Unittest for `caching`

September 22, 2024

Contents

1 Test Information

1.1 Test Candidate Information

The Module caching is designed to store information in json or pickle files to support them much faster then generating them from the original source file. For more Information read the documentation.

Library Information	
Name	caching
State	Released
Supported Interpreters	python3
Version	df0033510331110d2a749ab2e6ec5f9f
Dependencies	

1.2 Unittest Information

Unittest Information	
Version	ac6e9667753d32025048abc5366ddb10
Testruns with	python 3.11.2 (final)

1.3 Test System Information

System Information	
Architecture	64bit
Distribution	Debian GNU/Linux 12 bookworm
Hostname	ahorn
Kernel	6.1.0-17-amd64 (#1 SMP PREEMPT_DYNAMIC Debian 6.1.69-1 (2023-12-30))
Machine	x86_64
Path	/home/dirk/my_repositories/unittest/caching
System	Linux
Username	dirk

2 Statistic

2.1 Test-Statistic for testrun with python 3.11.2 (final)

Number of tests	**20**
Number of successfull tests	**20**
Number of possibly failed tests	**0**
Number of failed tests	**0**
Executionlevel	Full Test (all defined tests)
Time consumption	0.041s

2.2 Coverage Statistic

Module- or Filename	Line-Coverage	Branch-Coverage
caching	89.4%	79.3%
caching.__init__.py	89.4%	

3 Testcases with no corresponding Requirement

3.1 Summary for testrun with python 3.11.2 (final)

3.1.1 caching.property_cache_json: Test cached data (full init)

Testresult
This test was passed with the state: **Success**. See also full trace in section A.1.3!

Testrun:	python 3.11.2 (final)
Caller:	/home/dirk/my_repositories/unittest/caching/unittest/src/report/__init__.py (323)
Start-Time:	2024-09-22 12:01:39,830
Finished-Time:	2024-09-22 12:01:39,832
Time-Consumption	0.002s

Testsummary:

Info	Initialising property_cache_json (load_all_on_init=True).
Info	Collecting data from cache instance.
Success	Cached data is correct (Content {'str': 'string', 'unicode': 'unicode', 'integer': 17, 'float': 3.14159, 'list': [1, 'two', '3', 4], 'dict': {'1': 1, '2': 'two', '3': '3', '4': 4}, 'uncached': 'uncached_data_of_class'} and Type is <class 'dict'>).

3.1.2 caching.property_cache_json: Test cached data (partially init)

Testresult
This test was passed with the state: **Success**. See also full trace in section A.1.4!

Testrun:	python 3.11.2 (final)
Caller:	/home/dirk/my_repositories/unittest/caching/unittest/src/report/__init__.py (323)
Start-Time:	2024-09-22 12:01:39,832
Finished-Time:	2024-09-22 12:01:39,834
Time-Consumption	0.002s

Testsummary:

Info	Initialising property_cache_json (load_all_on_init=True).
Info	Collecting data from cache instance.
Success	Cached data is correct (Content {'str': 'string', 'unicode': 'unicode', 'integer': 17, 'float': 3.14159, 'list': [1, 'two', '3', 4], 'dict': {'1': 1, '2': 'two', '3': '3', '4': 4}, 'uncached': 'uncached_data_of_class'} and Type is <class 'dict'>).

3.1.3 caching.property_cache_json: Test execution of save callback (full init)

Testresult
This test was passed with the state: **Success**. See also full trace in section A.1.9!

Testrun:	python 3.11.2 (final)
Caller:	/home/dirk/my_repositories/unittest/caching/unittest/src/report/__init__.py (323)

Start-Time:	2024-09-22 12:01:39,849
Finished-Time:	2024-09-22 12:01:39,850
Time-Consumption	0.001s

Testsummary:

Info	Installing save_callback, which sets a variable to True on execution.
Success	Save callback execution variable is correct (Content True and Type is <class 'bool'>).

3.1.4 caching.property_cache_json: Test full initialised JSON-Cache-Object

Testresult

This test was passed with the state: **Success**. See also full trace in section A.1.1!

Testrun:	python 3.11.2 (final)
Caller:	/home/dirk/my_repositories/unittest/caching/unittest/src/report/__init__.py (323)
Start-Time:	2024-09-22 12:01:39,825
Finished-Time:	2024-09-22 12:01:39,827
Time-Consumption	0.002s

Testsummary:

Info	Initialising property_cache_json (load_all_on_init=True).
Info	Extracting storage object from property_cache_json for comparison.
Success	Cache object is correct (Content {'dict': {'1': 1, '2': 'two', '3': '3', '4': 4}, 'float': 3.14159, 'integer': 17, 'list': [1, 'two', '3', 4], 'str': 'string', 'unicode': 'unicode'} and Type is <class 'dict'>).

3.1.5 caching.property_cache_json: Test get from source caused by changed uid (full init)

Testresult

This test was passed with the state: **Success**. See also full trace in section A.1.7!

Testrun:	python 3.11.2 (final)
Caller:	/home/dirk/my_repositories/unittest/caching/unittest/src/report/__init__.py (323)
Start-Time:	2024-09-22 12:01:39,841
Finished-Time:	2024-09-22 12:01:39,844
Time-Consumption	0.003s

Testsummary:

Info	Initialising property_cache_json (load_all_on_init=True).
Success	Instance data after changing uid is correct (Content {'str': '__string__', 'unicode': '__unicode__', 'integer': 34, 'float': 2.71828, 'list': ['one', 2, 3, '4'], 'dict': {'1': '1', '2': 2, '3': 'three', '4': '4'}} and Type is <class 'dict'>).

3.1.6 caching.property_cache_json: Test get from source caused by changed uid (partially init)

Testresult

This test was passed with the state: **Success**. See also full trace in section A.1.8!

Testrun:	python 3.11.2 (final)
Caller:	/home/dirk/my_repositories/unittest/caching/unittest/src/report/__init__.py (323)
Start-Time:	2024-09-22 12:01:39,845
Finished-Time:	2024-09-22 12:01:39,848
Time-Consumption	0.004s

Testsummary:

Info	Initialising property_cache_json (load_all_on_init=True).
Success	Instance data after changing uid is correct (Content {'str': '__string__', 'unicode': '__unicode__', 'integer': 34, 'float': 2.71828, 'list': ['one', 2, 3, '4'], 'dict': {'1': '1', '2': 2, '3': 'three', '4': '4'}} and Type is <class 'dict'>).

3.1.7 caching.property_cache_json: Test get from source caused by increased data version (full init)

Testresult

This test was passed with the state: **Success**. See also full trace in section A.1.5!

Testrun:	python 3.11.2 (final)
Caller:	/home/dirk/my_repositories/unittest/caching/unittest/src/report/__init__.py (323)
Start-Time:	2024-09-22 12:01:39,834
Finished-Time:	2024-09-22 12:01:39,836
Time-Consumption	0.002s

Testsummary:

Info	Initialising property_cache_json (load_all_on_init=True).
Success	Instance data after increasing data_version is correct (Content {'str': '__string__', 'unicode': '__unicode__', 'integer': 34, 'float': 2.71828, 'list': ['one', 2, 3, '4'], 'dict': {'1': '1', '2': 2, '3': 'three', '4': '4'}} and Type is <class 'dict'>).

