Usage Instructions for gdal2tiles\_parallel.py

stevendlander edited this page on 16 Dec 2014 · [1 revision](https://github.com/GitHubRGI/geopackage-python/wiki/Usage-Instructions-for-gdal2tiles_parallel.py/_history)

**Purpose**

This document covers installation of dependencies, in-depth explanations of command-line arguments, and usage examples for gdal2tiles parallel.py. The file gdal2tiles parallel.py is a Python script that converts GDAL-supported raster imagery files into a folder of tiles in TMS format (z/x/y). This version improves upon the standard gdal2tiles.py file included with GDAL by adding multiprocessing improvements to obtain even better tile generation performance.

**Installing Dependencies**

Gdal2tiles parallel.py was written for Python 2.7.x and has not been tested on any other version. It will run on either 32 or 64 bit systems with no issues. The script relies on Python 2.7.x, GDAL core, and the GDAL Python bindings. Some imagery formats such as MrSID will require additional driver installers to work correctly. Please see the [wiki page](https://github.com/GitHubRGI/geopackage-python/wiki) for installing dependencies for more information.

**Usage**

**Command Line Arguments**

Gdal2tiles parallel.py supports additional functionality via command line arguments provided to the script at the time it is executed. Following is a outline of the important flags:

-h, -help Print the listing of commands available for the script.

-p, –profile Specify the tiling profile you would like these tiles to be created in. Valid options are

mercator or geodetic.

-e, -resume Instruct the script to not overwrite tiles that have already been created. This is a

\*mandatory\* flag when using default multiprocessing.

-z, -zoom The zoom levels to create. Allows the tiler to make tiles past the default zoom level that

GDAL detects. (Format: ’2-5’ or ’10’)

-a, -srcnodata Specify the RGB value that gdal2tiles should convert to transparency. Typical value should

be ’0,0,0’.

**Examples**

* Create a folder of tiles in the mercator projection based on a GeoTiff image named WhiteHorse.tif and name the folder ’WhiteHorse tiles’:
* python gdal2tiles\_parallel.py -e -p mercator /data/raw/WhiteHorse.tif /data/tiles/mercator/WhiteHorse\_tiles
* Create a folder of tiles for zoom level 15 in the mercator projection based on a GeoTiff image named WhiteHorse.tif and name the folder ’WhiteHorse tiles’:
* python gdal2tiles\_parallel.py -p mercator -e -z 15 /data/raw/WhiteHorse.tif /data/tiles/mercator/WhiteHorse\_tiles
* Create a folder of tiles in the geodetic projection based on a MrSID image named FortBelvoir 201307 A6.sid and name the folder belvoir tiles. Also, assigns the NODATA transparency to the RGB color value of 0,0,0:
* python gdal2tiles\_parallel.py -p geodetic -e -a 0,0,0 /data/raw/FortBelvoir\_201307\_A6.sid /data/tiles/belvoir\_tiles
* Create a folder of tiles in the geodetic projection based on a MrSID image named FortBelvoir 201307 A6.sid and name the folder belvoir tiles. Also, assigns the NODATA transparency to the RGB color value of 255,0,0:
* python gdal2tiles\_parallel.py -p geodetic -e -a 255,0,0 /data/raw/FortBelvoir\_201307\_A6.sid /data/tiles/belvoir\_tiles
* Create a folder of tiles for zoom levels 10 through 13 in the geodetic projection based on a MrSID image named FortBelvoir 201307 A6.sid and name the folder belvoir tiles. Also, assigns the NODATA transparency to the RGB color value of 0,0,0:
* python gdal2tiles\_parallel.py -p geodetic -e -a 0,0,0 -z 10-13 /data/raw/FortBelvoir\_201307\_A6.sid /data/tiles/belvoir\_tiles

**Known Issues**

Refer to the issues on this repository for an up-to-date listing of known issues and their status.

**Pages 7**



* [**Home**](https://github.com/GitHubRGI/geopackage-python/wiki)
* [**Installing dependencies on Linux**](https://github.com/GitHubRGI/geopackage-python/wiki/Installing-dependencies-on-Linux)
* [**Installing dependencies on Windows**](https://github.com/GitHubRGI/geopackage-python/wiki/Installing-dependencies-on-Windows)
* [**Profiling Results of Tiling and Packaging Features in GeoPackage Python**](https://github.com/GitHubRGI/geopackage-python/wiki/Profiling-Results-of-Tiling-and-Packaging-Features-in-GeoPackage-Python)
* [**Running Unit Tests On tiles2gpkg\_parallel.py**](https://github.com/GitHubRGI/geopackage-python/wiki/Running-Unit-Tests-On-tiles2gpkg_parallel.py)
* [**Usage Instructions for gdal2tiles\_parallel.py**](https://github.com/GitHubRGI/geopackage-python/wiki/Usage-Instructions-for-gdal2tiles_parallel.py)
* [**Usage Instructions for tiles2gpkg\_parallel.py**](https://github.com/GitHubRGI/geopackage-python/wiki/Usage-Instructions-for-tiles2gpkg_parallel.py)

