

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

```
1  #!/usr/bin/env bash
2  # This file is part of python-repeatable-iterable library.
3  #
4  # python-repeatable-iterable is free software:
5  # you can redistribute it and/or modify it under the terms
6  # of the GNU Lesser General Public License
7  # as published by the Free Software Foundation,
8  # either version 3 of the License,
9  # or (at your option) any later version.
10 #
11 # python-repeatable-iterable is distributed in the hope
12 # that it will be useful,
13 # but WITHOUT ANY WARRANTY;
14 # without even the implied warranty of
15 # MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.
16 # See the GNU Lesser General Public License for more details.
17 #
18 # You should have received a copy of
19 # the GNU Lesser General Public License
20 # along with python-repeatable-iterable.
21 # If not, see <https://www.gnu.org/licenses/>.
22 #
23 # ©Copyright 2023-2024 Laurent Lyaudet
24
25 source ./wget_sha512.sh
26
27 mkdir -p build_and_checks_dependencies
28 subdir="build_and_checks_dependencies"
29
30 personal_github="https://raw.githubusercontent.com/LLyaudet/"
31 dependencies="DevOrSysAdminScripts/main/build_and_checks_dependencies"
32 URL_beginning="$personal_github$dependencies"
33
34 script="$URL_beginning/common_build_and_checks.sh"
35 correct_sha512='a46cd00d7b2d90fala3c7923244879fad28e789ff7dda791a0bd0'
36 correct_sha512+='c723848c12cb73ccf2c4c5875cdb674237da9696ef9e4deac07d'
37 correct_sha512+='c2b04aed6d90ffb98b9b0c4'
38 wget_sha512 "$subdir/common_build_and_checks.sh" "$script"
```

```

39     "$correct_sha512"
40     chmod +x "$subdir/common_build_and_checks.sh"
41
42     cwd="."
43     if [[ -n "$1" ]];
44     then
45         cwd="$1"
46     fi
47
48     ./build_and_checks_dependencies/common_build_and_checks.sh "$cwd"
49
50     echo "Running pylint"
51     pylint src/python_repeatable_iterable/
52     pylint typing_test/

```

2.2 CONTRIBUTORS.md

```

1  # python-repeatable-iterable contributors
2
3  Laurent Lyaudet, creator of the package
4
5  David Salvisberg, suggested huge improvements here:
6  <https://discuss.python.org/t/repeatableiterable-type/42106/1>.
7

```

2.3 .gitignore

```

1  build_and_checks_dependencies/

```

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

```

1  %!/usr/bin/env bash
2  % This file is part of python-repeatable-iterable library.
3  %
4  % python-repeatable-iterable is free software:
5  % you can redistribute it and/or modify it under the terms
6  % of the GNU Lesser General Public License
7  % as published by the Free Software Foundation,
8  % either version 3 of the License,
9  % or (at your option) any later version.
10 %
11 % python-repeatable-iterable is distributed in the hope
12 % that it will be useful,
13 % but WITHOUT ANY WARRANTY;
14 % without even the implied warranty of
15 % MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

```

```

16 % See the GNU Lesser General Public License for more details.
17 %
18 % You should have received a copy of
19 % the GNU Lesser General Public License
20 % along with python-repeatable-iterable.
21 % If not, see <https://www.gnu.org/licenses/>.
22 %
23 % ©Copyright 2023–2024 Laurent Lyaudet
24
25 \documentclass{article}
26
27 \usepackage[utf8]{inputenc}
28 \usepackage{subfigure}
29 \usepackage{amsmath}
30 \usepackage{amssymb}
31 \usepackage{amsthm}
32 \usepackage[pdftex]{hyperref}
33 \usepackage{tikz}
34 \usepackage{caption}
35 \usepackage[round]{natbib}
36 \usepackage{fancyhdr}
37 \usepackage{amsfonts}
38 \usepackage{times}
39 \usepackage{ifpdf}
40 \usepackage{latexsym}
41 \usepackage{graphicx}
42 \usepackage{enumerate}
43 \usepackage{pmbboxdraw}
44 \usepackage{fancyvrb}
45
46 % *** les environnements ***
47 %\theoremstyle{break}
48 \newtheorem{definition}{Definition}[section]
49 \newtheorem{proposition}[definition]{Proposition}
50 \newtheorem{theorem}[definition]{Theorem}
51 \newtheorem{lemma}[definition]{Lemma}
52 \newtheorem{corollary}[definition]{Corollary}
53 \newtheorem{remark}[definition]{Remark}
54 \newtheorem{openproblem}[definition]{Open problem}
55
56 \begin{document}
57
58 \author{
59   Laurent Lyaudet\\
60   \url{https://lyaudet.eu/laurent/}\\
61   laurent.lyaudet@gmail.com

