Bold, use “DejaVu Mono Bold” for the 'face_name'

Alternative Approach to determine Available Styles

Using the Python console do the following:

```
>>> from mapnik import *
>>> for fname in FontEngine.face_names():
...     print fname
...
DejaVu Sans Bold
DejaVu Sans Bold Oblique
DejaVu Sans Book
```

and it will print out a complete list of 'face_name's that you can use.