checkon

Release 0.1.4

Sep 27, 2019

Contents

1 Overview **1**
- 1.1 Installation . . . 1
- 1.2 Documentation . . . 2

2 Installation **3**

3 Usage **5**

4 Reference **9**
- 4.1 checkon package . . . 9

5 Contributing **15**
- 5.1 Bug reports . . . 15
- 5.2 Documentation improvements . . . 15
- 5.3 Feature requests and feedback . . . 15
- 5.4 Development . . . 16

6 Authors **19**

7 Changelog **21**
- 7.1 0.1.0 (2019-09-07) . . . 21

8 Indices and tables **23**

Python Module Index **25**

Index **27**

CHAPTER 1

Overview

docs	
tests	
package	

Checkon is a tool to help library maintainers ensure backward-compatibilty by running downstream applications' test suites with library pre-release versions.

Supported meta-runners:

- tox

Supported test-runners:

- PyTest
- Trial

Currently missing support for:

- unittest
- nose

1.1 Installation

```
pip install checkon
```

1.2 Documentation

https://checkon.readthedocs.io/

CHAPTER 2

Installation

At the command line:

```
pip install checkon
```

CHAPTER 3

Usage

I maintain a library `lib1`. `lib1` is used as a dependency by other libraries, including `lib2`. I want to make some changes to `lib1`, but before releasing, I want to make sure it won't break the code of my users in `lib2`.

So I want to run `lib2`'s test suite on the new version of `lib1`.

```
$ checkon test \
--upstream-new ../lib1 \
--upstream-base git+https://github.com/metatooling/lib1.git@master \
dependents https://github.com/metatooling/lib2.git
```

Checkon will clone `lib2`, run its test suite via `tox`, and show if there are any failures in the version on my branch specified by `--upstream-new` that pass under the `master` version on GitHub specified by `--upstream-base`. (Expand table.)

```
envname    application                          classname        name          line
↪provider                                      message
↪                          text
---------  -----------------------------------  ---------------  ----------  ------  -
↪---------------------------------------  ---------------------------------------
↪----------------------  ------------------------------------------------------------
↪---------------------
py37       https://github.com/metatooling/lib2  tests.test_lib2  test_three       7
↪git+https://github.com/metatooling/lib1.git  TypeError: add() takes 2 positional
↪arguments but 3 were given  def test_three():
↪
↪                            >       assert lib2.app.add_args([1, 2, 3]) == 6
↪
↪                            tests/test_lib2.py:9:
↪
↪                            _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _
↪_ _ _ _ _ _ _ _ _ _ _ _ _
```

(continues on next page)

(continued from previous page)

```
↪                                                                                ␣
↪                                                                                ␣
↪                            args = [1, 2, 3]

↪                                                                                ␣
↪                                                                                ␣
↪                                def add_args(args: t.List[int]) -> int:

↪                                                                                ␣
↪                                                                                ␣
↪                            >       return lib1.app.add(*args)

↪                                                                                ␣
↪                                                                                ␣
↪                            E       TypeError: add() takes 2 positional arguments␣
↪but 3 were given

↪                                                                                ␣
↪                                                                                ␣
↪                            src/lib2/app.py:7: TypeError
```

Suppose I'm contributing to a popular project like attrs. I can retrieve a list of projects depending on it from the web:

```
$ checkon list dependents-from-librariesio --limit=5 attrs
https://github.com/pytest-dev/pytest
https://github.com/Julian/jsonschema
https://github.com/twisted/twisted
https://github.com/HypothesisWorks/hypothesis
https://github.com/pypa/packaging
```

And I can run all their tests using my forked version of `attrs`.

```
$ checkon test \
--upstream-new ../attrs \
--upstream-base git+https://github.com/python-attrs/attrs \
dependents-from-librariesio --limit=5 attrs
```

Or pick test suites in a configuration file. The file can specify repositories and tox environments to run.

```
# dependents.toml
[[dependents]]
repository = "https://github.com/Julian/jsonschema"
toxenv_regex = "py37"

[[dependents]]
repository = "https://github.com/twisted/twisted"
toxenv_regex = "py37"
```

```
$ checkon test \
--upstream-new ../attrs \
--upstream-base git+https://github.com/python-attrs/attrs \
dependents-from-file ./dependents.toml
```

