python-repeatable-iterable

Laurent Lyaudet
https://lyaudet.eu/laurent/
laurent.lyaudet@gmail.com

May 6, 2024

Abstract

A new type RepeatableIterable for Python and a way to obtain one instance

Current version: 2024-05-06
Current number of commits: 33
Current git SHA1: e4b32492802274eed61431b62f4894682606b998
Code lines: 1830 total lines, 1552 not empty lines, 278 empty lines.

1 Files tree

```
.
├── build_and_checks.sh
├── CONTRIBUTORS.md
├── COPYING
├── COPYING.LESSER
├── dist
│   ├── python_repeatable_iterable-1.0.0-py3-none-any.whl
│   ├── python_repeatable_iterable-1.0.0.tar.gz
│   ├── python_repeatable_iterable-1.0.1-py3-none-any.whl
│   ├── python_repeatable_iterable-1.0.1.tar.gz
│   ├── python_repeatable_iterable-1.1.0-py3-none-any.whl
│   ├── python_repeatable_iterable-1.1.0.tar.gz
│   ├── python_repeatable_iterable-1.1.1-py3-none-any.whl
│   ├── python_repeatable_iterable-1.1.1.tar.gz
│   ├── python_repeatable_iterable-2.0.0-py3-none-any.whl
│   ├── python_repeatable_iterable-2.0.0.tar.gz
│   ├── python_repeatable_iterable-2.1.0-py3-none-any.whl
│   ├── python_repeatable_iterable-2.1.0.tar.gz
│   ├── python_repeatable_iterable-2.1.1-py3-none-any.whl
│   ├── python_repeatable_iterable-2.1.1.tar.gz
│   ├── python_repeatable_iterable-2.1.2-py3-none-any.whl
│   ├── python_repeatable_iterable-2.1.2.tar.gz
│   ├── python_repeatable_iterable-2.1.3-py3-none-any.whl
│   ├── python_repeatable_iterable-2.1.3.tar.gz
│   ├── python_repeatable_iterable-2.1.4-py3-none-any.whl
│   ├── python_repeatable_iterable-2.1.4.tar.gz
│   ├── python_repeatable_iterable-2.1.5-py3-none-any.whl
│   ├── python_repeatable_iterable-2.1.5.tar.gz
│   ├── python_repeatable_iterable-2.1.6-py3-none-any.whl
│   ├── python_repeatable_iterable-2.1.6.tar.gz
│   ├── python_repeatable_iterable-2.1.7-py3-none-any.whl
│   └── python_repeatable_iterable-2.1.7.tar.gz
├── .gitignore
├── latex
│   ├── python-repeatable-iterable.tex
│   └── python-repeatable-iterable.tex.tpl
├── pyproject.toml
├── python-repeatable-iterable.pdf
├── README.md
├── README.md.tpl
├── src
│   └── python_repeatable_iterable
│       ├── __init__.py
│       └── py.typed
├── typing_test
│   └── __init__.py
└── wget_sha512.sh

6 directories, 41 files

[4.0K May  6 04:08]  ./
├── [1.6K May  6 04:07]  build_and_checks.sh*
├── [ 201 May  6 04:07]  CONTRIBUTORS.md
├── [ 34K May  2  2018]  COPYING
├── [7.5K Jul 24  2018]  COPYING.LESSER
├── [4.0K May  4 03:08]  dist/
│   ├── [ 20K Dec 20 00:26]
│   python_repeatable_iterable-1.0.0-py3-none-any.whl
│   ├── [ 16K Dec 20 00:26]  python_repeatable_iterable-1.0.0.tar.gz
│   ├── [ 20K Dec 20 00:37]
│   python_repeatable_iterable-1.0.1-py3-none-any.whl
│   ├── [ 16K Dec 20 00:37]  python_repeatable_iterable-1.0.1.tar.gz
│   ├── [ 20K Dec 20 11:48]
│   python_repeatable_iterable-1.1.0-py3-none-any.whl
│   ├── [ 16K Dec 20 11:48]  python_repeatable_iterable-1.1.0.tar.gz
│   ├── [ 21K Dec 20 16:42]
│   python_repeatable_iterable-1.1.1-py3-none-any.whl
│   ├── [ 17K Dec 20 16:42]  python_repeatable_iterable-1.1.1.tar.gz
│       ├── [ 21K Dec 30 23:32]
│        python_repeatable_iterable-2.0.0-py3-none-any.whl
│       ├── [ 17K Dec 30 23:32]  python_repeatable_iterable-2.0.0.tar.gz
│       ├── [ 21K Jan 12 01:05]
│        python_repeatable_iterable-2.1.0-py3-none-any.whl
│       ├── [ 18K Jan 12 01:05]  python_repeatable_iterable-2.1.0.tar.gz
│       ├── [ 22K Mar 20 02:59]
│        python_repeatable_iterable-2.1.1-py3-none-any.whl
│       ├── [ 19K Mar 20 02:59]  python_repeatable_iterable-2.1.1.tar.gz
│       ├── [ 22K Mar 21 20:48]
│        python_repeatable_iterable-2.1.2-py3-none-any.whl
│       ├── [ 19K Mar 21 20:48]  python_repeatable_iterable-2.1.2.tar.gz
│       ├── [ 22K Apr 28 00:04]
│        python_repeatable_iterable-2.1.3-py3-none-any.whl
│       ├── [ 20K Apr 28 00:04]  python_repeatable_iterable-2.1.3.tar.gz
│       ├── [ 22K May  2 00:44]
│        python_repeatable_iterable-2.1.4-py3-none-any.whl
│       ├── [ 21K May  2 00:44]  python_repeatable_iterable-2.1.4.tar.gz
│       ├── [ 22K May  3 03:58]
│        python_repeatable_iterable-2.1.5-py3-none-any.whl
│       ├── [ 21K May  3 03:58]  python_repeatable_iterable-2.1.5.tar.gz
│       ├── [ 22K May  3 04:17]
│        python_repeatable_iterable-2.1.6-py3-none-any.whl
│       ├── [ 21K May  3 04:17]  python_repeatable_iterable-2.1.6.tar.gz
│       ├── [ 22K May  4 03:08]
│        python_repeatable_iterable-2.1.7-py3-none-any.whl
│       └── [ 21K May  4 03:08]  python_repeatable_iterable-2.1.7.tar.gz
├── [  31 May  4 03:03]  .gitignore
├── [4.0K May  6 04:08]  latex/
│   ├── [4.1K May  6 04:08]  python-repeatable-iterable.tex
│   └── [4.0K May  5 16:39]  python-repeatable-iterable.tex.tpl
├── [1.4K May  6 04:06]  pyproject.toml
├── [ 80K May  6 04:08]  python-repeatable-iterable.pdf
├── [4.3K May  6 04:08]  README.md
├── [4.3K Apr 27 23:57]  README.md.tpl
├── [4.0K Dec 19 23:50]  src/
│   └── [4.0K May  6 04:08]  python_repeatable_iterable/
│       ├── [4.8K May  6 04:08]  __init__.py
│       └── [   0 Dec 30 23:19]  py.typed
├── [4.0K May  6 04:08]  typing_test/
│   └── [2.2K May  6 04:08]  __init__.py
└── [1.1K May  4 03:04]  wget_sha512.sh

6 directories, 41 files
```