3.1.8 caching.property_cache_json: Test get from source caused by increased data version (partially init)

Testresult

This test was passed with the state: **Success**. See also full trace in section A.1.6!

Testrun:	python 3.11.2 (final)
Caller:	/home/dirk/my_repositories/unittest/caching/unittest/src/report/__init__.py (323)
Start-Time:	2024-09-22 12:01:39,837
Finished-Time:	2024-09-22 12:01:39,841
Time-Consumption	0.004s

Testsummary:

Info	Initialising property_cache_json (load_all_on_init=True).
Success	Instance data after increasing data_version is correct (Content {'str': '__string__', 'unicode': '__unicode__', 'integer': 34, 'float': 2.71828, 'list': ['one', 2, 3, '4'], 'dict': {'1': '1', '2': 2, '3': 'three', '4': '4'}} and Type is <class 'dict'>).

3.1.9 caching.property_cache_json: Test internal key usage

Testresult

This test was passed with the state: **Success**. See also full trace in section A.1.10!

Testrun:	python 3.11.2 (final)
Caller:	/home/dirk/my_repositories/unittest/caching/unittest/src/report/__init__.py (323)
Start-Time:	2024-09-22 12:01:39,850
Finished-Time:	2024-09-22 12:01:39,853
Time-Consumption	0.003s

Testsummary:

Info	Initialising property_cache_json (load_all_on_init=True).
Info	Extracting storage object from property_cache_json for comparison.
Success	Cache is correct (Content {'__property_cache_data_version_': 'no second data version', '__property_cache_uid_': 'no second uid', '_property_cache_data_version_': 'no data version', '_property_cache_uid_': 'no uid'} and Type is <class 'dict'>).
Success	Keyfilter returnvalue for 5 (<class 'int'>) is correct (Content 5 and Type is <class 'int'>).

3.1.10 caching.property_cache_json: Test partially initialisation of JSON-Cache-Object

Testresult

This test was passed with the state: **Success**. See also full trace in section A.1.2!

Testrun:	python 3.11.2 (final)
Caller:	/home/dirk/my_repositories/unittest/caching/unittest/src/report/__init__.py (323)
Start-Time:	2024-09-22 12:01:39,827
Finished-Time:	2024-09-22 12:01:39,830
Time-Consumption	0.003s

Testsummary:

Info	Initialising property_cache_json (load_all_on_init=False).
Info	Partially initialising cache object by requesting some information.
Info	Extracting storage object from property_cache_json for comparison.
Success	Cache object is correct (Content {'integer': 17, 'str': 'string', 'unicode': 'unicode'} and Type is <class 'dict'>).

3.1.11 caching.property_cache_pickle: Test cached data (full init)

Testresult

This test was passed with the state: **Success**. See also full trace in section A.1.13!

Testrun:	python 3.11.2 (final)
Caller:	/home/dirk/my_repositories/unittest/caching/unittest/src/report/__init__.py (323)
Start-Time:	2024-09-22 12:01:39,856
Finished-Time:	2024-09-22 12:01:39,857
Time-Consumption	0.001s

Testsummary:	
Info	Initialising property_cache_pickle (load_all_on_init=True).
Info	Collecting data from cache instance.
Success	Cached data is correct (Content {'str': 'string', 'unicode': 'unicode', 'integer': 17, 'float': 3.14159, 'list': [1, 'two', '3', 4], 'dict': {'1': 1, '2': 'two', '3': '3', '4': 4}, 'uncached': 'uncached_data_of_class'} and Type is <class 'dict'>).

3.1.12 caching.property_cache_pickle: Test cached data (partially init)

Testresult

This test was passed with the state: **Success**. See also full trace in section A.1.14!

Testrun:	python 3.11.2 (final)
Caller:	/home/dirk/my_repositories/unittest/caching/unittest/src/report/__init__.py (323)
Start-Time:	2024-09-22 12:01:39,858
Finished-Time:	2024-09-22 12:01:39,859
Time-Consumption	0.001s

Testsummary:	
Info	Initialising property_cache_pickle (load_all_on_init=True).
Info	Collecting data from cache instance.
Success	Cached data is correct (Content {'str': 'string', 'unicode': 'unicode', 'integer': 17, 'float': 3.14159, 'list': [1, 'two', '3', 4], 'dict': {'1': 1, '2': 'two', '3': '3', '4': 4}, 'uncached': 'uncached_data_of_class'} and Type is <class 'dict'>).

3.1.13 caching.property_cache_pickle: Test execution of save callback (full init)

Testresult

This test was passed with the state: **Success**. See also full trace in section A.1.19!

Testrun:	python 3.11.2 (final)
Caller:	/home/dirk/my_repositories/unittest/caching/unittest/src/report/__init__.py (323)
Start-Time:	2024-09-22 12:01:39,868
Finished-Time:	2024-09-22 12:01:39,869
Time-Consumption	0.000s

Testsummary:	
Info	Installing save_callback, which sets a variable to True on execution.
Success	Save callback execution variable is correct (Content True and Type is <class 'bool'>).

3.1.14 caching.property_cache_pickle: Test full initialised PICKLE-Cache-Object

Testresult

This test was passed with the state: **Success**. See also full trace in section A.1.11!

Testrun:	python 3.11.2 (final)

Caller:	/home/dirk/my_repositories/unittest/caching/unittest/src/report/__init__.py (323)
Start-Time:	2024-09-22 12:01:39,853
Finished-Time:	2024-09-22 12:01:39,854
Time-Consumption	0.001s

Testsummary:

Info	Initialising property_cache_pickle (load_all_on_init=True).
Info	Extracting storage object from property_cache_pickle for comparison.
Success	Cache object is correct (Content {'str': 'string', 'unicode': 'unicode', 'integer': 17, 'float': 3.14159, 'list': [1, 'two', '3', 4], 'dict': {'1': 1, '2': 'two', '3': '3', '4': 4}} and Type is <class 'dict'>).

3.1.15 caching.property_cache_pickle: Test get from source caused by changed uid (full init)

Testresult

This test was passed with the state: **Success**. See also full trace in section A.1.17!

Testrun:	python 3.11.2 (final)
Caller:	/home/dirk/my_repositories/unittest/caching/unittest/src/report/__init__.py (323)
Start-Time:	2024-09-22 12:01:39,863
Finished-Time:	2024-09-22 12:01:39,865
Time-Consumption	0.002s

Testsummary:

Info	Initialising property_cache_pickle (load_all_on_init=True).
Success	Instance data after changing uid is correct (Content {'str': '__string__', 'unicode': '__unicode__', 'integer': 34, 'float': 2.71828, 'list': ['one', 2, 3, '4'], 'dict': {'1': '1', '2': 2, '3': 'three', '4': '4'}} and Type is <class 'dict'>).

3.1.16 caching.property_cache_pickle: Test get from source caused by changed uid (partially init)

Testresult

This test was passed with the state: **Success**. See also full trace in section A.1.18!

