
urlfix

Release 0.2.3

Nelson Gonzabato

Aug 06, 2022

CONTENTS

1	urlfix: Check and Fix Outdated URLs	1
2	urlfix	5
2.1	urlfix package	5
3	Welcome to urlfix’s changelog	7
4	Contributing to urlfix	9
5	Indices and tables	11
	Python Module Index	13
	Index	15

URLFIX: CHECK AND FIX OUTDATED URLS

`urlfix` aims to find all outdated URLs in a given file and fix them.

Features List

- [x] Commandline and programmer-friendly modes.
- [x] Replace outdated URLs/links in a single file
- [x] Replace outdated URLs/links in a directory
- [x] Replace outdated URLs/links in the same file or in the same files in a directory i.e. inplace.
- [x] Replace outdated links in files in nested directories
- [x] Replace outdated links in files in sub-nested directories

Supported file formats

`urlfix` fixes URLs given a file of the following types:

- [x] Markdown (.md)
- [x] Plain Text files (.txt)
- [x] RMarkdown (.rmd)
- [x] ReStructured Text (.rst)

- [] PDF (.pdf)
- [] Word (.docx)
- [] ODF (.odf)

Installation

The simplest way to install the latest release is as follows:

```
pip install urlfix
```

To install the development version:

Open the Terminal/CMD/Git bash/shell and enter

```
pip install git+https://github.com/Nelson-Gon/urlfix.git

# or for the less stable dev version
pip install git+https://github.com/Nelson-Gon/urlfix.git@dev
```

Otherwise:

```
# clone the repo
git clone git@github.com:Nelson-Gon/urlfix.git
cd urlfix
pip install -e .
```

Sample usage

Script Mode

To use at the commandline, please use:

```
python -m urlfix --mode "f" --verbose 1 --inplace 1 --inpath myfile.md
```

If not replacing within the same file, then:

```
python -m urlfix --mode "f" --verbose 1 --inplace 0 --inpath myfile.md --output-file_
↪myoutputfile.md
```

To get help:

```
python -m urlfix -h

#usage: main.py [-h] -m MODE -in INPUT_FILE [-o OUTPUT_FILE] -v {False,false,0,True,true,
↪1} -i {False,false,0,True,true,1}
#
#optional arguments:
# -h, --help            show this help message and exit
# -m MODE, --mode MODE  Mode to use. One of f for file or d for directory
# -in INPUT_FILE, --input-file INPUT_FILE
#                        Input file for which link updates are required.
# -o OUTPUT_FILE, --output-file OUTPUT_FILE
#                        Output file to write to. Optional, only necessary if not_
↪replacing inplace
# -v {False,false,0,True,true,1}, --verbose {False,false,0,True,true,1}
#                        String to control verbosity. Defaults to True.
```

(continues on next page)

(continued from previous page)

```
# -i {False,false,0,True,true,1}, --inplace {False,false,0,True,true,1}
#           Should links be replaced inplace? This should be safe but to be
↳sure, test with an output file first.
```

Programmer-Friendly Mode

```
from urlfix.urlfix import URLFix
from urlfix.dirurlfix import DirURLFix
```

Create an object of class URLFix

```
urlfix_object = URLFix("testfiles/testurls.txt", output_file="replacement.txt")
```

Replacing URLs

After creating our object, we can replace outdated URLs as follows:

```
urlfix_object.replace_urls(verbose=1)
```

The above uses default arguments and will not replace a file inplace. This is a safety mechanism to ensure one does not damage their files.

Since we set verbose to True, we get the following output:

```
urlfix_object.replace_urls()
```

To replace silently, simply set verbose to False (which is the default).

```
urlfix_object.replace_urls()
```

If there are URLs known to be valid, pass these to the `correct_urls` argument to save some time.

```
urlfix_object.replace_urls(correct_urls=[urls_here]) # Use a Sequence eg tuple, list, etc
```

Replacing several files in a directory

To replace several files in a directory, we can use `DirURLFix` as follows.