I can check all the pull requests in the `attrs` repository against specified dependents.

```
$ checkon test \
--output-format=json \
```

(continues on next page)

(continued from previous page)

```
--upstream-pull-requests https://github.com/python-attrs/attrs \
--upstream-base git+https://github.com/python-attrs/attrs@master \
dependents-from-file dependents.toml
```

Or check `master` against dependents, relative to the latest release.

```
$ checkon test \
--output-format=json \
--upstream-new https://github.com/python-attrs/attrs@master \
--upstream-base git+https://github.com/python-attrs/attrs@19.1.0 \
dependents-from-file dependents.toml
```

CHAPTER 4

Reference

4.1 checkon package

4.1.1 Submodules

4.1.2 checkon.app module

class checkon.app.**Dependent**(*repository: str*, *toxenv_regex: str*)
Bases: object

class checkon.app.**GitRepo**(*url*, *project: checkon.app.Project*)
Bases: object

class checkon.app.**Project**(*test_command=['tox']*)
Bases: object

checkon.app.**get_dependents**(*pypi_name*, *api_key*, *limit*)

checkon.app.**get_pull_requests**(*url: hyperlink._url.URL*) → List[str]

checkon.app.**install_hooks**(*module: str*)

Parameters **module** – The module to insert.

checkon.app.**resolve_upstream**(*upstream*)
Resolve local requirements path.

checkon.app.**run_many**(*dependents: List[checkon.app.Dependent], upstream: str, log_file*)

checkon.app.**run_one**(*dependent*, *upstream: str*, *log_file*)

checkon.app.**run_toxenv**(*dependent: checkon.app.Dependent*, *toxenv: str*, *upstream: str*)

checkon.app.**test**(*dependents: List[checkon.app.Dependent], upstream_new: List[str], upstream_pull_requests: str, upstream_base: str, log_file*)

4.1.3 checkon.cli module

checkon.cli.**compare_cli**(*dependents_lists*, *output_format*, *log_file*, ***kw*)

checkon.cli.**make_config**(*dependents*)

checkon.cli.**read_from_file**(*file*)

checkon.cli.**run_cli**(*dependents_lists*, ***kw*)

4.1.4 checkon.results module

class checkon.results.**AppSuiteRun**(*upstreamed: str, dependent_result: checkon.results.DependentResult*)

Bases: object

class checkon.results.**Comparison**(*base_requirement: str, new_requirement: str, base_failures: FrozenSet[checkon.results.FailedTest], new_failures: FrozenSet[checkon.results.FailedTest]*)

Bases: object

class checkon.results.**DependentResult**(*url: str, suite_runs: List[checkon.results.ToxTestSuiteRun]*)

Bases: object

classmethod from_dir(*output_dir*, *url*)

class checkon.results.**FailedTest**(*name: str, classname: str, file: Optional[str], line: Optional[str], failure*)

Bases: object

classmethod from_test_case(*test*)

class checkon.results.**Failure**(*message: str, lines: List[str]*)

Bases: object

classmethod from_dict(*data*)

class checkon.results.**FailureField**(**, default: Any = <marshmallow.missing>, missing: Any = <marshmallow.missing>, data_key: str = None, attribute: str = None, validate: Union[Callable[[Any], Any], Sequence[Callable[[Any], Any]], Generator[Callable[[Any], Any], None, None]] = None, required: bool = False, allow_none: bool = None, load_only: bool = False, dump_only: bool = False, error_messages: Dict[str, str] = None, **metadata*)

Bases: marshmallow.fields.Field

class checkon.results.**TestCaseRun**(*name: str, classname: str, file: Union[str, NoneType], line: Union[int, NoneType], time: str, failure: Union[checkon.results.Failure, NoneType] = None, skipped: Any = None, system_err: Union[Any, NoneType] = None*)