2 Listing of files

The following source code is covered by LGPLv3+. The text of the license is available at: `https://www.gnu.org/licenses/`. The git repository of this source code is also available at: `https://github.com/LLyaudet/python-repeatable-iterable/`.

2.1 build_and_checks.sh

```
#!/usr/bin/env bash
# This file is part of python-repeatable-iterable library.
#
# python-repeatable-iterable is free software:
# you can redistribute it and/or modify it under the terms
# of the GNU Lesser General Public License
# as published by the Free Software Foundation,
# either version 3 of the License,
# or (at your option) any later version.
#
# python-repeatable-iterable is distributed in the hope
# that it will be useful,
# but WITHOUT ANY WARRANTY;
# without even the implied warranty of
# MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.
# See the GNU Lesser General Public License for more details.
#
# You should have received a copy of
# the GNU Lesser General Public License
# along with python-repeatable-iterable.
# If not, see <https://www.gnu.org/licenses/>.
#
# ©Copyright 2023-2024 Laurent Lyaudet

source ./wget_sha512.sh

mkdir -p build_and_checks_dependencies
subdir="build_and_checks_dependencies"

personal_github="https://raw.githubusercontent.com/LLyaudet/"
dependencies="DevOrSysAdminScripts/main/build_and_checks_dependencies"
URL_beginning="$personal_github$dependencies"

script="$URL_beginning/common_build_and_checks.sh"
correct_sha512='a46cd00d7b2d90fa1a3c7923244879fad28e789ff7dda791a0bd0'
correct_sha512+='c723848c12cb73ccf2c4c5875cdb674237da9696ef9e4deac07d'
correct_sha512+='c2b04aed6d90ffb98b9b0c4'
wget_sha512 "./$subdir/common_build_and_checks.sh" "$script"\
  "$correct_sha512"
chmod +x "./$subdir/common_build_and_checks.sh"

cwd="."
if [[ -n "$1" ]];
then
  cwd="$1"
fi

./build_and_checks_dependencies/common_build_and_checks.sh "$cwd"

echo "Running pylint"
pylint src/python_repeatable_iterable/
pylint typing_test/
```

2.2 CONTRIBUTORS.md

```
# python-repeatable-iterable contributors

Laurent Lyaudet, creator of the package

David Salvisberg, suggested huge improvements here:
<https://discuss.python.org/t/repeatableiterable-type/42106/1>.

```

2.3 .gitignore

```
build_and_checks_dependencies/
```

2.4 latex/python-repeatable-iterable.tex.tpl

```
%!/usr/bin/env bash
% This file is part of python-repeatable-iterable library.
%
% python-repeatable-iterable is free software:
% you can redistribute it and/or modify it under the terms
% of the GNU Lesser General Public License
% as published by the Free Software Foundation,
% either version 3 of the License,
% or (at your option) any later version.
%
% python-repeatable-iterable is distributed in the hope
% that it will be useful,
% but WITHOUT ANY WARRANTY;
% without even the implied warranty of
% MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.
% See the GNU Lesser General Public License for more details.
%
% You should have received a copy of
% the GNU Lesser General Public License
% along with python-repeatable-iterable.
% If not, see <https://www.gnu.org/licenses/>.
%
% ©Copyright 2023-2024 Laurent Lyaudet

\documentclass{article}

\usepackage[utf8]{inputenc}
\usepackage{subfigure}