Testrun:	python 3.11.2 (final)
Caller:	/home/dirk/my_repositories/unittest/caching/unittest/src/report/__init__.py (323)
Start-Time:	2024-09-22 12:01:39,866
Finished-Time:	2024-09-22 12:01:39,868
Time-Consumption	0.002s

Testsummary:

Info	Initialising property_cache_pickle (load_all_on_init=True).
Success	Instance data after changing uid is correct (Content {'str': '__string__', 'unicode': '__unicode__', 'integer': 34, 'float': 2.71828, 'list': ['one', 2, 3, '4'], 'dict': {'1': '1', '2': 2, '3': 'three', '4': '4'}} and Type is <class 'dict'>).

3.1.17 caching.property_cache_pickle: Test get from source caused by increased data version (full init)

Testresult

This test was passed with the state: **Success**. See also full trace in section A.1.15!

Testrun:	python 3.11.2 (final)
Caller:	/home/dirk/my_repositories/unittest/caching/unittest/src/report/__init__.py (323)
Start-Time:	2024-09-22 12:01:39,859
Finished-Time:	2024-09-22 12:01:39,861
Time-Consumption	0.002s

Testsummary:

Info	Initialising property_cache_pickle (load_all_on_init=True).
Success	Instance data after increasing data_version is correct (Content {'str': '__string__', 'unicode': '__unicode__', 'integer': 34, 'float': 2.71828, 'list': ['one', 2, 3, '4'], 'dict': {'1': '1', '2': 2, '3': 'three', '4': '4'}} and Type is <class 'dict'>).

3.1.18 caching.property_cache_pickle: Test get from source caused by increased data version (partially init)

Testresult

This test was passed with the state: **Success**. See also full trace in section A.1.16!

Testrun:	python 3.11.2 (final)
Caller:	/home/dirk/my_repositories/unittest/caching/unittest/src/report/__init__.py (323)
Start-Time:	2024-09-22 12:01:39,861
Finished-Time:	2024-09-22 12:01:39,863
Time-Consumption	0.002s

Testsummary:

Info	Initialising property_cache_pickle (load_all_on_init=True).
Success	Instance data after increasing data_version is correct (Content {'str': '__string__', 'unicode': '__unicode__', 'integer': 34, 'float': 2.71828, 'list': ['one', 2, 3, '4'], 'dict': {'1': '1', '2': 2, '3': 'three', '4': '4'}} and Type is <class 'dict'>).

3.1.19 caching.property_cache_pickle: Test internal key usage

Testresult

This test was passed with the state: **Success**. See also full trace in section A.1.20!

Testrun:	python 3.11.2 (final)
Caller:	/home/dirk/my_repositories/unittest/caching/unittest/src/report/__init__.py (323)
Start-Time:	2024-09-22 12:01:39,869
Finished-Time:	2024-09-22 12:01:39,871
Time-Consumption	0.002s

Testsummary:

Info	Initialising property_cache_pickle (load_all_on_init=True).
Info	Extracting storage object from property_cache_pickle for comparison.

Success	Cache is correct (Content {'_property_cache_uid_': 'no uid', '__property_cache_uid_': 'no second uid', '_property_cache_data_version_': 'no data version', '__property_cache_data_version_': 'no second data version'} and Type is <class 'dict'>).
Success	Keyfilter returnvalue for 5 (<class 'int'>) is correct (Content 5 and Type is <class 'int'>).

3.1.20 caching.property_cache_pickle: Test partially initialised PICKLE-Cache-Object

Testresult

This test was passed with the state: **Success**. See also full trace in section A.1.12!

Testrun:	python 3.11.2 (final)
Caller:	/home/dirk/my_repositories/unittest/caching/unittest/src/report/__init__.py (323)
Start-Time:	2024-09-22 12:01:39,855
Finished-Time:	2024-09-22 12:01:39,856
Time-Consumption	0.001s

Testsummary:

Info	Initialising property_cache_pickle (load_all_on_init=False).
Info	Partially initialising cache object by requesting some information.
Info	Extracting storage object from property_cache_pickle for comparison.
Success	Cache object is correct (Content {'integer': 17, 'str': 'string', 'unicode': 'unicode'} and Type is <class 'dict'>).

A Trace for testrun with python 3.11.2 (final)

A.1 Tests with status Info (20)

A.1.1 caching.property_cache_json: Test full initialised JSON-Cache-Object

Testresult
This test was passed with the state: **Success**.

Info	Initialising property_cache_json (load_all_on_init=True).

```
Deleting cache file from filesystem to ensure identical conditions for each test run.
```

```
Initialising cached class with
↪ /home/dirk/my_repositories/unittest/caching/unittest/output_data/load_on_init.json as
↪ cache file.
```

Info	Extracting storage object from property_cache_json for comparison.

```
Using storage object of cache class for comparison: {'dict': {'1': 1, '2': 'two', '3': '3',
↪ '4': 4}, 'float': 3.14159, 'integer': 17, 'list': [1, 'two', '3', 4], 'str': 'string',
↪ 'unicode': 'unicode'}
```

Success	Cache object is correct (Content {'dict': {'1': 1, '2': 'two', '3': '3', '4': 4}, 'float': 3.14159, 'integer': 17, 'list': [1, 'two', '3', 4], 'str': 'string', 'unicode': 'unicode'} and Type is <class 'dict'>).

```
Result (Cache object): { 'dict': { '1': 1, '2': 'two', '3': '3', '4': 4 }, 'float': 3.14159,
↪ 'integer': 17, 'list': [ 1, 'two', '3', 4 ], 'str': 'string', 'unicode': 'unicode' }
↪ (<class 'dict'>)
```

```
Expectation (Cache object): result = { 'str': 'string', 'unicode': 'unicode', 'integer': 17,
↪ 'float': 3.14159, 'list': [ 1, 'two', '3', 4 ], 'dict': { '1': 1, '2': 'two', '3': '3',
↪ '4': 4 } } (<class 'dict'>)
```

A.1.2 caching.property_cache_json: Test partially initialisation of JSON-Cache-Object

Testresult
This test was passed with the state: **Success**.

Info	Initialising property_cache_json (load_all_on_init=False).

```
Deleting cache file from filesystem to ensure identical conditions for each test run.
```

```
Initialising cached class with
↪ /home/dirk/my_repositories/unittest/caching/unittest/output_data/no_load_on_init.json as
↪ cache file.
```

Info	Partially initialising cache object by requesting some information.

Info	Extracting storage object from property_cache_json for comparison.

```
Using storage object of cache class for comparison: {'integer': 17, 'str': 'string',
↪  'unicode': 'unicode'}
```

Success	Cache object is correct (Content {'integer': 17, 'str': 'string', 'unicode': 'unicode'} and Type is <class 'dict'>).

```
Result (Cache object): { 'integer': 17, 'str': 'string', 'unicode': 'unicode' } (<class
↪  'dict'>)
```

```
Expectation (Cache object): result = { 'str': 'string', 'unicode': 'unicode', 'integer': 17 }
↪  (<class 'dict'>)
```

A.1.3 caching.property_cache_json: Test cached data (full init)

Testresult

This test was passed with the state: **Success**.

Info	Initialising property_cache_json (load_all_on_init=True).

```
Deleting cache file from filesystem to ensure identical conditions for each test run.
```

```
Initialising cached class with /home/dirk/my_repositories/unittest/caching/unittest/output_da
↪  ta/cache_data_test_load_on_init.json as cache
↪  file.
```

Info	Collecting data from cache instance.

