

User guide for the repository **IO-TEMPLATE-LIB**

IO-TEMPLATE-LIB is a template repository for creating Python libraries. This document describes how to use this repository to create a new repository. In the following instructions, we assume that the new repository should be named **my-lib** and the library to be created with it should be named **mylib**.

I. Requirements

Regarding operating system, Ubuntu version 20.04 and above and Windows version 10 and above are supported. An existing Python 3 installation is required. Furthermore, the use of an IDE or a text editor that can replace texts across files is useful.

II. Repository creation

1. Create the new repository **my-lib**

As described [here](#), the new repository my-lib must first be created. The creation of a very minimal basic version is sufficient, i.e. the only necessary parameter is the repository name.

2. Copy the repository **io-template-lib**

- Open Git Bash

-
- Create a bare clone of the repository.

```
git clone --bare https://github.com/io-aero/io-template-lib
```

-
- Mirror-push to the new repository

```
cd io-template-lib.git git push --mirror https://github.com/io-aero/my-lib
```

-
- Remove the temporary local repository you created earlier

```
cd .. rm -rf io-template-lib.git
```

3. Create a local copy of the new repository **my-lib**

```
git clone https://github.com/io-aero/my-lib
```

4. Delete the two files with the User's Guide

```
`user_guide.md`  
`user_guide.pdf`
```

5. Rename the following file directories and files

Old name	New name
<code>iotemplatelib</code>	<code>mylib</code>
<code>run_io_template_lib.bat</code>	<code>run_my_lib.bat</code>
<code>run_io_template_lib.sh</code>	<code>run_my_lib.sh</code>
<code>settings_io_template_lib.toml</code>	<code>settings_my_lib.toml</code>

4. Replacing texts in the new repository `my-lib`

It is absolutely necessary to respect the capitalization!

Old text	New text
<code>IO-TEMPLATE-LIB</code>	<code>MY-LIB</code>
<code>IO_TEMPLATE_LIB</code>	<code>MY_LIB</code>
<code>io-template-lib</code>	<code>my-lib</code>
<code>io_template_lib</code>	<code>my_lib</code>
<code>iotemplatelib</code>	<code>mylib</code>

5. Store your AWS access rights in file `~/.aws/credentials`

```
[default]
aws_access_key_id=...
aws_secret_access_key=...
```

6. Create the package index configuration file `~/.pypirc`

```
[distutils]
index-servers =
    codeartifact
    pypi
    testpypi

[codeartifact]
repository = https://io-aero-444046118275.d.codeartifact.us-east-1.amazonaws.com/pypi/io-aero-pypi/
username = aws
password = <password>

[pypi]
repository = https://upload.pypi.org/legacy/
```

```
username = io-aero
password = <password>

[testpypi]
repository = https://test.pypi.org/legacy/
username = io-aero-test
password = <password>
```

7. Test the current state of the new library

7.1 If Miniconda is required

- Install Miniconda
- Run `make conda-dev`
- Run `make-final`

7.2 If Miniconda is not required

- Run `make pipenv-dev`
- Run `make-final`

8. Define GitHub Actions secrets

Under 'settings' -> 'Secrets and variables' -> 'Actions' -> Tab 'Secrets' define the following 'New repository secret's:

```
AWS_ACCESS_KEY_ID
AWS_SECRET_ACCESS_KEY
GLOBAL_USER_EMAIL
```

9. Define GitHub repository variables

Under 'settings' -> 'Secrets and variables' -> 'Actions' -> Tab 'Variables' define the following 'New repository variable's:

Name	Value	Reason
CONDA	true	Include Miniconda
COVERALLS	true	Run coveralls.io

10. Commit and push all changes to the repository as 'Base version'