Landsat-util Documentation

Release 0.7.0

Development Seed

May 29, 2015

Contents

1 Installation **3**
- 1.1 Mac OSX . . . 3
- 1.2 Ubuntu 14.10 . . . 3
- 1.3 Other systems . . . 3
- 1.4 Docker . . . 3
- 1.5 Upgrade . . . 4
- 1.6 Running Tests . . . 4

2 Overview: What can landsat-util do? **5**
- 2.1 Search . . . 5
- 2.2 Download . . . 5
- 2.3 Image processing . . . 6

3 To Do List **7**

4 Module Index **9**
- 4.1 downloader.py . . . 9
- 4.2 uploader.py . . . 11
- 4.3 image.py . . . 11
- 4.4 landsat.py . . . 11
- 4.5 mixins.py . . . 11
- 4.6 search.py . . . 11
- 4.7 utils.py . . . 13

5 Important Notes **17**

Python Module Index **19**

Landsat-util is a command line utility that makes it easy to search, download, and process Landsat imagery.

This tool uses Development Seed's API for Landsat Metadata.

This API is accessible here: https://api.developmentseed.org/landsat

You can also run your own API and connect it to this tool.

Table of Contents:

CHAPTER 1

Installation

1.1 Mac OSX

```
$: pip install landsat-util
```

1.2 Ubuntu 14.10

Use pip to install landsat-util. If you are not using virtualenv, you might have to run `pip` as `sudo`:

```
$: sudo apt-get update
$: sudo apt-get install python-pip python-numpy python-scipy libgdal-dev libatlas-base-dev gfortran l
$: pip install landsat-util
```

1.3 Other systems

Make sure Python setuptools is installed:

```
$: python setup.py numpy six
$: python setup.py install
```

1.4 Docker

If you have docker installed, you can use landsat-util image on docker:

```
$: docker pull developmentseed/landsat-util
$: docker run -it developmentseed/landsat-util:latest /bin/sh -c "landsat -h"
```

To use docker version run:

```
$: docker run -it -v ~/landsat:/root/landsat developmentseed/landsat-util:latest landsat -h
```

Example commands:

```
$: docker run -it -v ~/landsat:/root/landsat developmentseed/landsat-util:latest landsat search --clo
$: docker run -it -v ~/landsat:/root/landsat developmentseed/landsat-util:latest landsat download LC8
```

This commands mounts `landsat` folder in your home directory to `/root/landsat` in docker. All downloaded and processed images are stored in `~/landsat` folder of your computer.

If you are using Windows replace `~/landsat` with `/c/Users/<path>`.

1.5 Upgrade

```
$: pip install -U landsat-util
```

If you have installed previous version of landsat using brew, first run:

```
$: brew uninstall landsat-util
```

1.6 Running Tests

```
$: pip install -U requirements/dev.txt
$: nosetests
```

Or:

```
$: python setup.py test
```

CHAPTER 2

Overview: What can landsat-util do?

Landsat-util has three main functions:

- **Search** for landsat tiles based on several search parameters.
- **Download** landsat images.
- **Image processing** and pan sharpening on landsat images.

These three functions have to be performed separately.

Help: Type `landsat -h` for detailed usage parameters.

2.1 Search

Search returns information about all landsat tiles that match your criteria. This includes a link to an unprocessed preview of the tile. The most important result is the tile's *sceneID*, which you will need to download the tile (see step 2 below).

Search for landsat tiles in a given geographical region, using any of the following:

- **Paths and rows**: If you know the paths and rows you want to search for.
- **Latidue and Longitude**: If you need the latitude and longitude of the point you want to search for.

Additionally filter your search using the following parameters:

- **Start and end dates** for when imagery was taken
- **Maximum percent cloud cover** (default is 20%)

Examples of search:

Search by path and row:

```
$: landsat search --cloud 4 --start "january 1 2014" --end "january 10 2014" -p 009,045
```

Search by latitude and longitude:

```
$: landsat search --lat 38.9004204 --lon -77.0237117
```

2.2 Download

You can download tiles using their unique sceneID, which you get from landsat search.

Landsat-util will download a zip file that includes all the bands. You have the option of specifying the bands you want to down. In this case, landsat-util only downloads those bands if they are available online.

Examples of download:

Download images by their custom sceneID, which you get from landsat search:

```
$: landsat download LC80090452014008LGN00
```

Download only band 4, 3 and 2 for a particular sceneID:

```
$: landsat download LC80090452014008LGN00 --bands 432
```

Download multiple sceneIDs:

```
$: landsat download LC80090452014008LGN00 LC80090452015008LGN00 LC80090452013008LGN00
```

2.3 Image processing

You can process your downloaded tiles with our custom image processing algorithms. In addition, you can choose to pansharpen your images and specify which bands to process.

