import-git-files

Release 0.1.dev15+g2645ea8

Michael T. Neylon

Jul 13, 2020

CONTENTS:

CHAPTER

ONE

INSTALLATION AND USAGE

This package has only been developed and tested against GitHub repositories. Other hosted repositories may work but have not been tested. It is only tested on Ubuntu and OS X and is not tested or supported on Windows.

1.1 Installation

```
pip install import-git-files
```

1.2 Usage

`import-git-files` is offered both as an command-line executable and a module that you can import into your project. To use as a module, continue to the module documentation. To use on the command-line, the usage is:

1.2.1 Example

An example could include creating a JSON file to define an import from a release of a public repository such as:

```
{
  "https://github.com/<username>/<repo>": {
    "README.md": "README.md"
  }
}
```

We can checkout a specific version:

```
{
  "https://github.com/<username>/<repo>@<revision>": {
    "README.md": "README.md"
  }
}
```

revision can be a commit sha, branch name, release, or tag.

The destination can be an absolute path or relative to PWD:

```
{
  "https://github.com/<username>/<repo>": {
    "README.md": "another_directory/README.md",
    "CHANGES.md": "/home/users/test/CHANGES.md"
```

(continues on next page)

(continued from previous page)

```
  }
}
```

After saving any of these formats to a file, such as `test.json`, we can run the command-line tool:

```
import-git-pages test.json
```

Relative destination values will be relative to where we execute this command. Some logging will indicate exactly what is happening and verbosity can be increased by passing `-v`

1.2.2 Private Repositories

Private repositories are supported by `import-git-files`. The default method will used cached credentials or prompt interactively for them. If you are in an environment where the credentials are not cached and there is not an interactive TTY, then you can supply an SSH Deploy Key or a GitHub Personal Access Token.

SSH Deploy Key

To utilize an SSH deploy key define the path to it via `--ssh-key`. This file needs to have 600 (owner read only) permissions. Do not use this in combination with a token.

There is no need to update the URLs from HTTPS to SSH protocol in your JSON file as these will be automagically handled.

Personal Access Token

If you cannot use an SSH deploy key, you can use a Personal Access Token. Set the Environment Variable `GITHUB_TOKEN` to this value to use it with the command-line tool. Do not use this in combination with the SSH key. If you use this method, it is recommended not to enable verbose logging as this can expose the token in logs.

CHAPTER

TWO

LATEST

2.1 import_git_files module

Import files from one or more git repositories to local destinations.

class `import_git_files.`**`GitExtractedFiles`**(*git_url: str*, *source_destination_map: Mapping[str, str]*, ***kwargs*)

Bases: `object`

Clone a repository into a temporary directory and then copy files out of it to new, final destinations locally. If the repository is private and requires authentication, either interactively enter the credentials or define ssh_key_path for a deploy key or token for personal access token to enable non-interactive authentication. This class is a context manager (*with GitExtractedFiles(. . .) as alias:*) to be able to operate on the temporary directory containing the cloned repository before it is removed and the return value in the alias is the list of Paths.

Parameters

- **`git_url`** (*`str`*) – a git repo URL. The default behavior checks out the default branch or the URL can be suffixed with @{revision} to specify a commit sha, branch, tag, or release.
- **`source_destination_map`** (*`Mapping[str, str]`*) – a dictionary defining the source file path as keyword, relative to the root of the repository, and its value as an absolute path or relative path to PWD of where the file should be copied.

Keyword Arguments

- **`ssh_key_path`** – git SSH access key path. This SSH key could be a deploy key. This file should have permissions 600 or will fail. If this is defined, there is not a need to change URLs from HTTPS to SSH as this will update those URLs before doing the clone. This cannot be defined at the same time as token.
- **`token`** – GitHub Personal Access Token optionally provided as a way to authenticate against private registries. Default will use the current git login, so this method is useful in non-interactive methods that are not already logged in and do not support SSH deploy keys. This cannot be defined at the same time as token.
- **`temp_dir_base`** – Optionally define the parent directory path where the temporary directory should be created rather than relying on systems' defined tmpfs.

Returns List of PosixPaths for the destination paths of the new files

`clone_repo`()

Clone the repository. Checkout branch or revision if specified

`copy_content`() → Iterator[Union[pathlib.Path, os.PathLike]]

Copy the files from the cloned repository to a new destination. The source is relative to the repo root location (temporary directory) and the destination can be absolute or relative to the PWD.

static create_parents(*path: str*)

Create parent directories for the destination path if they don't already exist.

static get_git(*git_url: str*) → Tuple[str, Optional[str]]

GitPython doesn't support @revision URLs, so detect when that is specified and split it up to get the base URL and revision separated.

Parameters **git_url** (*str*) – git url which can include the revision (commit sha, branch, tag, or release) with @revision style suffixing the git URL.

Returns base_git_url, revision if defined else None

import_git_files.**command_line**()

Command-line interface.

Returns arguments via a parser.parse_args() object

import_git_files.**main**()

Main entry point when invoking script via CLI.

CHAPTER THREE

INDICES AND TABLES

- genindex
- modindex
- search

PYTHON MODULE INDEX

INDEX