\usepackage{amsmath}
\usepackage{amssymb}
\usepackage{amsthm}
\usepackage[pdftex]{hyperref}
\usepackage{tikz}
\usepackage{caption}
\usepackage[round]{natbib}
\usepackage{fancyhdr}
\usepackage{amsfonts}
\usepackage{times}
\usepackage{ifpdf}
\usepackage{latexsym}
\usepackage{graphicx}
\usepackage{enumerate}
\usepackage{pmboxdraw}
\usepackage{fancyvrb}

% *** les environnements ***
%\theoremstyle{break}
\newtheorem{definition}{Definition}[section]
\newtheorem{proposition}[definition]{Proposition}
\newtheorem{theorem}[definition]{Theorem}
\newtheorem{lemma}[definition]{Lemma}
\newtheorem{corollary}[definition]{Corollary}
\newtheorem{remark}[definition]{Remark}
\newtheorem{openproblem}[definition]{Open problem}

\begin{document}

\author{
  Laurent Lyaudet\\
  \url{https://lyaudet.eu/laurent/}\\
  laurent.lyaudet@gmail.com
```

```
}
\title{python-repeatable-iterable}

\maketitle
\begin{abstract}
A new type RepeatableIterable for Python
and a way to obtain one instance
\end{abstract}

Current version: @current_date@

Current number of commits: @number_of_commits@

Current git SHA1: @current_git_SHA1@

Code lines: @number_of_lines@

\section{Files tree}
\label{section:tree}

\begin{verbatim}
@current_tree_light@
\end{verbatim}

\begin{verbatim}
@current_tree@
\end{verbatim}

\section{Listing of files}
\label{section:listing}

The following source code is covered by LGPLv3+.
The text of the license is available at:
\url{https://www.gnu.org/licenses/}.
The git repository of this source code is also available at:
\url{https://github.com/LLyaudet/python-repeatable-iterable/}.


\subsection{
  build\_and\_checks.sh
}
\label{
  build_and_checkssh
}

\VerbatimInput[numbers=left,xleftmargin=-5mm]{
  build_and_checks.sh
}


\subsection{
  CONTRIBUTORS.md
}
\label{
  CONTRIBUTORSmd
}

\VerbatimInput[numbers=left,xleftmargin=-5mm]{
  CONTRIBUTORS.md
}


\subsection{
  .gitignore
}
\label{
  gitignore
}

\VerbatimInput[numbers=left,xleftmargin=-5mm]{
  .gitignore
}


\subsection{
  latex/python-repeatable-iterable.tex.tpl
}
\label{
  latex:python-repeatable-iterabletextpl
}

\VerbatimInput[numbers=left,xleftmargin=-5mm]{
  latex/python-repeatable-iterable.tex.tpl
}


\subsection{
  pyproject.toml
}
\label{
  pyprojecttoml
}
```

```

\VerbatimInput[numbers=left,xleftmargin=-5mm]{
  pyproject.toml
}


\subsection{
  README.md.tpl
}
\label{
  READMEmdtpl
}

\VerbatimInput[numbers=left,xleftmargin=-5mm]{
  README.md.tpl
}


\subsection{
  src/python\_repeatable\_iterable/\_\_init\_\_.py
}
\label{
  src:python_repeatable_iterable:__init__py
}

\VerbatimInput[numbers=left,xleftmargin=-5mm]{
  src/python_repeatable_iterable/__init__.py
}


\subsection{
  src/python\_repeatable\_iterable/py.typed
}
\label{
  src:python_repeatable_iterable:pytyped
}

\VerbatimInput[numbers=left,xleftmargin=-5mm]{
  src/python_repeatable_iterable/py.typed
}


\subsection{
  typing\_test/\_\_init\_\_.py
}
\label{
  typing_test:__init__py
}

\VerbatimInput[numbers=left,xleftmargin=-5mm]{
  typing_test/__init__.py
}


\subsection{
  wget\_sha512.sh
}
\label{
  wget_sha512sh
}

\VerbatimInput[numbers=left,xleftmargin=-5mm]{
  wget_sha512.sh
}


Merci Dieu ! Merci P\`ere ! Merci Seigneur ! Merci Saint Esprit !

\end{document}
```