Examples of image processing:

Process images that are already downloaded. Remember, the program accepts both zip files and unzipped folders:

```
$: landsat process path/to/LC80090452014008LGN00.tar.bz
```

If unzipped:

```
$: landsat process path/to/LC80090452014008LGN00
```

Specify bands 3, 5 and 1:

```
$: landsat process path/to/LC80090452014008LGN00  --bands 351
```

Process *and* pansharpen a downloaded image:

```
$: landsat process path/to/LC80090452014008LGN00.tar.bz --pansharpen
```

CHAPTER 3

To Do List

- Add Sphinx Documentation
- Add capacity for NDVI output
- Add alternative projections (currently only option is default web-mercator; EPSG: 3857)
- Connect search to Google Address API
- Include 16-bit image variant in output
- Add support for color correct looping over multiple compressed inputs (currently just 1)

CHAPTER 4

Module Index

4.1 downloader.py

class `landsat.downloader.`**`Downloader`**(*verbose=False*, *download_dir=None*)

Bases: `landsat.mixins.VerbosityMixin`

The downloader class

`amazon_s3`(*scene*, *band*, *path*)

Amazon S3 downloader

Parameters

- **`scene`** (*String*) – The scene ID.
- **`band`** (*String, Integer*) – The band number.
- **`path`** (*String*) – The directory path to where the image should be stored

Returns Boolean

`amazon_s3_url`(*sat*, *filename*)

Return an amazon s3 url the contains the scene and band provided.

Parameters

- **`sat`** (*dict*) – Expects an object created by scene_interpreter method
- **`filename`** (*String*) – The filename that has to be downloaded from Amazon

Returns (String) The URL to a S3 file

`download`(*scenes*, *bands=None*)

Download scenese from Google Storage or Amazon S3 if bands are provided

Parameters

- **`scenes`** (*List*) – A list of scene IDs
- **`bands`** – A list of bands. Default value is None.

Returns (List) includes downloaded scenes as key and source as value (aws or google)

`fetch`(*url*, *path*, *filename*)

Downloads the given url.

Parameters

- **`url`** (*String*) – The url to be downloaded.

- **`path`** (*String*) – The directory path to where the image should be stored
- **`filename`** (*String*) – The filename that has to be downloaded

Returns Boolean

`get_remote_file_size`(*url*)
Gets the filesize of a remote file.

Parameters **`url`** (*String*) – The url that has to be checked.

Returns int

`google_storage`(*scene*, *path*)
Google Storage Downloader.

Parameters

- **`scene`** (*String*) – The scene id
- **`path`** (*String*) – The directory path to where the image should be stored

Returns Boolean

`google_storage_url`(*sat*)
Returns a google storage url the contains the scene provided.

Parameters **`sat`** (*dict*) – Expects an object created by scene_interpreter method

Returns (String) The URL to a google storage file

`remote_file_exists`(*url*)
Checks whether the remote file exists.

Parameters **`url`** (*String*) – The url that has to be checked.

Returns **True** if remote file exists and **False** if it doesn't exist.

`scene_interpreter`(*scene*)
Conver sceneID to rows, paths and dates.

Parameters **`scene`** (*String*) – The scene ID.

Returns dict

Example output

```
>>> anatomy = {
        'path': None,
        'row': None,
        'sat': None,
        'scene': scene
    }
```

exception `landsat.downloader.`**`IncorrectSceneId`**
Bases: `exceptions.Exception`

Exception to be used when scene id is incorrect

exception `landsat.downloader.`**`RemoteFileDoesntExist`**
Bases: `exceptions.Exception`

Exception to be used when the remote file does not exist

4.2 uploader.py

4.3 image.py

4.4 landsat.py

4.5 mixins.py

class `landsat.mixins.`**`VerbosityMixin`**

Bases: `object`

Verbosity Mixin that generates beautiful stdout outputs.

`exit`(*message*)

outputs an exit message and exits

Parameters **`message`** (*String*) – The message to be outputed

Returns void

`output`(*value*, *normal=False*, *color=None*, *error=False*, *arrow=False*, *indent=None*)

Handles verbosity of this calls. if priority is set to 1, the value is printed

if class instance verbose is True, the value is printed

Parameters

- **`value`** (*String*) – a string representing the message to be printed
- **`normal`** (*String*) – if set to true the message is always printed, otherwise it is only shown if verbosity is set
- **`color`** – The color of the message, choices: ‘red’, ‘green’, ‘blue’
- **`error`** (*Boolean*) – if set to true the message appears in red
- **`arrow`** (*Boolean*) – if set to true an arrow appears before the message
- **`indent`** (*Boolean*) – indents the message based on the number provided

Returns void

`subprocess`(*argv*)

Execute subprocess commands with proper ouput. This is no longer used in landsat-util

Parameters **`argv`** (*List*) – A list of subprocess arguments

Returns void

`verbose` **= False**

4.6 search.py

class `landsat.search.`**`Search`**

Bases: `object`

The search class

cloud_cover_prct_range_builder(*min=0*, *max=100*)
Builds cloud cover percentage range query.