Success	Cached data is correct (Content {'str': 'string', 'unicode': 'unicode', 'integer': 17, 'float': 3.14159, 'list': [1, 'two', '3', 4], 'dict': {'1': 1, '2': 'two', '3': '3', '4': 4}, 'uncached': 'uncached_data_of_class'} and Type is <class 'dict'>).

```
Result (Cached data): { 'str': 'string', 'unicode': 'unicode', 'integer': 17, 'float':
↪  3.14159, 'list': [ 1, 'two', '3', 4 ], 'dict': { '1': 1, '2': 'two', '3': '3', '4': 4 },
↪  'uncached': 'uncached_data_of_class' } (<class 'dict'>)
```

```
Expectation (Cached data): result = { 'str': 'string', 'unicode': 'unicode', 'integer': 17,
↪  'float': 3.14159, 'list': [ 1, 'two', '3', 4 ], 'dict': { '1': 1, '2': 'two', '3': '3',
↪  '4': 4 }, 'uncached': 'uncached_data_of_class' } (<class 'dict'>)
```

A.1.4 caching.property_cache_json: Test cached data (partially init)

Testresult
This test was passed with the state: **Success**.

Info	Initialising property_cache_json (load_all_on_init=True).

```
Deleting cache file from filesystem to ensure identical conditions for each test run.
```

```
Initialising cached class with /home/dirk/my_repositories/unittest/caching/unittest/output_da
↪ ta/cache_data_test_no_load_on_init.json as cache
↪ file.
```

Info	Collecting data from cache instance.

Success	Cached data is correct (Content {'str': 'string', 'unicode': 'unicode', 'integer': 17, 'float': 3.14159, 'list': [1, 'two', '3', 4], 'dict': {'1': 1, '2': 'two', '3': '3', '4': 4}, 'uncached': 'uncached_data_of_class'} and Type is <class 'dict'>).

```
Result (Cached data): { 'str': 'string', 'unicode': 'unicode', 'integer': 17, 'float':
↪ 3.14159, 'list': [ 1, 'two', '3', 4 ], 'dict': { '1': 1, '2': 'two', '3': '3', '4': 4 },
↪ 'uncached': 'uncached_data_of_class' } (<class 'dict'>)
```

```
Expectation (Cached data): result = { 'str': 'string', 'unicode': 'unicode', 'integer': 17,
↪ 'float': 3.14159, 'list': [ 1, 'two', '3', 4 ], 'dict': { '1': 1, '2': 'two', '3': '3',
↪ '4': 4 }, 'uncached': 'uncached_data_of_class' } (<class 'dict'>)
```

A.1.5 caching.property_cache_json: Test get from source caused by increased data version (full init)

Testresult
This test was passed with the state: **Success**.

Info	Initialising property_cache_json (load_all_on_init=True).

```
Deleting cache file from filesystem to ensure identical conditions for each test run.
```

```
Initialising cached class with /home/dirk/my_repositories/unittest/caching/unittest/output_da
↪ ta/data_version_test_load_on_init.json as cache
↪ file.
```

Success	Instance data after increasing data_version is correct (Content {'str': '__string__', 'unicode': '__unicode__', 'integer': 34, 'float': 2.71828, 'list': ['one', 2, 3, '4'], 'dict': {'1': '1', '2': 2, '3': 'three', '4': '4'}} and Type is <class 'dict'>).

```
Result (Instance data after increasing data_version): { 'str': '__string__', 'unicode':
↪ '__unicode__', 'integer': 34, 'float': 2.71828, 'list': [ 'one', 2, 3, '4' ], 'dict': {
↪ '1': '1', '2': 2, '3': 'three', '4': '4' } } (<class 'dict'>)
```

```
Expectation (Instance data after increasing data_version): result = { 'str': '__string__',
↪ 'unicode': '__unicode__', 'integer': 34, 'float': 2.71828, 'list': [ 'one', 2, 3, '4' ],
↪ 'dict': { '1': '1', '2': 2, '3': 'three', '4': '4' } } (<class 'dict'>)
```

A.1.6 caching.property_cache_json: Test get from source caused by increased data version (partially init)

Testresult
This test was passed with the state: **Success**.

Info	Initialising property_cache_json (load_all_on_init=True).

```
Deleting cache file from filesystem to ensure identical conditions for each test run.
```

```
Initialising cached class with /home/dirk/my_repositories/unittest/caching/unittest/output_da
↪ ta/data_version_test_no_load_on_init.json as cache
↪ file.
```

Success	Instance data after increasing data_version is correct (Content {'str': '__string__', 'unicode': '__unicode__', 'integer': 34, 'float': 2.71828, 'list': ['one', 2, 3, '4'], 'dict': {'1': '1', '2': 2, '3': 'three', '4': '4'}} and Type is <class 'dict'>).

```
Result (Instance data after increasing data_version): { 'str': '__string__', 'unicode':
↪ '__unicode__', 'integer': 34, 'float': 2.71828, 'list': [ 'one', 2, 3, '4' ], 'dict': {
↪ '1': '1', '2': 2, '3': 'three', '4': '4' } } (<class 'dict'>)
```

```
Expectation (Instance data after increasing data_version): result = { 'str': '__string__',
↪ 'unicode': '__unicode__', 'integer': 34, 'float': 2.71828, 'list': [ 'one', 2, 3, '4' ],
↪ 'dict': { '1': '1', '2': 2, '3': 'three', '4': '4' } } (<class 'dict'>)
```

A.1.7 caching.property_cache_json: Test get from source caused by changed uid (full init)

Testresult
This test was passed with the state: **Success**.

Info	Initialising property_cache_json (load_all_on_init=True).

```
Deleting cache file from filesystem to ensure identical conditions for each test run.
```

```
Initialising cached class with /home/dirk/my_repositories/unittest/caching/unittest/output_da
↪ ta/uid_test_load_on_init.json as cache
↪ file.
```

Success	Instance data after changing uid is correct (Content {'str': '__string__', 'unicode': '__unicode__', 'integer': 34, 'float': 2.71828, 'list': ['one', 2, 3, '4'], 'dict': {'1': '1', '2': 2, '3': 'three', '4': '4'}} and Type is <class 'dict'>).

```
Result (Instance data after changing uid): { 'str': '__string__', 'unicode': '__unicode__',
↪ 'integer': 34, 'float': 2.71828, 'list': [ 'one', 2, 3, '4' ], 'dict': { '1': '1', '2':
↪ 2, '3': 'three', '4': '4' } } (<class 'dict'>)
```

```
Expectation (Instance data after changing uid): result = { 'str': '__string__', 'unicode':
↪ '__unicode__', 'integer': 34, 'float': 2.71828, 'list': [ 'one', 2, 3, '4' ], 'dict': {
↪ '1': '1', '2': 2, '3': 'three', '4': '4' } } (<class 'dict'>)
```

A.1.8 caching.property_cache_json: Test get from source caused by changed uid (partially init)

Testresult
This test was passed with the state: **Success**.