```

```

62 }
63 \title{python-repeatable-iterable}
64
65 \maketitle
66 \begin{abstract}
67 A new type RepeatableIterable for Python
68 and a way to obtain one instance
69 \end{abstract}
70
71 Current version: @current_date@
72
73 Current number of commits: @number_of_commits@
74
75 Current git SHA1: @current_git_SHA1@
76
77 Code lines: @number_of_lines@
78
79 \section{Files tree}
80 \label{section:tree}
81
82 \begin{verbatim}
83 @current_tree_light@
84 \end{verbatim}
85
86 \begin{verbatim}
87 @current_tree@
88 \end{verbatim}
89
90 \section{Listing of files}
91 \label{section:listing}
92
93 The following source code is covered by LGPLv3+.
94 The text of the license is available at:
95 \url{https://www.gnu.org/licenses/}.
96 The git repository of this source code is also available at:
97 \url{https://github.com/LLyaudet/python-repeatable-iterable/}.
98
99
100 \subsection{
101     build\_and\_checks.sh
102 }
103 \label{
104     build_and_checkssh
105 }
106
107 \VerbatimInput[numbers=left,xleftmargin=-5mm]{

```

```

108     build_and_checks.sh
109 }
110
111
112 \subsection{
113     CONTRIBUTORS.md
114 }
115 \label{
116     CONTRIBUTORSmd
117 }
118
119 \VerbatimInput[numbers=left,xleftmargin=-5mm]{
120     CONTRIBUTORS.md
121 }
122
123
124 \subsection{
125     .gitignore
126 }
127 \label{
128     gitignore
129 }
130
131 \VerbatimInput[numbers=left,xleftmargin=-5mm]{
132     .gitignore
133 }
134
135
136 \subsection{
137     latex/python-repeatable-iterable.tex.tpl
138 }
139 \label{
140     latex:python-repeatable-iterabletexttpl
141 }
142
143 \VerbatimInput[numbers=left,xleftmargin=-5mm]{
144     latex/python-repeatable-iterable.tex.tpl
145 }
146
147
148 \subsection{
149     pyproject.toml
150 }
151 \label{
152     pyprojecttoml
153 }

```



```

154
155 \VerbatimInput[numbers=left,xleftmargin=-5mm]{
156     pyproject.toml
157 }
158
159
160 \subsection{
161     README.md.tpl
162 }
163 \label{
164     READMEmdtpl
165 }
166
167 \VerbatimInput[numbers=left,xleftmargin=-5mm]{
168     README.md.tpl
169 }
170
171
172 \subsection{
173     src/python/_repeatable/_iterable/_\__init___.py
174 }
175 \label{
176     src:python_repeatable_iterable:__init__.py
177 }
178
179 \VerbatimInput[numbers=left,xleftmargin=-5mm]{
180     src/python_repeatable_iterable/__init__.py
181 }
182
183
184 \subsection{
185     src/python/_repeatable/_iterable/py.typed
186 }
187 \label{
188     src:python_repeatable_iterable:pytyped
189 }
190
191 \VerbatimInput[numbers=left,xleftmargin=-5mm]{
192     src/python_repeatable_iterable/py.typed
193 }
194
195
196 \subsection{
197     typing\_test/_\__init___.py
198 }
199 \label{

```

```

200     typing_test: __init__.py
201 }
202
203 \VerbatimInput[numbers=left,xleftmargin=-5mm]{
204     typing_test/__init__.py
205 }
206
207
208 \subsection{
209     wget\_sha512.sh
210 }
211 \label{
212     wget_sha512sh
213 }
214
215 \VerbatimInput[numbers=left,xleftmargin=-5mm]{
216     wget_sha512.sh
217 }
218
219
220 Merci Dieu ! Merci P\`ere ! Merci Seigneur ! Merci Saint Esprit !
221
222 \end{document}