**Clone this wiki locally**

Usage Instructions for tiles2gpkg\_parallel.py

stevendlander edited this page on 16 Dec 2014 · [1 revision](https://github.com/GitHubRGI/geopackage-python/wiki/Usage-Instructions-for-tiles2gpkg_parallel.py/_history)

**Purpose**

This document covers installation of dependencies, in-depth explanations of command-line arguments, and usage examples for tiles2gpkg parallel.py. The file tiles2gpkg parallel.py is a Python script that accepts a folder of tiles in TMS format (z/x/y) and outputs an Open Geospatial Consortium (OGC)- compliant Geopackage. The script leverages as much hardware capability as possible in order package this data. For more information about the OGC Specification for Geopackage, please visit the Open Geospatial Consortium Website: <http://www.opengeospatial.org/standards/geopackage>.

**Installing Dependencies**

Tiles2gpkg parallel.py was written for Python 2.7.x and has not been tested on any other version. It will run on either 32 or 64 bit systems with no issues. The script relies on the Python Imaging Library (PIL) in order to allow fine-grain control of the MIME type (PNG/JPEG) of each individual tile that resides within the output Geopackage. Without PIL, the script will simply detect the image type of the input tile and maintain that MIME type when it is stored to the Geopackage. Please see the [wiki page](https://github.com/GitHubRGI/geopackage-python/wiki) for installing dependencies for more information.

**Usage**

**Command Line Arguments**

Tiles2gpkg parallel.py supports additional functionality via command line arguments provided to the script at the time it is executed. Following is a outline of each one and their purpose: -h Print the listing of commands available for this script. -tileorigin Specify the origin of the tiles contained within the input data folder. Gdal2tiles.py creates tiles referenced by the bottom-left corner by default, which follows TMS convention. Other tile providers can create tiles with a tile origin of upper-left. Valid options are ul, ll, nw, or sw. The default option is ll for lower-left. -srs Specify the spatial reference system of the tiles contained within the input data folder. This could also be called the tile grid profile. Valid options are 3857 (mercator), 4326 (geodetic), and 3395 (global mercator). The default value for this field is 3857.

-imagery Convert the MIME type of the tiles on-disk to a new type when they are stroed in the

GeoPackage. Valid options are source, mixed, png, and jpeg. Specifying mixed mode will convert

all tile images in the source folderthat do \*not\* have transparency to JPEG with compression

enabled for space savings. Specifying source mode will preserve the file type of the input

tile images. The default value for this field is source.

-q When the -imagery flag is set to either mixed or jpeg, this flag specifies the jpeg quality

value. Acceptable values are from 1-100 inclusive. Lower numbers result in smaller size images

but greatly reduced image quality.

-T By default, tiles2gpkg\_parallel.py takes advantage of all the processors available to the

hardware that it is executed on. This can mean that other computing tasks on the machine may

suffer depending on the size of the packaging job. The -T flag disables this behavoir and only

uses a single-core process to execute the job.

**Examples**

* Create a geopackage from a folder of tiles named WhiteHorse in the geodetic tile profile, and name the new Geopackage whitehorse.gpkg:
* python tiles2gpkg\_parallel.py -srs 4326 /data/tiles/geodetic/WhiteHorse /data/geopackage/whitehorse.gpkg
* Create a geopackage from a folder of tiles named belvoir in the mercator profile and changing the tile images to a mix of PNG and JPEG images:
* python tiles2gpkg\_parallel.py -srs 3857 -imagery mixed /data/tiles/mercator/belvoir /data/geopackage/belvoir-3857.gpkg
* Create a geopackage from a folder of tiles named gnc in the world mercator profile with a tile origin of upper-left and also converting the tile images to JPEG with 50% quality:
* python tiles2gpkg\_parallel.py -srs 3395 -tileorigin ul -imagery jpeg -q 50 /data/tiles/world-mercator/gnc /data/geopackage/gnc-wm.gpkg

**Known Issues**

Refer to the issues on this repository for an up-to-date listing of known issues and their status.