Info	Initialising property_cache_json (load_all_on_init=True).

```
Deleting cache file from filesystem to ensure identical conditions for each test run.
```

```
Initialising cached class with /home/dirk/my_repositories/unittest/caching/unittest/output_da
↪ ta/uid_test_no_load_on_init.json as cache
↪ file.
```

Success	Instance data after changing uid is correct (Content {'str': '__string__', 'unicode': '__unicode__', 'integer': 34, 'float': 2.71828, 'list': ['one', 2, 3, '4'], 'dict': {'1': '1', '2': 2, '3': 'three', '4': '4'}} and Type is <class 'dict'>).

```
Result (Instance data after changing uid): { 'str': '__string__', 'unicode': '__unicode__',
↪ 'integer': 34, 'float': 2.71828, 'list': [ 'one', 2, 3, '4' ], 'dict': { '1': '1', '2':
↪ 2, '3': 'three', '4': '4' } } (<class 'dict'>)
```

```
Expectation (Instance data after changing uid): result = { 'str': '__string__', 'unicode':
↪ '__unicode__', 'integer': 34, 'float': 2.71828, 'list': [ 'one', 2, 3, '4' ], 'dict': {
↪ '1': '1', '2': 2, '3': 'three', '4': '4' } } (<class 'dict'>)
```

A.1.9 caching.property_cache_json: Test execution of save callback (full init)

Testresult
This test was passed with the state: **Success**.

Info	Installing save_callback, which sets a variable to True on execution.

Success	Save callback execution variable is correct (Content True and Type is <class 'bool'>).

```
Result (Save callback execution variable): True (<class 'bool'>)
```

```
Expectation (Save callback execution variable): result = True (<class 'bool'>)
```

A.1.10 caching.property_cache_json: Test internal key usage

Testresult
This test was passed with the state: **Success**.

Info	Initialising property_cache_json (load_all_on_init=True).

```
Deleting cache file from filesystem to ensure identical conditions for each test run.
```

```
Initialising cached class with
↪ /home/dirk/my_repositories/unittest/caching/unittest/output_data/internal_keys_test.json
↪ as cache file.
```

Info	Extracting storage object from property_cache_json for comparison.

```
Using storage object of cache class for comparison: {'__property_cache_data_version_': 'no
↪  second data version', '__property_cache_uid_': 'no second uid',
↪  '_property_cache_data_version_': 'no data version', '_property_cache_uid_': 'no uid'}
```

Success	Cache is correct (Content {'__property_cache_data_version_': 'no second data version', '__property_cache_uid_': 'no second uid', '_property_cache_data_version_': 'no data version', '_property_cache_uid_': 'no uid'} and Type is <class 'dict'>).

```
Result (Cache): { '__property_cache_data_version_': 'no second data version',
↪  '__property_cache_uid_': 'no second uid', '_property_cache_data_version_': 'no data
↪  version', '_property_cache_uid_': 'no uid' } (<class 'dict'>)
Expectation (Cache): result = { '_property_cache_uid_': 'no uid', '__property_cache_uid_':
↪  'no second uid', '_property_cache_data_version_': 'no data version',
↪  '__property_cache_data_version_': 'no second data version' } (<class 'dict'>)
```

Success	Keyfilter returnvalue for 5 (<class 'int'>) is correct (Content 5 and Type is <class 'int'>).

```
Result (Keyfilter returnvalue for 5 (<class 'int'>)): 5 (<class 'int'>)
Expectation (Keyfilter returnvalue for 5 (<class 'int'>)): result = 5 (<class 'int'>)
```

A.1.11 caching.property_cache_pickle: Test full initialised PICKLE-Cache-Object

Testresult

This test was passed with the state: **Success**.

Info	Initialising property_cache_pickle (load_all_on_init=True).

```
Deleting cache file from filesystem to ensure identical conditions for each test run.
Initialising cached class with
↪  /home/dirk/my_repositories/unittest/caching/unittest/output_data/load_on_init.pkl as
↪  cache file.
```

Info	Extracting storage object from property_cache_pickle for comparison.

```
Using storage object of cache class for comparison: {'str': 'string', 'unicode': 'unicode',
↪  'integer': 17, 'float': 3.14159, 'list': [1, 'two', '3', 4], 'dict': {'1': 1, '2': 'two',
↪  '3': '3', '4': 4}}
```

Success	Cache object is correct (Content {'str': 'string', 'unicode': 'unicode', 'integer': 17, 'float': 3.14159, 'list': [1, 'two', '3', 4], 'dict': {'1': 1, '2': 'two', '3': '3', '4': 4}} and Type is <class 'dict'>).

```
Result (Cache object): { 'str': 'string', 'unicode': 'unicode', 'integer': 17, 'float':
↪  3.14159, 'list': [ 1, 'two', '3', 4 ], 'dict': { '1': 1, '2': 'two', '3': '3', '4': 4 } }
↪  (<class 'dict'>)
```

```
Expectation (Cache object): result = { 'str': 'string', 'unicode': 'unicode', 'integer': 17,
↪  'float': 3.14159, 'list': [ 1, 'two', '3', 4 ], 'dict': { '1': 1, '2': 'two', '3': '3',
↪  '4': 4 } } (<class 'dict'>)
```

A.1.12 caching.property_cache_pickle: Test partially initialised PICKLE-Cache-Object

Testresult

This test was passed with the state: **Success**.

Info	Initialising property_cache_pickle (load_all_on_init=False).

```
Deleting cache file from filesystem to ensure identical conditions for each test run.
```

```
Initialising cached class with
↪  /home/dirk/my_repositories/unittest/caching/unittest/output_data/no_load_on_init.pkl as
↪  cache file.
```

Info	Partially initialising cache object by requesting some information.

Info	Extracting storage object from property_cache_pickle for comparison.

```
Using storage object of cache class for comparison: {'integer': 17, 'str': 'string',
↪  'unicode': 'unicode'}
```

Success	Cache object is correct (Content {'integer': 17, 'str': 'string', 'unicode': 'unicode'} and Type is <class 'dict'>).

```
Result (Cache object): { 'integer': 17, 'str': 'string', 'unicode': 'unicode' } (<class
↪  'dict'>)
```

```
Expectation (Cache object): result = { 'str': 'string', 'unicode': 'unicode', 'integer': 17 }
↪  (<class 'dict'>)
```

A.1.13 caching.property_cache_pickle: Test cached data (full init)

Testresult

This test was passed with the state: **Success**.

Info	Initialising property_cache_pickle (load_all_on_init=True).

```
Deleting cache file from filesystem to ensure identical conditions for each test run.
```

```
Initialising cached class with /home/dirk/my_repositories/unittest/caching/unittest/output_da
↪  ta/cache_data_test_load_on_init.pkl as cache
↪  file.
```

Info	Collecting data from cache instance.