- Instantiate an object of class `DirURLFix`

```
replace_in_dir = DirURLFix("path_to_dir")
```

- Call `replace_urls`

```
replace_in_dir.replace_urls()
```

Recursively replacing links in nested directories

To replace outdated links in several files located in several directories, we set `recursive` to True. Currently, replacing links in directories nested within nested directories is not (yet) supported.

```
recursive_object = DirURLFix("path_to_root_directory", recursive=True)
```

We can then proceed as above

```
recursive_object.replace_urls() # provide other arguments as you may wish.
```

To report any issues, suggestions or improvement, please do so at [issues](#).

If you would like to cite this work, please use:

Nelson Gonzabato (2021) urlfix: Check and Fix Outdated URLs <https://github.com/Nelson-Gon/urlfix>

Thank you very much.

“Before software can be reusable it first has to be usable.” – Ralph Johnson

2.1 urlfix package

2.1.1 Submodules

2.1.2 urlfix.dirurlfix module

class urlfix.dirurlfix.**DirURLFix**(*input_dir*, *recursive=False*, *sub_recursive=False*)

Bases: object

Replace Outdated URLs given a directory of files.

replace_urls(***kwargs*)

urlfix.dirurlfix.**replace_urls_root**(*in_dir*, *recursive=False*, *sub_recursive=False*, ***kwargs*)

Parameters

- **in_dir** – Input directory
- **recursive** – Bool, should URLs be replaced in sub-directories if they exist?
- **kwargs** – Other arguments to URLFix.replace_urls
- **sub_recursive** – Bool, should URLs be replaced sub-recursively? Defaults to False.

Returns

Files with outdated links validated/replaced, as requested.

2.1.3 urlfix.urlfix module

class urlfix.urlfix.**URLFix**(*input_file*, *output_file=None*)

Bases: object

replace_urls(*verbose=False*, *correct_urls=None*, *inplace=False*)

:param *verbose* Logical. Should you be notified of what URLs have moved? Defaults to False. :param *correct_urls* A sequence of urls known to be correct. :param *inplace* Flag for inplace update operation. :return Replaces outdated URL and writes to the specified file. It also returns the number of URLs that have changed. The latter is useful for tests.

urlfix.urlfix.**file_format**(*in_file*)

2.1.4 Module contents

Check and Fix Outdated URLs

WELCOME TO URLFIX'S CHANGELOG

urlfix 0.2.3

- We now use `pytest` as our testing framework making our testing easier and more robust.
- Fixed issues with recursive replacements failing due to non-deleted intermediate files.
- Verbose messages are now printed to a log file to make the console less convoluted. The log level is set to debug so we get as much information as possible. Importantly, warnings are now also not printed on the console, the user should check the log files to see these too. This might be reverted in the future.
- Output file deletion has been optimised to avoid errors when running tests repeatedly.
- Tests are now broken down into much simpler functions for easier debugging.
- `URLFix` now warns with HTTP error codes, wherever applicable.

urlfix 0.2.2

- Script mode now supports updating links in sub-nested directories.
- `DirURLFix` now supports replacement of outdated links in sub-nested directories.
- Now supporting ReStructured text (`.rst`) files.
- Added support for RMarkdown (`.rmd`) files.
- A `recursive` argument was added to script mode. The argument `input-file` was renamed `inpath` to reflect that this may be a file or directory.
- `DirURLFix` is now fully recursive.
- Extended tests to ensure that recursion works as expected.
- Refactored `dirurlfix`'s main replacement method to allow for greater flexibility in recursive replacements.
- Fixed a bug due to differences in file orders between Linux and Windows.
- Initial support for recursive link updates. See [#24](#).

urlfix 0.2.1

- Download URL is now automated, please release new versions as `v-version-number-here`.
- Script mode has been added as a `__main__.py` module. You can now therefore call `urlfix` at the command line/Terminal via `python -m urlfix`.
- A script mode has been added to enable commandline replacement of outdated links. See [#22](#).
- Fixed issues with links not being replaced following changes to directory replacement.
- Restored inplace replacement. Using temporary files for now. See [#15](#) and [#10](#).
- Versioning is now automated. You can now check version number via `urlfix.__version__`

