landez Documentation

Release latest

December 16, 2014

Contents

Landez manipulates tiles, builds MBTiles, does tiles compositing and arrange tiles together into single images.

Tiles can either be obtained from a remote tile service URL, from a local Mapnik stylesheet, a WMS server or from MBTiles files.

For building MBTiles, Landez embeds *mbutil* from Mapbox https://github.com/mapbox/mbutil at the final stage. The land covered is specified using a list of bounding boxes and zoom levels.

Landez is pure python and has no external dependency.

```
sudo easy_install landez
```

However, it requires *mapnik* if the tiles are rendered locally.

```
sudo aptitude install python-mapnik
```

And *PIL* to blend tiles together or export arranged tiles into images.

```
sudo aptitude install python-imaging
```

CHAPTER 1

USAGE

1.1 Building MBTiles files

1.1.1 Remote tiles

Using a remote tile service (OpenStreetMap.org by default):

```
import logging
from landez import MBTilesBuilder

logging.basicConfig(level=logging.DEBUG)

mb = MBTilesBuilder(cache=False)
mb.add_coverage(bbox=(-180.0, -90.0, 180.0, 90.0),
                zoomlevels=[0, 1])
mb.run()
```

Please respect *Tile usage policies <http://wiki.openstreetmap.org/wiki/Tile_usage_policy>*

1.1.2 Local rendering

Using mapnik to render tiles:

```
import logging
from landez import MBTilesBuilder

logging.basicConfig(level=logging.DEBUG)

mb = MBTilesBuilder(stylefile="yourstyle.xml", filepath="dest.mbtiles")
mb.add_coverage(bbox=(-180.0, -90.0, 180.0, 90.0),
                zoomlevels=[0, 1])
mb.run()
```

And with UTFGrids:

```
import logging
from landez import MBTilesBuilder

logging.basicConfig(level=logging.DEBUG)

mb = MBTilesBuilder(stylefile="yourstyle.xml",
```

```
                    grid_fields=["field1", "field2", "field3", ...] ,
                    filepath="dest.mbtiles")
mb.add_coverage(bbox=(-180, -90, 180, 90),
                zoomlevels=[0, 1, 2, 3])
mb.run()
```

1.1.3 From an other MBTiles file

```
import logging
from landez import MBTilesBuilder

logging.basicConfig(level=logging.DEBUG)

mb = MBTilesBuilder(mbtiles_file="yourfile.mbtiles", filepath="dest.mbtiles")
mb.add_coverage(bbox=(-180.0, -90.0, 180.0, 90.0),
                zoomlevels=[0, 1])
mb.run()
```

1.1.4 From a WMS server

```
mb = MBTilesBuilder(wms_server="http://yourserver.com/geoserver/wms",
                    wms_layers=["ign:departements"],
                    wms_options=dict(format="image/png",
                                     transparent=True),
                    filepath="dest.mbtiles")
mb.add_coverage(bbox=([-0.9853,43.6435.1126,44.0639]))
mb.run()
```

1.2 Blend tiles together

Merge multiple sources of tiles (URL, WMS, MBTiles, Mapnik stylesheet) together. *(requires python PIL)*

For example, build a new MBTiles by blending tiles of a MBTiles on top of OpenStreetMap tiles :

```
mb = MBTilesBuilder(filepath="merged.mbtiles")
overlay = TilesManager(mbtiles_file="carto.mbtiles")
mb.add_layer(overlay)
mb.run()
```

Or composite a WMS layer with OpenStreetMap using transparency (40%):

```
mb = MBTilesBuilder(wms_server="http://yourserver.com/geoserver/wms",
                    wms_layers=["img:orthophoto"])
overlay = TilesManager(remote=True)
mb.add_layer(overlay, 0.4)
mb.run()
```

1.3 Export Images

Assemble and arrange tiles together into a single image. *(requires python PIL)*

Specify tiles sources in the exact same way as for building MBTiles files.

```
import logging
from landez import ImageExporter

logging.basicConfig(level=logging.DEBUG)

ie = ImageExporter(mbtiles_file="yourfile.mbtiles")
ie.export_image(bbox=(-180.0, -90.0, 180.0, 90.0), zoomlevel=3, imagepath="image.png")
```

1.4 Add post-processing filters

Convert map tiles to gray scale, more suitable for information overlay :

```
from landez.filters import GrayScale

ie = ImageExporter()
ie.add_filter(GrayScale())
```

Replace a specific color by transparent pixels (i.e. color to alpha, *a-la-Gimp*) :

```
from landez.filters import ColorToAlpha

overlay = TileManager()
overlay.add_filter(ColorToAlpha('#ffffff'))  # white will be transparent

ie = ImageExporter()
ie.add_layer(overlay)
...
```

1.5 Extract MBTiles content

```
from landez.sources import MBTilesReader

mbreader = MBTilesReader("yourfile.mbtiles")

# Metadata
print mbreader.metadata()

# Zoom levels
print mbreader.zoomlevels()

# Image tile
with open('tile.png', 'wb') as out:
    out.write(mbreader.tile(z, x, y))

# UTF-Grid tile
print mbreader.grid(z, x, y, 'callback')
```

1.6 Manipulate tiles

```
from landez import MBTilesBuilder

# From a TMS tile server
# tm = TilesManager(tiles_url="http://{s}.tile.openstreetmap.org/{z}/{x}/{y}.png")

# From a MBTiles file
tm = TilesManager(mbtiles_file="yourfile.mbtiles")

tiles = tm.tileslist(bbox=(-180.0, -90.0, 180.0, 90.0),
                     zoomlevels=[0, 1])
for tile in tiles:
    tilecontent = tm.tile(tile)  # download, extract or take from cache
    ...
```

Cache tiles are stored using TMS scheme by default (with `y` value flipped). It can be changed to WMTS (a.k.a `xyz`) :

```
tm = TilesManager(your_sources_options, cache=True, cache_scheme="wmts")
```

1.7 Run tests

Run tests with nosetests (if you are working in a virtualenv, don't forget to install nose in it!):

```
cd landez
nosetests
```

The Mapnik stylesheet for the test about grid content comes from <https://github.com/springmeyer/gridsforkids>

CHAPTER 2

AUTHORS

- Mathieu Leplatre <mathieu.leplatre@makina-corpus.com>
- Sergej Tatarincev
- Éric Bréhault
- Waldemar Osuch
- Isabelle Vallet
- Thanks to mbutil authors <https://github.com/mapbox/mbutil>

CHAPTER 3

LICENSE

- Lesser GNU Public License