Bases: object

failure = None

skipped = None

system_err = None

class checkon.results.**TestSuiteRun**(*errors: int, failures: int, tests: int, time: str, timestamp: Union[datetime.datetime, NoneType], hostname: Union[str, NoneType], name: Union[str, NoneType], test_cases: List[checkon.results.TestCaseRun], envname: Union[str, NoneType], skipped: Union[int, NoneType] = None*)

Bases: object

classmethod from_bytes(*data*, *envname*)

classmethod from_path(*path*)

skipped = None

class checkon.results.**ToxTestSuiteRun**(*suite: checkon.results.TestSuiteRun*, *tox_run: checkon.tox.ToxRun*, *envname: str*)

Bases: object

A toxenv result.

classmethod from_dir(*toxenv_dir*)

checkon.results.**format_comparison**(*comparison*)

checkon.results.**format_suite_failures**(*requirement*, *test_cases*)

4.1.5 checkon.satests module

class checkon.satests.**Application**(***kwargs*)

Bases: sqlalchemy.ext.declarative.api.Base

application_id

name

class checkon.satests.**Database**(*engine: Any*, *session: Any*, *cache: Dict[KT, VT] = NOTHING*)

Bases: object

classmethod from_string(*connection_string='sqlite:///:memory:'*, *echo=False*)

init()

transform(*result: object*)

class checkon.satests.**FailureOutput**(***kwargs*)

Bases: sqlalchemy.ext.declarative.api.Base

failure_output_id

message

text

class checkon.satests.**Provider**(***kwargs*)

Bases: sqlalchemy.ext.declarative.api.Base

provider_id

requirement

class checkon.satests.**TestCase**(***kwargs*)

Bases: sqlalchemy.ext.declarative.api.Base

classname

file

line

name

test_case_id

class checkon.satests.**TestCaseRun**(***kwargs*)
Bases: sqlalchemy.ext.declarative.api.Base

duration

test_case

test_case_id

test_case_run_id

test_failure

test_failure_id

test_suite_run

test_suite_run_id

class checkon.satests.**TestFailure**(***kwargs*)
Bases: sqlalchemy.ext.declarative.api.Base

failure_output

failure_output_id

test_failure_id

class checkon.satests.**TestSuite**(***kwargs*)
Bases: sqlalchemy.ext.declarative.api.Base

test_cases

test_cases_id

test_suite_id

class checkon.satests.**TestSuiteRun**(***kwargs*)
Bases: sqlalchemy.ext.declarative.api.Base

duration

envname

start_time

test_case_runs

test_suite_run_id

class checkon.satests.**ToxRun**(***kwargs*)
Bases: sqlalchemy.ext.declarative.api.Base

application

provider

tox_run_id

toxenv_runs

class checkon.satests.**Toxenv**(***kwargs*)
Bases: sqlalchemy.ext.declarative.api.Base

application

application_id

name

toxenv_id

class checkon.satests.**ToxenvRun**(***kwargs*)
Bases: sqlalchemy.ext.declarative.api.Base

envname

start_time

test_suite_run

test_suite_run_id

tox_run

tox_run_id

toxenv

toxenv_id

toxenv_run_id

checkon.satests.**compare**(*db*)

checkon.satests.**insert_result**(*db: checkon.satests.Database*, *result: checkon.results.DependentResult*)

checkon.satests.**relation**(*cls=None*, *name=None*)

checkon.satests.**singledispatch_method**(*func*)
Singledispatch on second argument, i.e. the one that isn't *self*.

4.1.6 checkon.tests module

class checkon.tests.**Application**(*name: str*)
Bases: object

class checkon.tests.**FailureOutput**(*message: str, lines: List[str]*)
Bases: object

class checkon.tests.**Provider**(*requirement: str*)
Bases: object

class checkon.tests.**ProviderApplicationToxEnvRun**(*provider: checkon.tests.Provider*, *application: checkon.tests.Application*, *toxenv_run: checkon.tests.ToxEnvRun*)
Bases: object

class checkon.tests.**TestCase**(*name: str*, *classname: str*, *file: str*, *line: int*, *skipped: Any*)
Bases: object

class checkon.tests.**TestCaseRun**(*duration: str*, *test_case: checkon.tests.TestCase*)
Bases: object