2.5 pyproject.toml

```
# pyproject.toml

[build-system]
requires = ["hatchling"]
build-backend = "hatchling.build"

[project]
name = "python-repeatable-iterable"
version = "2.1.8"
description = """\
Add a RepeatableIterable type and a function to obtain it\
"""
readme = "README.md"
authors = [
    { name = "Laurent Lyaudet", email = "laurent.lyaudet@gmail.com" },
]
maintainers = [
    { name = "Laurent Lyaudet", email = "laurent.lyaudet@gmail.com" },
]
license = { file = "COPYING.LESSER" }
```

```
classifiers = [
    "Development Status :: 5 - Production/Stable",
    "Intended Audience :: Developers",
    """\
License :: OSI Approved :: \
GNU Lesser General Public License v3 or later (LGPLv3+)\
""",
    "Operating System :: OS Independent",
    "Programming Language :: Python",
    "Programming Language :: Python :: 3",
    "Topic :: Software Development :: Libraries :: Python Modules",
    "Typing :: Typed",
]
keywords = ["Python", "Iterable", "Repeatable", "RepeatableIterable"]
dependencies = [
   "python-none-objects==1.1.11",
]
requires-python = ">=3.11"

[project.optional-dependencies]
dev = [
    "black",
    "isort",
    "mypy",
    "pylint",
]

[project.urls]
"Homepage" = "https://github.com/LLyaudet/python-repeatable-iterable"
"Bug Tracker" = """\
https://github.com/LLyaudet/python-repeatable-iterable/issues\
"""

[tool.black]
line-length = 70

[tool.isort]
profile = "black"
line_length = 70
```