Success	Cached data is correct (Content {'str': 'string', 'unicode': 'unicode', 'integer': 17, 'float': 3.14159, 'list': [1, 'two', '3', 4], 'dict': {'1': 1, '2': 'two', '3': '3', '4': 4}, 'uncached': 'uncached_data_of_class'} and Type is <class 'dict'>).

```
Result (Cached data): { 'str': 'string', 'unicode': 'unicode', 'integer': 17, 'float':
↪  3.14159, 'list': [ 1, 'two', '3', 4 ], 'dict': { '1': 1, '2': 'two', '3': '3', '4': 4 },
↪  'uncached': 'uncached_data_of_class' } (<class 'dict'>)
Expectation (Cached data): result = { 'str': 'string', 'unicode': 'unicode', 'integer': 17,
↪  'float': 3.14159, 'list': [ 1, 'two', '3', 4 ], 'dict': { '1': 1, '2': 'two', '3': '3',
↪  '4': 4 }, 'uncached': 'uncached_data_of_class' } (<class 'dict'>)
```

A.1.14 caching.property_cache_pickle: Test cached data (partially init)

Testresult

This test was passed with the state: **Success**.

Info	Initialising property_cache_pickle (load_all_on_init=True).

```
Deleting cache file from filesystem to ensure identical conditions for each test run.
Initialising cached class with /home/dirk/my_repositories/unittest/caching/unittest/output_da
↪  ta/cache_data_test_no_load_on_init.pkl as cache
↪  file.
```

Info	Collecting data from cache instance.

Success	Cached data is correct (Content {'str': 'string', 'unicode': 'unicode', 'integer': 17, 'float': 3.14159, 'list': [1, 'two', '3', 4], 'dict': {'1': 1, '2': 'two', '3': '3', '4': 4}, 'uncached': 'uncached_data_of_class'} and Type is <class 'dict'>).

```
Result (Cached data): { 'str': 'string', 'unicode': 'unicode', 'integer': 17, 'float':
↪  3.14159, 'list': [ 1, 'two', '3', 4 ], 'dict': { '1': 1, '2': 'two', '3': '3', '4': 4 },
↪  'uncached': 'uncached_data_of_class' } (<class 'dict'>)
Expectation (Cached data): result = { 'str': 'string', 'unicode': 'unicode', 'integer': 17,
↪  'float': 3.14159, 'list': [ 1, 'two', '3', 4 ], 'dict': { '1': 1, '2': 'two', '3': '3',
↪  '4': 4 }, 'uncached': 'uncached_data_of_class' } (<class 'dict'>)
```

A.1.15 caching.property_cache_pickle: Test get from source caused by increased data version (full init)

Testresult

This test was passed with the state: **Success**.

Info	Initialising property_cache_pickle (load_all_on_init=True).

```
Deleting cache file from filesystem to ensure identical conditions for each test run.
```

```
Initialising cached class with /home/dirk/my_repositories/unittest/caching/unittest/output_da
↪ ta/data_version_test_load_on_init.pkl as cache
↪ file.
```

Success	Instance data after increasing data_version is correct (Content {'str': '__string__', 'unicode': '__unicode__', 'integer': 34, 'float': 2.71828, 'list': ['one', 2, 3, '4'], 'dict': {'1': '1', '2': 2, '3': 'three', '4': '4'}} and Type is <class 'dict'>).

```
Result (Instance data after increasing data_version): { 'str': '__string__', 'unicode':
↪ '__unicode__', 'integer': 34, 'float': 2.71828, 'list': [ 'one', 2, 3, '4' ], 'dict': {
↪ '1': '1', '2': 2, '3': 'three', '4': '4' } } (<class 'dict'>)
```

```
Expectation (Instance data after increasing data_version): result = { 'str': '__string__',
↪ 'unicode': '__unicode__', 'integer': 34, 'float': 2.71828, 'list': [ 'one', 2, 3, '4' ],
↪ 'dict': { '1': '1', '2': 2, '3': 'three', '4': '4' } } (<class 'dict'>)
```

A.1.16 caching.property_cache_pickle: Test get from source caused by increased data version (partially init)

Testresult

This test was passed with the state: **Success**.

Info	Initialising property_cache_pickle (load_all_on_init=True).

```
Deleting cache file from filesystem to ensure identical conditions for each test run.
```

```
Initialising cached class with /home/dirk/my_repositories/unittest/caching/unittest/output_da
↪ ta/data_version_test_no_load_on_init.pkl as cache
↪ file.
```

Success	Instance data after increasing data_version is correct (Content {'str': '__string__', 'unicode': '__unicode__', 'integer': 34, 'float': 2.71828, 'list': ['one', 2, 3, '4'], 'dict': {'1': '1', '2': 2, '3': 'three', '4': '4'}} and Type is <class 'dict'>).

```
Result (Instance data after increasing data_version): { 'str': '__string__', 'unicode':
↪ '__unicode__', 'integer': 34, 'float': 2.71828, 'list': [ 'one', 2, 3, '4' ], 'dict': {
↪ '1': '1', '2': 2, '3': 'three', '4': '4' } } (<class 'dict'>)
```

```
Expectation (Instance data after increasing data_version): result = { 'str': '__string__',
↪ 'unicode': '__unicode__', 'integer': 34, 'float': 2.71828, 'list': [ 'one', 2, 3, '4' ],
↪ 'dict': { '1': '1', '2': 2, '3': 'three', '4': '4' } } (<class 'dict'>)
```

A.1.17 caching.property_cache_pickle: Test get from source caused by changed uid (full init)

Testresult

This test was passed with the state: **Success**.

Info	Initialising property_cache_pickle (load_all_on_init=True).

```
Deleting cache file from filesystem to ensure identical conditions for each test run.
```

```
Initialising cached class with /home/dirk/my_repositories/unittest/caching/unittest/output_da
↪ ta/uid_test_load_on_init.pkl as cache
↪ file.
```

Success	Instance data after changing uid is correct (Content {'str': '__string__', 'unicode': '__unicode__', 'integer': 34, 'float': 2.71828, 'list': ['one', 2, 3, '4'], 'dict': {'1': '1', '2': 2, '3': 'three', '4': '4'}} and Type is <class 'dict'>).

```
Result (Instance data after changing uid): { 'str': '__string__', 'unicode': '__unicode__',
↪ 'integer': 34, 'float': 2.71828, 'list': [ 'one', 2, 3, '4' ], 'dict': { '1': '1', '2':
↪ 2, '3': 'three', '4': '4' } } (<class 'dict'>)
Expectation (Instance data after changing uid): result = { 'str': '__string__', 'unicode':
↪ '__unicode__', 'integer': 34, 'float': 2.71828, 'list': [ 'one', 2, 3, '4' ], 'dict': {
↪ '1': '1', '2': 2, '3': 'three', '4': '4' } } (<class 'dict'>)
```

A.1.18 caching.property_cache_pickle: Test get from source caused by changed uid (partially init)

Testresult

This test was passed with the state: **Success**.