class checkon.tests.**TestFailure**(*output: checkon.tests.FailureOutput*, *test_case_run: checkon.tests.TestCaseRun*)
Bases: object

class checkon.tests.**TestSuite**(*test_cases: List[checkon.tests.TestCase]*)
Bases: object

class checkon.tests.**TestSuiteRun**(*test_suite: checkon.tests.TestSuite*, *start_time: datetime.datetime*, *duration: Any*)
Bases: object

class checkon.tests.**ToxEnv**(*name: str*, *application: checkon.tests.Application*)
Bases: object

class checkon.tests.**ToxEnvRun**(*toxenv: checkon.tests.ToxEnv*, *test_suite_run: checkon.tests.TestSuiteRun*, *start_time: datetime.datetime*)
Bases: object

class checkon.tests.**ToxRun**(*toxenv_runs: List[checkon.tests.ToxEnvRun]*)
Bases: object

4.1.7 checkon.tox module

class checkon.tox.**Python**(*is_64: bool, version_info: Tuple[int, int, int, str, int], executable: str, name: str, sysplatform: str, version: str*)
Bases: object

class checkon.tox.**Setup**(*retcode: int*, *output: str*, *command*)
Bases: object

class checkon.tox.**Test**(*retcode: int*, *output: str*, *command*)
Bases: object

class checkon.tox.**TestEnv**(*test: List[checkon.tox.Test], installed_packages, python: checkon.tox.Python, setup: List[checkon.tox.Setup], name: Optional[str]*)
Bases: object

classmethod from_dict(*data*, *name*)

class checkon.tox.**ToxRun**(*toxversion: str*, *commands*, *platform: str*, *host: str*, *testenvs*, *reportversion: str*)
Bases: object

classmethod from_path(*path*)

class checkon.tox.**VersionInfo**(*major: int*, *minor: int*, *micro: int*, *releaselevel: str*, *serial: int*)
Bases: object

classmethod from_tuple(*tup*)

4.1.8 Module contents

CHAPTER 5

Contributing

Contributions are welcome, and they are greatly appreciated! Every little bit helps, and credit will always be given.

5.1 Bug reports

When reporting a bug please include:

- Your operating system name and version.
- Any details about your local setup that might be helpful in troubleshooting.
- Detailed steps to reproduce the bug.

5.2 Documentation improvements

checkon could always use more documentation, whether as part of the official checkon docs, in docstrings, or even on the web in blog posts, articles, and such.

5.3 Feature requests and feedback

The best way to send feedback is to file an issue at https://github.com/metatooling/checkon/issues.

If you are proposing a feature:

- Explain in detail how it would work.
- Keep the scope as narrow as possible, to make it easier to implement.
- Remember that this is a volunteer-driven project, and that code contributions are welcome :)

5.4 Development

To set up *checkon* for local development:

1. Fork checkon (look for the "Fork" button).
2. Clone your fork locally:

```
git clone git@github.com:your_name_here/checkon.git
```

3. Create a branch for local development:

```
git checkout -b name-of-your-bugfix-or-feature
```

   Now you can make your changes locally.

4. When you're done making changes, run all the checks, doc builder and spell checker with tox one command:

```
tox
```

5. Commit your changes and push your branch to GitHub:

```
git add .
git commit -m "Your detailed description of your changes."
git push origin name-of-your-bugfix-or-feature
```

6. Submit a pull request through the GitHub website.

5.4.1 Pull Request Guidelines

If you need some code review or feedback while you're developing the code just make the pull request.

For merging, you should:

1. Include passing tests (run `tox`)[1].
2. Update documentation when there's new API, functionality etc.
3. Add a file in `changelog.d/` describing the changes. The filename should be `{id}.{type}.rst`, where `{id}` is the number of the GitHub issue or pull request and `{type}` is one of `breaking` (for breaking changes), `deprecation` (for deprecations), or `change` (for non-breaking changes). For example, to add a new feature requested in GitHub issue #1234, add a file called `changelog.d/1234.change.rst` describing the change.
4. Add yourself to `AUTHORS.rst`.