- `dirurlfix` is a new module dedicated to directory replacements.
- Fixed issues with markdown links in the format `[] () [] ()` not being fully matched. See [#17](#)
- Fixed issues with double text appearing in the replacement file. Related to [Issue 20](#).
- Fixed issues with URLs not being matched if they are on the same line. Issue [#20](#).
- Users are now warned if a target URL is outdated and no newer URL exists. See [#18](#)
- Fixed issues with text loss in output markdown files. See [#16](#)
- Fixed issues with tests failing when run [consecutively](#)
- Inplace replacement is no longer supported via the `inplace=True` argument.
- Replacement of files now supports adding exceptions that is URLs whose links are known to be valid.
- Added support for automatic detection of file extensions negating the need to manually specify file formats.
- Initial support for directory replacements, thanks to [nirolada](#).

urlfix 0.2.0

- `show_parsed_urls` was dropped. Future plans to find a better way to validate matched URLs.
- `find_links` was dropped. Everything is now done under `replace_urls`.
- `URLFix` is a new class to make it easier to write class methods and access class variables.
- `show_parsed_urls` is no longer necessary and may be dropped in future versions.
- `replace_urls` was refactored to avoid unnecessary loops that would otherwise slow down the process.
- Extended sanity tests to ensure that input and output files exist.
- `returned_matched` was dropped in `replace_urls`. Use `show_parsed_urls` for low level returns.
- `verbose` in `replace_urls` is now more human friendly by providing the actual name of the output file.
- Updated `testurls.md` to ensure only markdown like links are replaced.
- The regular expression in `find_links` was replaced with a more robust one.
- `visit_urls` was renamed to `replace_urls` and extended to allow inplace replacement (or not) as well as writing to an output file.
- `fixurls` was renamed to `urlfix`.
- `check_url` was removed but may be replaced.
- Add more sanity checks to ensure that the regular expressions used work as detected.
- Made expected formats and return types more explicit.
- Initial support for tests.
- Fixed issues with installation

urlfix 0.1.0

- Initial release to preserve name on PyPI.

CONTRIBUTING TO URLFIX

This document provides guidelines for contributions to `urlfix`.

Kinds of contribution

- Typo fixes
- Documentation enhancements
- Pull requests

Fixing typos and enhancing documentation

To fix typos and/or grammatical errors, please edit the corresponding `.py` or `.md` file that generates the documentation.

Please also update documentation by running `bash scripts/mkdocs.sh` as described also later.

Pull Requests

- Please raise an issue for discussion and reproducibility checks at <https://github.com/Nelson-Gon/urlfix/issues>
- Once the bug/enhancement is approved, please create a Git branch for the pull request.
- Make changes and ensure that builds are passing the necessary checks on Travis.
- Update `changelog.md` to reflect the changes made.
- Do the following:

```
bash scripts/mkdocs.sh
```

See also for a guide on Sphinx documentation.

Commit messages

Please write commit messages in the format “Extends functionality” instead of “Extended functionality”.

Maintainers only notice

Releasing

This should ideally not be run since we are auto-releasing via a GitHub action. If for some reason you would like to release a new version manually, then:

```
bash scripts/release.sh
```

The above does the following:

- Makes dist with `python setup.py sdist` at the very minimum. Ensure everything necessary is included in `Manifest.in`.

- Uploads `dist` to `test.pypi.org` with `twine upload --repository-url https://test.pypi.org/legacy/ dist/*`
- If everything looks good, asks you to upload to `pypi.org` with `twine upload dist/*`

Please note that the ‘urlfix’ project is released with a [Contributor Code of Conduct](#). By contributing to this project, you agree to abide by its terms.

INDICES AND TABLES

- `genindex`
- `modindex`
- `search`

PYTHON MODULE INDEX

U

`urlfix`, 6
`urlfix.dirurlfix`, 5
`urlfix.urlfix`, 5

D

`DirURLFix` (*class in `urlfix.dirurlfix`*), 5

F

`file_format()` (*in module `urlfix.urlfix`*), 5

M

module

`urlfix`, 6

`urlfix.dirurlfix`, 5

`urlfix.urlfix`, 5

R

`replace_urls()` (*`urlfix.dirurlfix.DirURLFix` method*), 5

`replace_urls()` (*`urlfix.urlfix.URLFix` method*), 5

`replace_urls_root()` (*in module `urlfix.dirurlfix`*), 5

U

`urlfix`

 module, 6

`URLFix` (*class in `urlfix.urlfix`*), 5

`urlfix.dirurlfix`

 module, 5

`urlfix.urlfix`

 module, 5