Parameters

- **min** (*float*) – float specifying the minimum percentage. Default is 0
- **max** (*float*) – float specifying the maximum percentage. Default is 100

Returns String

date_range_builder(*start='2013-02-11'*, *end=None*)
Builds date range query.

Parameters

- **start** (*String*) – Date string. format: YYYY-MM-DD
- **end** (*String*) – date string. format: YYYY-MM-DD

Returns String

lat_lon_builder(*lat=0*, *lon=0*)
Builds lat and lon query.

Parameters

- **lat** (*float*) – The latitude. Default is 0
- **lon** (*float*) – The The longitude. Default is 0

Returns String

query_builder(*paths_rows=None*, *lat=None*, *lon=None*, *start_date=None*, *end_date=None*, *cloud_min=None*, *cloud_max=None*)
Builds the proper search syntax (query) for Landsat API.

Parameters

- **paths_rows** (*String*) – A string in this format: "003,003,004,004". Must be in pairs and separated by comma.
- **lat** (*String, float, integer*) – The latitude
- **lon** (*String, float, integer*) – The The longitude
- **start_date** (*String*) – Date string. format: YYYY-MM-DD
- **end_date** (*String*) – date string. format: YYYY-MM-DD
- **cloud_min** (*float*) – float specifying the minimum percentage. e.g. 4.3
- **cloud_max** (*float*) – float specifying the maximum percentage. e.g. 78.9

Returns String

row_path_builder(*path=''*, *row=''*)
Builds row and path query.

Parameters

- **path** (*String*) – Landsat path. Must be three digits
- **row** (*String*) – Landsat row. Must be three digits

Returns String

search (*paths_rows=None*, *lat=None*, *lon=None*, *start_date=None*, *end_date=None*, *cloud_min=None*, *cloud_max=None*, *limit=1*)

The main method of Search class. It searches Development Seed's Landsat API.

Parameters

- **paths_rows** (*String*) – A string in this format: "003,003,004,004". Must be in pairs and separated by comma.
- **lat** (*String, float, integer*) – The latitude
- **lon** (*String, float, integer*) – The The longitude
- **start_date** (*String*) – Date string. format: YYYY-MM-DD
- **end_date** (*String*) – date string. format: YYYY-MM-DD
- **cloud_min** (*float*) – float specifying the minimum percentage. e.g. 4.3
- **cloud_max** (*float*) – float specifying the maximum percentage. e.g. 78.9
- **limit** (*integer*) – integer specigying the maximum results return.

Returns dict

Example

```
>>> search = Search()
>>> search('003,003', '2014-01-01', '2014-06-01')
>>> {
        'status': u'SUCCESS',
        'total_returned': 1,
        'total': 1,
        'limit': 1
        'results': [
            {
                'sat_type': u'L8',
                'sceneID': u'LC80030032014142LGN00',
                'date': u'2014-05-22',
                'path': u'003',
                'thumbnail': u'http://....../landsat_8/2014/003/003/LC80030032014142L
                'cloud': 33.36,
                'row': u'003
            }
        ]
    }
```

4.7 utils.py

class landsat.utils.**Capturing**

Bases: list

Captures a subprocess stdout.

Usage

```
>>> with Capturing():
...     subprocess(args)
```

`landsat.utils.`**`check_create_folder`**(*folder_path*)
Check whether a folder exists, if not the folder is created.

Parameters **`folder_path`** (*String*) – Path to the folder

Returns (String) the path to the folder

`landsat.utils.`**`convert_to_integer_list`**(*value*)
Converts a comma separate string to a list

Parameters **`value`** (*String*) – the format must be 003,003,004,004 (commas with no space)

Returns List

Example

```
>>> convert_to_integer_list('003,003,004,004')
['003', '003', '004', '004']
```

`landsat.utils.`**`create_paired_list`**(*value*)
Create a list of paired items from a string.

Parameters **`value`** (*String*) – the format must be 003,003,004,004 (commas with no space)

Returns List

Example

```
>>> create_paired_list('003,003,004,004')
[['003','003'], ['004', '004']]
```

`landsat.utils.`**`exit`**(*message*, *code=0*)
output a message to stdout and terminates the process.