2.6 README.md.tpl

```
# python-repeatable-iterable

[![PyPI-version-badge]][PyPI-package-page]
[![Downloads-badge]][PyPIStats-package-page]
```

~~~
[![Code-style:black:badge]][Black-GitHub.com]
[![Imports:isort:badge]][Isort-GitHub.io]
[![Typecheck:mypy:badge]][Typecheck-mypy-lang.org]
[![Linting:pylint:badge]][Pylint-GitHub.com]
[![CodeFactor-badge]][CodeFactor-package-page]
[![CodeClimateMaintainability-badge]][CodeClimateM13y-package-page]
[![Codacy-badge]][Codacy-package-page]
![GitHub-top-language-badge]
![GitHub-license-badge]
![PyPI-python-version-badge]
![GitHub-code-size-in-bytes-badge]

| **A new type RepeatableIterable for Python** |
|:--------------------------------------------:|
|     **and a way to obtain one instance**     |


Since in Python an Iterator is an Iterable
and that you cannot iterate multiple times on an iterator,
you may encounter WTF bugs, even with type checking.
This package provides possible solutions to this problem.
See here for a discussion on this problem:
<https://stackoverflow.com/questions/63104689>
(/what-is-the-pythonic-way-to-represent-an-iterable
-that-can-be-iterated-over-mult).

Before:
```python3
def foo(iterable: Iterable):
    for that in iterable:
        bar(that)
    for that in iterable:
        # possible bug
        baz(that)

foo(something)
```

After solution 1:
```python3
from python_repeatable_iterable import RepeatableIterable

def foo(iterable: RepeatableIterable[object]):
    for that in iterable:
        bar(that)
    for that in iterable:
~~~

~~~
        baz(that)

something_else = RepeatableIterable(something)
foo(something_else)
```

After solution 2:
```python3
from python_repeatable_iterable import RepeatableIterable

def foo(iterable: Iterable):
    iterable = RepeatableIterable(iterable)
    for that in iterable:
        bar(that)
    for that in iterable:
        baz(that)

foo(something)
```

If you develop something where you have no control on
what another dev might give you as input,
you have 2 possibilities:

- hope for the best ;),
- or harden your code to have less support work to do :).

This applies if you dev something that is:

- closed source or open source,
- available to everyone on the Internet,
  available only to customers or colleagues
  that you may personally know or not.

Solution 2 above is a nice solution
with a reasonable performance cost :).

[PyPI-version-badge]: https://img.shields.io/pypi/v/\
python-repeatable-iterable.svg

[PyPI-package-page]: https://pypi.org/project/\
python-repeatable-iterable/

[Downloads-badge]: https://img.shields.io/pypi/dm/\
python-repeatable-iterable

~~~

```
[PyPIStats-package-page]: https://pypistats.org/packages/\
python-repeatable-iterable

[Code-style:black:badge]: https://img.shields.io/badge/\
code%20style-black-000000.svg

[Black-GitHub.com]: https://github.com/psf/black

[Imports:isort:badge]: https://img.shields.io/badge/\
%20imports-isort-%231674b1?style=flat&labelColor=ef8336

[Isort-GitHub.io]: https://pycqa.github.io/isort/

[Typecheck:mypy:badge]: https://www.mypy-lang.org/static/\
mypy_badge.svg

[Typecheck-mypy-lang.org]: https://mypy-lang.org/

[Linting:pylint:badge]: https://img.shields.io/badge/\
linting-pylint-yellowgreen

[Pylint-GitHub.com]: https://github.com/pylint-dev/pylint

[CodeFactor-badge]: https://www.codefactor.io/repository/github/\
llyaudet/python-repeatable-iterable/badge/main

[CodeFactor-package-page]: https://www.codefactor.io/repository/\
github/llyaudet/python-repeatable-iterable/overview/main

[CodeClimateMaintainability-badge]: https://api.codeclimate.com/v1/\
badges/89044bfd52999e4f07f6/maintainability

[CodeClimateM13y-package-page]: https://codeclimate.com/github/\
LLyaudet/python-repeatable-iterable/maintainability

[Codacy-badge]: https://app.codacy.com/project/badge/Grade/\
1c70116c2d714e3889606519937cb11d

[Codacy-package-page]: https://app.codacy.com/gh/LLyaudet/\
python-repeatable-iterable/dashboard?utm_source=gh\
&utm_medium=referral&utm_content=&utm_campaign=Badge_grade

[GitHub-top-language-badge]: https://img.shields.io/github/\
languages/top/llyaudet/python-repeatable-iterable

[GitHub-license-badge]: https://img.shields.io/github/license/\
llyaudet/python-repeatable-iterable

[PyPI-python-version-badge]: https://img.shields.io/pypi/pyversions/\
python-repeatable-iterable

[GitHub-code-size-in-bytes-badge]: https://img.shields.io/github/\
languages/code-size/llyaudet/python-repeatable-iterable