Info	Initialising property_cache_pickle (load_all_on_init=True).

```
Deleting cache file from filesystem to ensure identical conditions for each test run.
Initialising cached class with /home/dirk/my_repositories/unittest/caching/unittest/output_da
↪ ta/uid_test_no_load_on_init.pkl as cache
↪ file.
```

Success	Instance data after changing uid is correct (Content {'str': '__string__', 'unicode': '__unicode__', 'integer': 34, 'float': 2.71828, 'list': ['one', 2, 3, '4'], 'dict': {'1': '1', '2': 2, '3': 'three', '4': '4'}} and Type is <class 'dict'>).

```
Result (Instance data after changing uid): { 'str': '__string__', 'unicode': '__unicode__',
↪ 'integer': 34, 'float': 2.71828, 'list': [ 'one', 2, 3, '4' ], 'dict': { '1': '1', '2':
↪ 2, '3': 'three', '4': '4' } } (<class 'dict'>)
Expectation (Instance data after changing uid): result = { 'str': '__string__', 'unicode':
↪ '__unicode__', 'integer': 34, 'float': 2.71828, 'list': [ 'one', 2, 3, '4' ], 'dict': {
↪ '1': '1', '2': 2, '3': 'three', '4': '4' } } (<class 'dict'>)
```

A.1.19 caching.property_cache_pickle: Test execution of save callback (full init)

Testresult

This test was passed with the state: **Success**.

Info	Installing save_callback, which sets a variable to True on execution.

Success	Save callback execution variable is correct (Content True and Type is <class 'bool'>).

```
Result (Save callback execution variable): True (<class 'bool'>)
Expectation (Save callback execution variable): result = True (<class 'bool'>)
```

A.1.20 caching.property_cache_pickle: Test internal key usage

Testresult
This test was passed with the state: **Success**.

Info	Initialising property_cache_pickle (load_all_on_init=True).

```
Deleting cache file from filesystem to ensure identical conditions for each test run.
```

```
Initialising cached class with
↪ /home/dirk/my_repositories/unittest/caching/unittest/output_data/internal_keys_test.pkl
↪ as cache file.
```

Info	Extracting storage object from property_cache_pickle for comparison.

```
Using storage object of cache class for comparison: {'_property_cache_uid_': 'no uid',
↪ '__property_cache_uid_': 'no second uid', '_property_cache_data_version_': 'no data
↪ version', '__property_cache_data_version_': 'no second data version'}
```

Success	Cache is correct (Content {'_property_cache_uid_': 'no uid', '__property_cache_uid_': 'no second uid', '_property_cache_data_version_': 'no data version', '__property_cache_data_version_': 'no second data version'} and Type is <class 'dict'>).

```
Result (Cache): { '_property_cache_uid_': 'no uid', '__property_cache_uid_': 'no second uid',
↪ '_property_cache_data_version_': 'no data version', '__property_cache_data_version_': 'no
↪ second data version' } (<class 'dict'>)
```

```
Expectation (Cache): result = { '_property_cache_uid_': 'no uid', '__property_cache_uid_':
↪ 'no second uid', '_property_cache_data_version_': 'no data version',
↪ '__property_cache_data_version_': 'no second data version' } (<class 'dict'>)
```

Success	Keyfilter returnvalue for 5 (<class 'int'>) is correct (Content 5 and Type is <class 'int'>).

```
Result (Keyfilter returnvalue for 5 (<class 'int'>)): 5 (<class 'int'>)
```

```
Expectation (Keyfilter returnvalue for 5 (<class 'int'>)): result = 5 (<class 'int'>)
```

B Test-Coverage

B.1 caching

The line coverage for caching was 89.4%
The branch coverage for caching was 79.3%