```

2.5 pyproject.toml

```

1  # pyproject.toml
2
3  [build-system]
4  requires = ["hatchling"]
5  build-backend = "hatchling.build"
6
7  [project]
8  name = "python-repeatable-iterable"
9  version = "2.1.8"
10 description = """\
11 Add a RepeatableIterable type and a function to obtain it\
12 """
13 readme = "README.md"
14 authors = [
15     { name = "Laurent Lyaudet", email = "laurent.lyaudet@gmail.com" },
16 ]
17 maintainers = [
18     { name = "Laurent Lyaudet", email = "laurent.lyaudet@gmail.com" },
19 ]
20 license = { file = "COPYING.LESSER" }

```

```

21 classifiers = [
22     "Development Status :: 5 - Production/Stable",
23     "Intended Audience :: Developers",
24     """\
25 License :: OSI Approved :: \
26 GNU Lesser General Public License v3 or later (LGPLv3+)\
27 """,
28     "Operating System :: OS Independent",
29     "Programming Language :: Python",
30     "Programming Language :: Python :: 3",
31     "Topic :: Software Development :: Libraries :: Python Modules",
32     "Typing :: Typed",
33 ]
34 keywords = ["Python", "Iterable", "Repeatable", "RepeatableIterable"]
35 dependencies = [
36     "python-none-objects==1.1.11",
37 ]
38 requires-python = ">=3.11"
39
40 [project.optional-dependencies]
41 dev = [
42     "black",
43     "isort",
44     "mypy",
45     "pylint",
46 ]
47
48 [project.urls]
49 "Homepage" = "https://github.com/LLyaudet/python-repeatable-iterable"
50 "Bug Tracker" = """\
51 https://github.com/LLyaudet/python-repeatable-iterable/issues\
52 """
53
54 [tool.black]
55 line-length = 70
56
57 [tool.isort]
58 profile = "black"
59 line_length = 70

```

2.6 README.md.tpl

```

1 # python-repeatable-iterable
2
3 [[PyPI-version-badge]][PyPI-package-page]
4 [[Downloads-badge]][PyPIStats-package-page]

```

```

5  [![Code-style:black:badge]] [Black-GitHub.com]
6  [![Imports:isort:badge]] [Isort-GitHub.io]
7  [![Typecheck:mypy:badge]] [Typecheck-mypy-lang.org]
8  [![Linting:pylint:badge]] [Pylint-GitHub.com]
9  [![CodeFactor-badge]] [CodeFactor-package-page]
10 [![CodeClimateMaintainability-badge]] [CodeClimateM13y-package-page]
11 [![Codacy-badge]] [Codacy-package-page]
12 ![GitHub-top-language-badge]
13 ![GitHub-license-badge]
14 ![PyPI-python-version-badge]
15 ![GitHub-code-size-in-bytes-badge]
16
17 | **A new type RepeatableIterable for Python** |
18 | :-----: |
19 |         **and a way to obtain one instance**         |
20
21
22 Since in Python an Iterator is an Iterable
23 and that you cannot iterate multiple times on an iterator,
24 you may encounter WTF bugs, even with type checking.
25 This package provides possible solutions to this problem.
26 See here for a discussion on this problem:
27 <https://stackoverflow.com/questions/63104689>
28 (/what-is-the-pythonic-way-to-represent-an-iterable
29 -that-can-be-iterated-over-mult).
30
31 Before:
32 ```python3
33 def foo(iterable: Iterable):
34     for that in iterable:
35         bar(that)
36     for that in iterable:
37         # possible bug
38         baz(that)
39
40 foo(something)
41 ```
42
43 After solution 1:
44 ```python3
45 from python_repeatable_iterable import RepeatableIterable
46
47 def foo(iterable: RepeatableIterable[object]):
48     for that in iterable:
49         bar(that)
50     for that in iterable:

```

```

51         baz(that)
52
53 something_else = RepeatableIterable(something)
54 foo(something_else)
55 '''
56
57 After solution 2:
58 ```python3
59 from python_repeatable_iterable import RepeatableIterable
60
61 def foo(iterable: Iterable):
62     iterable = RepeatableIterable(iterable)
63     for that in iterable:
64         bar(that)
65     for that in iterable:
66         baz(that)
67
68 foo(something)
69 '''
70
71 If you develop something where you have no control on
72 what another dev might give you as input,
73 you have 2 possibilities:
74
75 - hope for the best ;),
76 - or harden your code to have less support work to do :).
77
78 This applies if you dev something that is:
79
80 - closed source or open source,
81 - available to everyone on the Internet,
82   available only to customers or colleagues
83   that you may personally know or not.
84
85 Solution 2 above is a nice solution
86 with a reasonable performance cost :).
87
88 [PyPI-version-badge]: https://img.shields.io/pypi/v/\
89 python-repeatable-iterable.svg
90
91 [PyPI-package-page]: https://pypi.org/project/\
92 python-repeatable-iterable/
93
94 [Downloads-badge]: https://img.shields.io/pypi/dm/\
95 python-repeatable-iterable
96

```

```

97 [PyPIStats-package-page]: https://pypistats.org/packages/\
98 python-repeatable-iterable
99
100 [Code-style:black:badge]: https://img.shields.io/badge/\
101 code%20style-black-000000.svg
102
103 [Black-GitHub.com]: https://github.com/psf/black
104
105 [Imports:isort:badge]: https://img.shields.io/badge/\
106 %20imports-isort-%231674b1?style=flat&labelColor=ef8336
107
108 [Isort-GitHub.io]: https://pycqa.github.io/isort/
109
110 [Typecheck:mypy:badge]: https://www.mypy-lang.org/static/\
111 mypy_badge.svg
112
113 [Typecheck-mypy-lang.org]: https://mypy-lang.org/
114
115 [Linting:pylint:badge]: https://img.shields.io/badge/\
116 linting-pylint-yellowgreen
117
118 [Pylint-GitHub.com]: https://github.com/pylint-dev/pylint
119
120 [CodeFactor-badge]: https://www.codefactor.io/repository/github/\
121 llyaudet/python-repeatable-iterable/badge/main
122
123 [CodeFactor-package-page]: https://www.codefactor.io/repository/\
124 github/llyaudet/python-repeatable-iterable/overview/main
125
126 [CodeClimateMaintainability-badge]: https://api.codeclimate.com/v1/\
127 badges/89044bfd52999e4f07f6/maintainability
128
129 [CodeClimateM13y-package-page]: https://codeclimate.com/github/\
130 Llyaudet/python-repeatable-iterable/maintainability
131
132 [Codacy-badge]: https://app.codacy.com/project/badge/Grade/\
133 1c70116c2d714e3889606519937cb11d
134
135 [Codacy-package-page]: https://app.codacy.com/gh/Llyaudet/\
136 python-repeatable-iterable/dashboard?utm_source=gh\
137 &utm_medium=referral&utm_content=&utm_campaign=Badge_grade
138
139 [GitHub-top-language-badge]: https://img.shields.io/github/\
140 languages/top/llyaudet/python-repeatable-iterable
141
142 [GitHub-license-badge]: https://img.shields.io/github/license/\

```

```

143 llyaudet/python-repeatable-iterable
144
145 [PyPI-python-version-badge]: https://img.shields.io/pypi/pyversions/\
146 python-repeatable-iterable
147
148 [GitHub-code-size-in-bytes-badge]: https://img.shields.io/github/\
149 languages/code-size/llyaudet/python-repeatable-iterable
150

```

2.7 src/python_repeatable_iterable/__init__.py

```

1  """
2  This file is part of python-repeatable-iterable library.
3
4  python-repeatable-iterable is free software:
5  you can redistribute it and/or modify it under the terms
6  of the GNU Lesser General Public License
7  as published by the Free Software Foundation,
8  either version 3 of the License,
9  or (at your option) any later version.
10
11  python-repeatable-iterable is distributed in the hope
12  that it will be useful,
13  but WITHOUT ANY WARRANTY;
14  without even the implied warranty of
15  MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.
16  See the GNU Lesser General Public License for more details.
17
18  You should have received a copy of
19  the GNU Lesser General Public License
20  along with python-repeatable-iterable.
21  If not, see <https://www.gnu.org/licenses/>.
22
23  ©Copyright 2023–2024 Laurent Lyaudet
24  -----
25  from typing import Iterable, NewType, Type
26  from _collections_abc import dict_keys, dict_values, dict_items
27  from python_none_objects import NoneIterable
28
29  A first attempt at defining the RepeatableIterable type,
30  but it is not generic.
31  It defines a RepeatableIterable() function that cannot be subscripted.
32  RepeatableIterable = NewType("RepeatableIterable", Iterable)
33
34  -----
35