```

2.7 src/python_repeatable_iterable/__init__.py

```
"""
This file is part of python-repeatable-iterable library.

python-repeatable-iterable is free software:
you can redistribute it and/or modify it under the terms
of the GNU Lesser General Public License
as published by the Free Software Foundation,
either version 3 of the License,
or (at your option) any later version.

python-repeatable-iterable is distributed in the hope
that it will be useful,
but WITHOUT ANY WARRANTY;
without even the implied warranty of
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.
See the GNU Lesser General Public License for more details.

You should have received a copy of
the GNU Lesser General Public License
along with python-repeatable-iterable.
If not, see <https://www.gnu.org/licenses/>.

©Copyright 2023-2024 Laurent Lyaudet
----------------------------------------------------------------------
from typing import Iterable, NewType, Type
from _collections_abc import dict_keys, dict_values, dict_items
from python_none_objects import NoneIterable

A first attempt at defining the RepeatableIterable type,
but it is not generic.
It defines a RepeatableIterable() function that cannot be subscripted.
RepeatableIterable = NewType("RepeatableIterable", Iterable)

----------------------------------------------------------------------

```

```
The following function is a first attempt
that conveys the intent more clearly.
But it is not safe, see discussion just after.
def get_repeatable_iterable(
    iterable: Iterable,
    safe_classes: Iterable[Type] = NoneIterable,
) -> RepeatableIterable:
    if isinstance(
        iterable,
        (
            list,
            tuple,
            range,
            str,
            bytes,
            bytearray,
            memoryview,
            set,
            frozenset,
            dict,
            dict_keys,
            dict_values,
            dict_items,
        ),
    ):
        return iterable
    if isinstance(iterable, safe_classes):
        return iterable
    return list(iterable)

Indeed this function is not safe, since you can subclass builtins
or other classes to make them not RepeatableIterable
from the point of view of the semantic of this type.
Consider the following code for example:
>>> class MySet(set):
...     def __init__(self, *args, **kwargs):
...         super().__init__(*args, **kwargs)
...         self.iteration_count = 0
...     def __iter__(self):
...         self.iteration_count += 1
...         if self.iteration_count == 1:
...             return super().__iter__()
...         return ().__iter__()
...
>>> s = MySet('abcd')
>>> for x in s: print(x)
```

```
...
b
a
d
c
>>> for x in s: print(x)
...
>>> for x in s: print(x)
...
>>> isinstance(s, set)
True

See here a list of builtins that can be subclassed or not:
https://stackoverflow.com/questions/10061752
/which-classes-cannot-be-subclassed

----------------------------------------------------------------------

This second attempt has been included in the class RepeatableIterable.
def get_repeatable_iterable(
    iterable: Iterable,
    safe_classes: Iterable[Type] = NoneIterable,
) -> RepeatableIterable:
    # Here is an implementation avoiding the previous problem.
    iterable_type = type(iterable)
    for some_class in (
        list,
        tuple,
        range,
        str,
        bytes,
        bytearray,
        memoryview,
        set,
        frozenset,
        dict,
        dict_keys,
        dict_values,
        dict_items,
        *safe_classes,
    ):
        if iterable_type is some_class:
            return iterable
    return list(iterable)
"""