B.1.1 caching.__init__.py

The line coverage for caching.__init__.py was 89.4%
The branch coverage for caching.__init__.py was 79.3%

```
#!/usr/bin/env python
# -*- coding: utf-8 -*-
#
"""
caching (Caching Module)
========================

**Author:**

* Dirk Alders <sudo-dirk@mount-mockery.de>

**Description:**

    This Module supports functions and classes for caching e.g. properties of other instances.

**Submodules:**

* :class:`caching.property_cache_json`
* :class:`caching.property_cache_pickle`

**Unittest:**

    See also the :download:`unittest <caching/_testresults_/unittest.pdf>` documentation.
"""
__DEPENDENCIES__ = []

import json
import logging
import os
import pickle
import time

try:
    from config import APP_NAME as ROOT_LOGGER_NAME
except ImportError:
    ROOT_LOGGER_NAME = 'root'
logger = logging.getLogger(ROOT_LOGGER_NAME).getChild(__name__)

__DESCRIPTION__ = """The Module {\\tt %s} is designed to store information in {\\tt json} or {\\tt pickle} files to support them much faster then generating them from the original source file.
For more Information read the documentation.""" % __name__.replace('_', '\_')
"""The Module Description"""
__INTERPRETER__ = (3, )
"""The Tested Interpreter-Versions"""


class property_cache_pickle(object):
    """
    This class caches the data from a given `source_instance`. It takes the data from the cache instead of generating the data from the `source_instance`,
    if the conditions for the cache usage are given.

    .. admonition:: Required properties for the `source_instance`

                * **uid():** returns the unique id of the source's source or None, if you don't want to use the unique id.
                * **keys():** returns a list of all available keys.
                * **data_version():** returns a version number of the current data (it should be increased, if the get method of the source instance returns improved values or the data structure had been changed).
```

```
56                * **get(key, default):** returns the property for a key. If key does not exists, default will be returned.
57
58    :param source_instance: The source instance holding the data
59    :type source_instance: instance
60    :param cache_filename: File name, where the properties are stored as cache
61    :type cache_filename: str
62    :param load_all_on_init: True will load all data from the source instance, when the cache will be initialised the first time.
63    :type load_all_on_init: bool
64    :param callback_on_data_storage: The callback will be executed every time when the cache file is stored. It will be executed with the instance of this class as first argument.
65    :type callback_on_data_storage: method
66    :param max_age: The maximum age of the cached data in seconds or None for no maximum age.
67    :type max_age: int or None
68    :param store_on_get: False will prevent cache storage with execution of the `.get(key, default)` method. You need to store the cache somewhere else.
69    :type store_on_get: bool
70
71    .. admonition:: The cache will be used, if all following conditions are given
72
73                * The key is in the list returned by `.keys()` method of the `source_instance`
74                * The key is not in the list of keys added by the `.add_source_get_keys()` method.
75                * The cache age is less then the given max_age parameter or the given max_age is None.
76                * The uid of the source instance (e.g. a checksum or unique id of the source) is identically to to uid stored in the cache.
77                * The data version of the `source_instance` is <= the data version stored in the cache.
78                * The value is available in the previous stored information
79
80    **Example:**
81
82    .. literalinclude:: caching/_examples_/property_cache_pickle.py
83
84    Will result on the first execution to the following output (with a long execution time):
85
86    .. literalinclude:: caching/_examples_/property_cache_pickle_1.log
87
88    With every following execution the time cosumption my by much smaller:
89
90    .. literalinclude:: caching/_examples_/property_cache_pickle_2.log
91    """
92    DATA_VERSION_TAG = '_property_cache_data_version_'
93    STORAGE_VERSION_TAG = '_storage_version_'
94    UID_TAG = '_property_cache_uid_'
95    DATA_TAG = '_data_'
96    AGE_TAG = '_age_'
97    #
98    STORAGE_VERSION = 1
99
100   def __init__(self, source_instance, cache_filename, load_all_on_init=False, callback_on_data_storage=None, max_age=None, store_on_get=True):
101       self._source_instance = source_instance
102       self._cache_filename = cache_filename
103       self._load_all_on_init = load_all_on_init
104       self._callback_on_data_storage = callback_on_data_storage
105       self._max_age = max_age
106       self._store_on_get = store_on_get
107       #
```

```
108         self._source_get_keys = []
109         self._cached_props = None
110 
111     def add_source_get_keys(self, keys):
112         """
113         This will add one or more keys to a list of keys which will always be provided by the `source_instance` instead of the cache.
114 
115         :param keys: The key or keys to be added
116         :type keys: list, tuple, str
117         """
118         if type(keys) in [list, tuple]:
119             self._source_get_keys.extend(keys)
120         else:
121             self._source_get_keys.append(keys)
122 
123     def full_update(self, sleep_between_keys=0):
124         """
125         With the execution of this method, the complete source data which needs to be cached, will be read from the source instance
126         and the resulting cache will be stored to the given file.
127 
128         :param sleep_between_keys: Time to sleep between each source data generation
129         :type sleep_between_keys: float, int
130 
131         .. hint:: Use this method, if you initiallised the class with `store_on_get=False`
132         """
133         self._load_source(sleep_between_keys=sleep_between_keys)
134         self._save_cache()
135 
136     def get(self, key, default=None):
137         """
138         Method to get the cached property. If the key does not exists in the cache or `source_instance`, `default` will be returned.
139 
140         :param key: key for value to get.
141         :param default: value to be returned, if key does not exists.
142         :returns: value for a given key or default value.
143         """
144         if key in self._source_instance.keys() and key not in self._source_get_keys:
145             if self._cached_props is None:
146                 self._init_cache()
147             if self._max_age is None:
148                 cache_old = False
149             else:
150                 cache_old = time.time() - self._cached_props[self.AGE_TAG].get(self._key_filter(key), 0) > self._max_age
151                 if cache_old:
152                     logger.debug("The cached value is old, cached value will be ignored")
153             if self._key_filter(key) not in self._cached_props[self.DATA_TAG] or cache_old:
154                 logger.debug("Loading property for key='%s' from source instance", key)
155                 val = self._source_instance.get(key, None)
156                 if self._store_on_get:
157                     tm = int(time.time())
158                     logger.debug("Adding key=%s, value=%s with timestamp=%d to chache", key, val, tm)
159                     self._cached_props[self.DATA_TAG][self._key_filter(key)] = val
160                     self._cached_props[self.AGE_TAG][self._key_filter(key)] = tm
161                     self._save_cache()
162                 else:
163                     return val
164             else:
```

```
165                 logger.debug("Providing property for '%s' from cache", key)
166             return self._cached_props[self.DATA_TAG].get(self._key_filter(key), default)
167         else:
168             if key not in self._source_instance.keys():
169                 logger.debug("Key '%s' is not in cached_keys. Uncached data will be returned.",
    key)
170             elif key in self._source_get_keys:
171                 logger.debug("Key '%s' is excluded by .add_source_get_keys(). Uncached data will
    be returned.", key)
172             return self._source_instance.get(key, default)
173
174     def _data_version(self):
175         if self._cached_props is None:
176             return None
177         else:
178             return self._cached_props.get(self.DATA_VERSION_TAG, None)
179
180     def _storage_version(self):
181         if self._cached_props is None:
182             return None
183         else:
184             return self._cached_props.get(self.STORAGE_VERSION_TAG, None)
185
186     def _init_cache(self):
187         load_cache = self._load_cache()
188         uid = self._source_instance.uid() != self._uid()
189         try:
190             data_version = self._source_instance.data_version() > self._data_version()
191         except TypeError:
192             data_version = True
193         try:
194             storage_version = self._storage_version() != self.STORAGE_VERSION
195         except TypeError:
196             storage_version = True
197         #
198         if not load_cache or uid or data_version or storage_version:
199             if load_cache:
200                 if self._uid() is not None and uid:
201                     logger.debug("Source uid changed, ignoring previous cache data")
202                 if self._data_version() is not None and data_version:
203                     logger.debug("Data version increased, ignoring previous cache data")
204                 if storage_version:
205                     logger.debug("Storage version changed, ignoring previous cache data")
206             self._cached_props = {self.AGE_TAG: {}, self.DATA_TAG: {}}
207             if self._load_all_on_init:
208                 self._load_source()
209             self._cached_props[self.UID_TAG] = self._source_instance.uid()
210             self._cached_props[self.DATA_VERSION_TAG] = self._source_instance.data_version()
211             self._cached_props[self.STORAGE_VERSION_TAG] = self.STORAGE_VERSION
212             self._save_cache()
213
214     def _load_cache(self):
215         if os.path.exists(self._cache_filename):
216             with open(self._cache_filename, 'rb') as fh:
217                 self._cached_props = pickle.load(fh)
218             logger.debug('Loading properties from cache (%s)', self._cache_filename)
219             return True
220         else:
221             logger.debug('Cache file does not exists (yet).')
222         return False
223
224     def _key_filter(self, key):
225         return key
```

```
226
227     def _load_source(self, sleep_between_keys=0):
228         if self._cached_props is None:
229             self._init_cache()
230         logger.debug('Loading all data from source - %s', repr(self._source_instance.keys()))
231         for key in self._source_instance.keys():
232             if key not in self._source_get_keys:
233                 self._cached_props[self.DATA_TAG][self._key_filter(key)] = self._source_instance.
    get(key)
234                 self._cached_props[self.AGE_TAG][self._key_filter(key)] = int(time.time())
235                 time.sleep(sleep_between_keys)
236
237     def _save_cache(self):
238         with open(self._cache_filename, 'wb') as fh:
239             pickle.dump(self._cached_props, fh)
240             logger.debug('cache-file stored (%s)', self._cache_filename)
241         if self._callback_on_data_storage is not None:
242             self._callback_on_data_storage(self)
243
244     def _uid(self):
245         if self._cached_props is None:
246             return None
247         else:
248             return self._cached_props.get(self.UID_TAG, None)
249
250
251 class property_cache_json(property_cache_pickle):
252     """
253     See also parent :py:class:`property_cache_pickle` for detailed information.
254
255     .. important::
256         * This class uses json. You should **only** use keys of type string!
257         * Unicode types are transfered to strings
258
259         See limitations of json.
260
261     **Example:**
262
263     .. literalinclude:: caching/_examples_/property_cache_json.py
264
265     Will result on the first execution to the following output (with a long execution time):
266
267     .. literalinclude:: caching/_examples_/property_cache_json_1.log
268
269     With every following execution the time cosumption my by much smaller:
270
271     .. literalinclude:: caching/_examples_/property_cache_json_2.log
272     """
273     def _load_cache(self):
274         if os.path.exists(self._cache_filename):
275             with open(self._cache_filename, 'r') as fh:
276                 self._cached_props = json.load(fh)
277             logger.debug('Loading properties from cache (%s)', self._cache_filename)
278             return True
279         else:
280             logger.debug('Cache file does not exists (yet).')
281         return False
282
283     def _save_cache(self):
284         with open(self._cache_filename, 'w') as fh:
285             json.dump(self._cached_props, fh, sort_keys=True, indent=4)
286             logger.debug('cache-file stored (%s)', self._cache_filename)
287         if self._callback_on_data_storage is not None:
288             self._callback_on_data_storage(self)
```