```

```

36 The following function is a first attempt
37 that conveys the intent more clearly.
38 But it is not safe, see discussion just after.
39 def get_repeatable_iterable(
40     iterable: Iterable,
41     safe_classes: Iterable[Type] = NoneIterable,
42 ) -> RepeatableIterable:
43     if isinstance(
44         iterable,
45         (
46             list,
47             tuple,
48             range,
49             str,
50             bytes,
51             bytearray,
52             memoryview,
53             set,
54             frozenset,
55             dict,
56             dict_keys,
57             dict_values,
58             dict_items,
59         ),
60     ):
61         return iterable
62     if isinstance(iterable, safe_classes):
63         return iterable
64     return list(iterable)
65
66 Indeed this function is not safe, since you can subclass builtins
67 or other classes to make them not RepeatableIterable
68 from the point of view of the semantic of this type.
69 Consider the following code for example:
70 >>> class MySet(set):
71 ...     def __init__(self, *args, **kwargs):
72 ...         super().__init__(*args, **kwargs)
73 ...         self.iteration_count = 0
74 ...     def __iter__(self):
75 ...         self.iteration_count += 1
76 ...         if self.iteration_count == 1:
77 ...             return super().__iter__()
78 ...         return ().__iter__()
79 ...
80 >>> s = MySet('abcd')
81 >>> for x in s: print(x)

```



```

82  ...
83  b
84  a
85  d
86  c
87  >>> for x in s: print(x)
88  ...
89  >>> for x in s: print(x)
90  ...
91  >>> isinstance(s, set)
92  True
93
94  See here a list of builtins that can be subclassed or not:
95  https://stackoverflow.com/questions/10061752
96  /which-classes-cannot-be-subclassed
97
98  -----
99
100 This second attempt has been included in the class RepeatablyIterable.
101 def get_repeatably_iterable(
102     iterable: Iterable,
103     safe_classes: Iterable[Type] = NoneIterable,
104 ) -> RepeatablyIterable:
105     # Here is an implementation avoiding the previous problem.
106     iterable_type = type(iterable)
107     for some_class in (
108         list,
109         tuple,
110         range,
111         str,
112         bytes,
113         bytearray,
114         memoryview,
115         set,
116         frozenset,
117         dict,
118         dict_keys,
119         dict_values,
120         dict_items,
121         *safe_classes,
122     ):
123         if iterable_type is some_class:
124             return iterable
125     return list(iterable)
126 """
127

```

```

128 from typing import Iterable, Iterator, TypeVar, cast
129 from _collections_abc import dict_items, dict_keys, dict_values
130
131 from python_none_objects import NoneIterable
132
133 T = TypeVar("T")
134
135
136 class RepeatableIterable(Iterable[T]):
137     """
138     An abstract class that is here to define a type and
139     cast other objects to this type if possible in its __new__ method.
140     """
141
142     # pylint: disable-next=non-iterator-returned
143     def __iter__(self) -> Iterator[T]:
144         # Instances of RepeatableIterable don't actually exist.
145         return NotImplemented
146
147     def __new__(
148         cls,
149         iterable: Iterable[T],
150         safe_classes: Iterable[type[object]] = NoneIterable,
151     ) -> "RepeatableIterable[T]":
152         """
153         Here is an implementation avoiding the previous problem.
154         """
155         iterable_type = type(iterable)
156         for some_class in (
157             list,
158             tuple,
159             range,
160             str,
161             bytes,
162             bytearray,
163             memoryview,
164             set,
165             frozenset,
166             dict,
167             dict_keys,
168             dict_values,
169             dict_items,
170             *safe_classes,
171         ):
172             if iterable_type is some_class:
173                 return cast("RepeatableIterable[T]", iterable)