```

```
from typing import Iterable, Iterator, TypeVar, cast
from _collections_abc import dict_items, dict_keys, dict_values

from python_none_objects import NoneIterable

T = TypeVar("T")


class RepeatableIterable(Iterable[T]):
    """
    An asbtract class that is here to define a type and
    cast other objects to this type if possible in its __new__ method.
    """

    # pylint: disable-next=non-iterator-returned
    def __iter__(self) -> Iterator[T]:
        # Instances of RepeatableIterable don't actually exist.
        return NotImplemented

    def __new__(
        cls,
        iterable: Iterable[T],
        safe_classes: Iterable[type[object]] = NoneIterable,
    ) -> "RepeatableIterable[T]":
        """
        Here is an implementation avoiding the previous problem.
        """
        iterable_type = type(iterable)
        for some_class in (
            list,
            tuple,
            range,
            str,
            bytes,
            bytearray,
            memoryview,
            set,
            frozenset,
            dict,
            dict_keys,
            dict_values,
            dict_items,
            *safe_classes,
        ):
            if iterable_type is some_class:
                return cast("RepeatableIterable[T]", iterable)
        return cast("RepeatableIterable[T]", list(iterable))
```

2.8 src/python_repeatable_iterable/py.typed

2.9 typing_test/__init__.py

```
"""
This file is part of python-repeatable-iterable library.

python-repeatable-iterable is free software:
you can redistribute it and/or modify it under the terms
of the GNU Lesser General Public License
as published by the Free Software Foundation,
either version 3 of the License,
or (at your option) any later version.

python-repeatable-iterable is distributed in the hope
that it will be useful,
but WITHOUT ANY WARRANTY;
without even the implied warranty of
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.
See the GNU Lesser General Public License for more details.

You should have received a copy of
the GNU Lesser General Public License
along with python-repeatable-iterable.
If not, see <https://www.gnu.org/licenses/>.

©Copyright 2023-2024 Laurent Lyaudet
"""

import sys
from typing import Iterable, List, Never, TypeVar

sys.path.insert(0, "../src/")
# pylint: disable-next=wrong-import-position
from python_repeatable_iterable import RepeatableIterable

# pylint: disable-next=invalid-name
T1 = TypeVar("T1")


def test_arg_to_return_typing(
    x: RepeatableIterable[List[T1]],
) -> List[T1]:
    """
    Check that mypy follows the types
    between the argument and the return of the function
    for the type of the content of the list.
    """
    result = []
    for y in x:
        result.extend(y)
    for y in x:
        result.extend(y)
    return result


def test_arg_to_return_via_call_typing(
    x: Iterable[List[T1]],
) -> List[T1]:
    """
    Check that mypy follows the types
    between the argument and the return of the function
    for the type of the content of the list
    with indirections.
    """
    return test_arg_to_return_typing(RepeatableIterable(x))


a: List[List[Never]] = [[], []]
print(test_arg_to_return_via_call_typing(a))

b = (x for x in a)
print(test_arg_to_return_via_call_typing(b))


def test_arg_to_return_via_cast_typing(
    x: RepeatableIterable[List[T1]],
) -> RepeatableIterable[T1]:
    """
    Check that mypy follows the types
    between the argument and the return of the function
    for the type of the content of the list
    with a final cast.
    """
    result = []
    for y in x:
        result.extend(y)
    for y in x:
        result.extend(y)
    return RepeatableIterable(result)
```

2.10 wget_sha512.sh

```
#!/usr/bin/env bash
# This file is part of DevOrSysAdminScripts library.
#
# DevOrSysAdminScripts is free software:
# you can redistribute it and/or modify it under the terms
# of the GNU Lesser General Public License
# as published by the Free Software Foundation,
# either version 3 of the License,
# or (at your option) any later version.
#
# DevOrSysAdminScripts is distributed in the hope
# that it will be useful,
# but WITHOUT ANY WARRANTY;
# without even the implied warranty of
# MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.
# See the GNU Lesser General Public License for more details.
#
# You should have received a copy of
# the GNU Lesser General Public License
# along with DevOrSysAdminScripts.
# If not, see <https://www.gnu.org/licenses/>.
#
# ©Copyright 2023-2024 Laurent Lyaudet

wget_sha512(){
  # $1 filename
  # $2 download_URL
  # $3 correct_sha512
  if [[ ! -f "$1" ]];
  then
    wget -O "$1" "$2"
  fi
  present_sha512=$(sha512sum "$1" | cut -f1 -d' ')
  if [[ "$present_sha512" != "$3" ]];
  then
    echo "$1 does not have correct sha512"
    echo "wanted $3"
    echo "found $present_sha512"
    exit
  fi
}
```

Merci Dieu ! Merci Père ! Merci Seigneur ! Merci Saint Esprit !