5.4.2 Tips

To run a subset of tests:

```
tox -e envname -- pytest -k test_myfeature
```

To run all the test environments in *parallel* (you need to `pip install detox`):

[1] If you don't have all the necessary python versions available locally you can rely on Travis - it will run the tests for each change you add in the pull request.
It will be slower though ...

```
detox
```

CHAPTER 6

Authors

- Checkon contributors - https://github.com/metatooling/checkon

CHAPTER 7

Changelog

7.1 0.1.0 (2019-09-07)

7.1.1 Changes

- First release on PyPI.

—

CHAPTER 8

Indices and tables

- genindex
- modindex
- search

Python Module Index

C

checkon, 14
checkon.app, 9
checkon.cli, 10
checkon.results, 10
checkon.satests, 11
checkon.tests, 13
checkon.tox, 14

Index

A

application (*checkon.satests.Toxenv attribute*), 12
application (*checkon.satests.ToxRun attribute*), 12
Application (*class in checkon.satests*), 11
Application (*class in checkon.tests*), 13
application_id (*checkon.satests.Application attribute*), 11
application_id (*checkon.satests.Toxenv attribute*), 13
AppSuiteRun (*class in checkon.results*), 10

C

checkon (*module*), 14
checkon.app (*module*), 9
checkon.cli (*module*), 10
checkon.results (*module*), 10
checkon.satests (*module*), 11
checkon.tests (*module*), 13
checkon.tox (*module*), 14
classname (*checkon.satests.TestCase attribute*), 11
compare() (*in module checkon.satests*), 13
compare_cli() (*in module checkon.cli*), 10
Comparison (*class in checkon.results*), 10

D

Database (*class in checkon.satests*), 11
Dependent (*class in checkon.app*), 9
DependentResult (*class in checkon.results*), 10
duration (*checkon.satests.TestCaseRun attribute*), 12
duration (*checkon.satests.TestSuiteRun attribute*), 12

E

envname (*checkon.satests.TestSuiteRun attribute*), 12
envname (*checkon.satests.ToxenvRun attribute*), 13

F

FailedTest (*class in checkon.results*), 10
failure (*checkon.results.TestCaseRun attribute*), 10
Failure (*class in checkon.results*), 10
failure_output (*checkon.satests.TestFailure attribute*), 12
failure_output_id (*checkon.satests.FailureOutput attribute*), 11
failure_output_id (*checkon.satests.TestFailure attribute*), 12
FailureField (*class in checkon.results*), 10
FailureOutput (*class in checkon.satests*), 11
FailureOutput (*class in checkon.tests*), 13
file (*checkon.satests.TestCase attribute*), 11
format_comparison() (*in module checkon.results*), 11
format_suite_failures() (*in module checkon.results*), 11
from_bytes() (*checkon.results.TestSuiteRun class method*), 11
from_dict() (*checkon.results.Failure class method*), 10
from_dict() (*checkon.tox.TestEnv class method*), 14
from_dir() (*checkon.results.DependentResult class method*), 10
from_dir() (*checkon.results.ToxTestSuiteRun class method*), 11
from_path() (*checkon.results.TestSuiteRun class method*), 11
from_path() (*checkon.tox.ToxRun class method*), 14
from_string() (*checkon.satests.Database class method*), 11
from_test_case() (*checkon.results.FailedTest class method*), 10
from_tuple() (*checkon.tox.VersionInfo class method*), 14

G

get_dependents() (*in module checkon.app*), 9
get_pull_requests() (*in module checkon.app*), 9
GitRepo (*class in checkon.app*), 9

I

init() (*checkon.satests.Database method*), 11

insert_result() (*in module checkon.satests*), 13
install_hooks() (*in module checkon.app*), 9

L

line (*checkon.satests.TestCase attribute*), 11

M

make_config() (*in module checkon.cli*), 10
message (*checkon.satests.FailureOutput attribute*), 11

N

name (*checkon.satests.Application attribute*), 11
name (*checkon.satests.TestCase attribute*), 12
name (*checkon.satests.Toxenv attribute*), 13

P

Project (*class in checkon.app*), 9
provider (*checkon.satests.ToxRun attribute*), 12
Provider (*class in checkon.satests*), 11
Provider (*class in checkon.tests*), 13
provider_id (*checkon.satests.Provider attribute*), 11
ProviderApplicationToxEnvRun (*class in checkon.tests*), 13
Python (*class in checkon.tox*), 14