```

```
174         return cast("RepeatableIterable[T]", list(iterable))
```

2.8 src/python_repeatable_iterable/py.typed

2.9 typing_test/__init__.py

```
1  """
2  This file is part of python-repeatable-iterable library.
3
4  python-repeatable-iterable is free software:
5  you can redistribute it and/or modify it under the terms
6  of the GNU Lesser General Public License
7  as published by the Free Software Foundation,
8  either version 3 of the License,
9  or (at your option) any later version.
10
11  python-repeatable-iterable is distributed in the hope
12  that it will be useful,
13  but WITHOUT ANY WARRANTY;
14  without even the implied warranty of
15  MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.
16  See the GNU Lesser General Public License for more details.
17
18  You should have received a copy of
19  the GNU Lesser General Public License
20  along with python-repeatable-iterable.
21  If not, see <https://www.gnu.org/licenses/>.
22
23  ©Copyright 2023–2024 Laurent Lyaudet
24  """
25
26  import sys
27  from typing import Iterable, List, Never, TypeVar
28
29  sys.path.insert(0, "../src/")
30  # pylint: disable-next=wrong-import-position
31  from python_repeatable_iterable import RepeatableIterable
32
33  # pylint: disable-next=invalid-name
34  T1 = TypeVar("T1")
35
36
37  def test_arg_to_return_typing(
38      x: RepeatableIterable[List[T1]],
39  ) -> List[T1]:
```

```

40     """
41     Check that mypy follows the types
42     between the argument and the return of the function
43     for the type of the content of the list.
44     """
45     result = []
46     for y in x:
47         result.extend(y)
48     for y in x:
49         result.extend(y)
50     return result
51
52
53 def test_arg_to_return_via_call_typing(
54     x: Iterable[List[T1]],
55 ) -> List[T1]:
56     """
57     Check that mypy follows the types
58     between the argument and the return of the function
59     for the type of the content of the list
60     with indirections.
61     """
62     return test_arg_to_return_typing(RepeatableIterable(x))
63
64
65 a: List[List[Never]] = [[], []]
66 print(test_arg_to_return_via_call_typing(a))
67
68 b = (x for x in a)
69 print(test_arg_to_return_via_call_typing(b))
70
71
72 def test_arg_to_return_via_cast_typing(
73     x: RepeatableIterable[List[T1]],
74 ) -> RepeatableIterable[T1]:
75     """
76     Check that mypy follows the types
77     between the argument and the return of the function
78     for the type of the content of the list
79     with a final cast.
80     """
81     result = []
82     for y in x:
83         result.extend(y)
84     for y in x:
85         result.extend(y)

```

```
86         return RepeatableIterable(result)
```

2.10 wget_sha512.sh

```
1  #!/usr/bin/env bash
2  # This file is part of DevOrSysAdminScripts library.
3  #
4  # DevOrSysAdminScripts is free software:
5  # you can redistribute it and/or modify it under the terms
6  # of the GNU Lesser General Public License
7  # as published by the Free Software Foundation,
8  # either version 3 of the License,
9  # or (at your option) any later version.
10 #
11 # DevOrSysAdminScripts is distributed in the hope
12 # that it will be useful,
13 # but WITHOUT ANY WARRANTY;
14 # without even the implied warranty of
15 # MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.
16 # See the GNU Lesser General Public License for more details.
17 #
18 # You should have received a copy of
19 # the GNU Lesser General Public License
20 # along with DevOrSysAdminScripts.
21 # If not, see <https://www.gnu.org/licenses/>.
22 #
23 # ©Copyright 2023–2024 Laurent Lyaudet
24
25 wget_sha512() {
26     # $1 filename
27     # $2 download_URL
28     # $3 correct_sha512
29     if [[ ! -f "$1" ]];
30     then
31         wget -O "$1" "$2"
32     fi
33     present_sha512=$(sha512sum "$1" | cut -f1 -d' ')
34     if [[ "$present_sha512" != "$3" ]];
35     then
36         echo "$1 does not have correct sha512"
37         echo "wanted $3"
38         echo "found $present_sha512"
39         exit
40     fi
41 }
```

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