Parameters

- **`message`** (*String*) – Message to be outputed.
- **`code`** (*int*) – The termination code. Default is 0

Returns void

`landsat.utils.`**`georgian_day`**(*date*)
Returns the number of days passed since the start of the year.

Parameters **`date`** (*String*) – The string date with this format %m/%d/%Y

Returns int

Example

```
>>> georgian_day('05/1/2015')
121
```

`landsat.utils.`**`get_file`**(*path*)
Separate the name of the file or folder from the path and return it.

Parameters **`path`** (*String*) – Path to the folder

Returns (String) the filename

Example

```
>>> get_file('/path/to/file.jpg')
'file.jpg'
```

`landsat.utils.`**`get_filename`**(*path*)

Return the filename without extension.

Parameters **`path`** (*String*) – Path to the folder

Returns (String) the filename without extension

Example

```
>>> get_filename('/path/to/file.jpg')
'file'
```

`landsat.utils.`**`reformat_date`**(*date*, *new_fmt='%Y-%m-%d'*)

Returns reformatted date.

Parameters

- **`date`** (*String*) – The string date with this format %m/%d/%Y
- **`new_fmt`** – date format string. Default is '%Y-%m-%d'

Returns int

Example

```
>>> reformat_date('05/1/2015', '%d/%m/%Y')
'1/05/2015'
```

`landsat.utils.`**`three_digit`**(*number*)

Add 0s to inputs that their length is less than 3.

Parameters **`number`** (*int*) – The number to convert

Returns String

Example

```
>>> three_digit(1)
'001'
```

class `landsat.utils.`**`timer`**

Bases: `object`

A timer class.

Usage

```
>>> with timer():
...     your code
```

`landsat.utils.`**`year`**(*date*)

Returns the year.

Parameters **`date`** (*String*) – The string date with this format %m/%d/%Y

Returns int

Example

```
>>> year('05/1/2015')
2015
```

Important Notes

- All downloaded and processed images are stored at your home directory in landsat forlder: `~/landsat`
- The image thumbnail web address that is included in the results can be used to make sure that clouds are not obscuring the subject of interest. Run the search again if you need to narrow down your result and then start downloading images. Each image is usually more than 700mb and it might takes a very long time if there are too many images to download
- Image processing is a very heavy and resource consuming task. Each process takes about 5-10 mins. We recommend that you run the processes in smaller badges. Pansharpening, while increasing image resolution 2x, substantially increases processing time.
- Landsat-util requires at least 2GB of Memory (RAM).

Python Module Index

l

landsat.downloader, 9
landsat.mixins, 11
landsat.search, 11
landsat.utils, 13

A

amazon_s3() (landsat.downloader.Downloader method), 9
amazon_s3_url() (landsat.downloader.Downloader method), 9

C

Capturing (class in landsat.utils), 13
check_create_folder() (in module landsat.utils), 13
cloud_cover_prct_range_builder() (landsat.search.Search method), 11
convert_to_integer_list() (in module landsat.utils), 14
create_paired_list() (in module landsat.utils), 14

D

date_range_builder() (landsat.search.Search method), 12
download() (landsat.downloader.Downloader method), 9
Downloader (class in landsat.downloader), 9

E

exit() (in module landsat.utils), 14
exit() (landsat.mixins.VerbosityMixin method), 11

F

fetch() (landsat.downloader.Downloader method), 9

G

georgian_day() (in module landsat.utils), 14
get_file() (in module landsat.utils), 14
get_filename() (in module landsat.utils), 15
get_remote_file_size() (landsat.downloader.Downloader method), 10
google_storage() (landsat.downloader.Downloader method), 10
google_storage_url() (landsat.downloader.Downloader method), 10

I

IncorrectSceneId, 10

L

landsat.downloader (module), 9
landsat.mixins (module), 11
landsat.search (module), 11
landsat.utils (module), 13
lat_lon_builder() (landsat.search.Search method), 12

O

output() (landsat.mixins.VerbosityMixin method), 11

Q

query_builder() (landsat.search.Search method), 12

R

reformat_date() (in module landsat.utils), 15
remote_file_exists() (landsat.downloader.Downloader method), 10
RemoteFileDoesntExist, 10
row_path_builder() (landsat.search.Search method), 12

S

scene_interpreter() (landsat.downloader.Downloader method), 10
Search (class in landsat.search), 11
search() (landsat.search.Search method), 12
subprocess() (landsat.mixins.VerbosityMixin method), 11

T

three_digit() (in module landsat.utils), 15
timer (class in landsat.utils), 15

V

verbose (landsat.mixins.VerbosityMixin attribute), 11
VerbosityMixin (class in landsat.mixins), 11

Y

year() (in module landsat.utils), 15