R

read_from_file() (*in module checkon.cli*), 10
relation() (*in module checkon.satests*), 13
requirement (*checkon.satests.Provider attribute*), 11
resolve_upstream() (*in module checkon.app*), 9
run_cli() (*in module checkon.cli*), 10
run_many() (*in module checkon.app*), 9
run_one() (*in module checkon.app*), 9
run_toxenv() (*in module checkon.app*), 9

S

Setup (*class in checkon.tox*), 14
singledispatch_method() (*in module checkon.satests*), 13
skipped (*checkon.results.TestCaseRun attribute*), 10
skipped (*checkon.results.TestSuiteRun attribute*), 11
start_time (*checkon.satests.TestSuiteRun attribute*), 12
start_time (*checkon.satests.ToxenvRun attribute*), 13
system_err (*checkon.results.TestCaseRun attribute*), 10

T

Test (*class in checkon.tox*), 14
test() (*in module checkon.app*), 9
test_case (*checkon.satests.TestCaseRun attribute*), 12
test_case_id (*checkon.satests.TestCase attribute*), 12
test_case_id (*checkon.satests.TestCaseRun attribute*), 12
test_case_run_id (*checkon.satests.TestCaseRun attribute*), 12
test_case_runs (*checkon.satests.TestSuiteRun attribute*), 12
test_cases (*checkon.satests.TestSuite attribute*), 12
test_cases_id (*checkon.satests.TestSuite attribute*), 12
test_failure (*checkon.satests.TestCaseRun attribute*), 12
test_failure_id (*checkon.satests.TestCaseRun attribute*), 12
test_failure_id (*checkon.satests.TestFailure attribute*), 12
test_suite_id (*checkon.satests.TestSuite attribute*), 12
test_suite_run (*checkon.satests.TestCaseRun attribute*), 12
test_suite_run (*checkon.satests.ToxenvRun attribute*), 13
test_suite_run_id (*checkon.satests.TestCaseRun attribute*), 12
test_suite_run_id (*checkon.satests.TestSuiteRun attribute*), 12
test_suite_run_id (*checkon.satests.ToxenvRun attribute*), 13
TestCase (*class in checkon.satests*), 11
TestCase (*class in checkon.tests*), 13
TestCaseRun (*class in checkon.results*), 10
TestCaseRun (*class in checkon.satests*), 12
TestCaseRun (*class in checkon.tests*), 13
TestEnv (*class in checkon.tox*), 14
TestFailure (*class in checkon.satests*), 12
TestFailure (*class in checkon.tests*), 13
TestSuite (*class in checkon.satests*), 12
TestSuite (*class in checkon.tests*), 13
TestSuiteRun (*class in checkon.results*), 10
TestSuiteRun (*class in checkon.satests*), 12
TestSuiteRun (*class in checkon.tests*), 14
text (*checkon.satests.FailureOutput attribute*), 11
tox_run (*checkon.satests.ToxenvRun attribute*), 13
tox_run_id (*checkon.satests.ToxenvRun attribute*), 13
tox_run_id (*checkon.satests.ToxRun attribute*), 12
toxenv (*checkon.satests.ToxenvRun attribute*), 13
Toxenv (*class in checkon.satests*), 12
ToxEnv (*class in checkon.tests*), 14
toxenv_id (*checkon.satests.Toxenv attribute*), 13
toxenv_id (*checkon.satests.ToxenvRun attribute*), 13
toxenv_run_id (*checkon.satests.ToxenvRun attribute*), 13
toxenv_runs (*checkon.satests.ToxRun attribute*), 12
ToxenvRun (*class in checkon.satests*), 13
ToxEnvRun (*class in checkon.tests*), 14

ToxRun (*class in checkon.satests*), 12
ToxRun (*class in checkon.tests*), 14
ToxRun (*class in checkon.tox*), 14
ToxTestSuiteRun (*class in checkon.results*), 11
transform() (*checkon.satests.Database method*), 11

V

VersionInfo (*class in checkon.tox*), 14