Unittest for `caching`

September 15, 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	624db9f532db2ba202265378292f8af1
Dependencies	

1.2 Unittest Information

Unittest Information	
Version	6120114abb148e96dd3b8bd41a7af82a
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	0.024s

2.2 Coverage Statistic

Module- or Filename	Line-Coverage	Branch-Coverage
caching	95.5%	91.3%
caching.__init__.py	95.5%	

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:	2024-09-15 14:31:27,861
Finished-Time:	2024-09-15 14:31:27,861
Time-Consumption	0.001s

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:	2024-09-15 14:31:27,862
Finished-Time:	2024-09-15 14:31:27,862
Time-Consumption	0.001s

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:	2024-09-15 14:31:27,869
Finished-Time:	2024-09-15 14:31:27,869
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'>).

3.1.4 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:	2024-09-15 14:31:27,858
Finished-Time:	2024-09-15 14:31:27,859
Time-Consumption	0.001s

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:	2024-09-15 14:31:27,865
Finished-Time:	2024-09-15 14:31:27,866
Time-Consumption	0.001s

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:	2024-09-15 14:31:27,866
Finished-Time:	2024-09-15 14:31:27,869
Time-Consumption	0.002s

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:	2024-09-15 14:31:27,862
Finished-Time:	2024-09-15 14:31:27,863
Time-Consumption	0.001s

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:	2024-09-15 14:31:27,863
Finished-Time:	2024-09-15 14:31:27,865
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'>).

3.1.9 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:	2024-09-15 14:31:27,870
Finished-Time:	2024-09-15 14:31:27,871
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 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:	2024-09-15 14:31:27,859
Finished-Time:	2024-09-15 14:31:27,861
Time-Consumption	0.002s

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:	2024-09-15 14:31:27,874
Finished-Time:	2024-09-15 14:31:27,875
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:	2024-09-15 14:31:27,875
Finished-Time:	2024-09-15 14:31:27,876
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:	2024-09-15 14:31:27,883
Finished-Time:	2024-09-15 14:31:27,883
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:	2024-09-15 14:31:27,871
Finished-Time:	2024-09-15 14:31:27,872
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:	2024-09-15 14:31:27,880
Finished-Time:	2024-09-15 14:31:27,881
Time-Consumption	0.001s

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:	2024-09-15 14:31:27,881
Finished-Time:	2024-09-15 14:31:27,882
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:	2024-09-15 14:31:27,877
Finished-Time:	2024-09-15 14:31:27,878
Time-Consumption	0.001s

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:	2024-09-15 14:31:27,878
Finished-Time:	2024-09-15 14:31:27,879
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:	2024-09-15 14:31:27,883
Finished-Time:	2024-09-15 14:31:27,884
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 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:	2024-09-15 14:31:27,873
Finished-Time:	2024-09-15 14:31:27,874
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 95.5%
The branch coverage for caching was 91.3%

B.1.1 caching.__init__.py

The line coverage for caching.__init__.py was 95.5%
The branch coverage for caching.__init__.py was 91.3%

```
#!/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 sys
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):
    """
    Class to cache properties, which take longer on initialising than reading a file in pickle format.

    :param source_instance: The source instance holding the data
    :type source_instance: instance
    :param cache_filename: File name, where the properties are stored as cache
    :type cache_filename: str
    :param load_all_on_init: Optionally init behaviour control parameter. True will load all available properties from source on init, False not.

    .. note:: source_instance needs to have at least the following methods: uid(), keys(), data_version(), get()
```

```
58
59              * uid(): returns the unique id of the source.
60              * keys(): returns a list of all available keys.
61              * 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).
62              * get(key, default): returns the property for a key. If key does not exists,
    default will be returned.
63
64      Reasons for updating the complete data set:
65
66      * UID of source_instance has changed (in comparison to the cached value).
67      * data_version is increased
68
69      **Example:**
70
71      .. literalinclude:: caching/_examples_/property_cache_pickle.py
72
73      Will result on the first execution to the following output (with a long execution time):
74
75      .. literalinclude:: caching/_examples_/property_cache_pickle_1.log
76
77      With every following execution (slow for getting "two" which is not cached – see
    implementation):
78
79      .. literalinclude:: caching/_examples_/property_cache_pickle_2.log
80      """
81      LOG_PREFIX = 'PickCache:'
82      DATA_VERSION_TAG = '_property_cache_data_version_'
83      STORAGE_VERSION_TAG = '_storage_version_'
84      UID_TAG = '_property_cache_uid_'
85      DATA_TAG = '_data_'
86      AGE_TAG = '_age_'
87      #
88      STORAGE_VERSION = 1
89
90      def __init__(self, source_instance, cache_filename, load_all_on_init=False,
    callback_on_data_storage=None, max_age=None, store_none_value=False):
91          self._source_instance = source_instance
92          self._cache_filename = cache_filename
93          self._load_all_on_init = load_all_on_init
94          self._callback_on_data_storage = callback_on_data_storage
95          self._max_age = max_age
96          self._store_none_value = store_none_value
97          self._cached_props = None
98
99      def get(self, key, default=None):
100         """
101         Method to get the cached property. If key does not exists in cache, the property will be
    loaded from source_instance and stored in cache (file).
102
103         :param key: key for value to get.
104         :param default: value to be returned, if key does not exists.
105         :returns: value for a given key or default value.
106         """
107         if key in self.keys():
108             if self._cached_props is None:
109                 self._init_cache()
110             if self._max_age is None:
111                 cache_old = False
112             else:
```

```
113                 cache_old = time.time() - self._cached_props[self.AGE_TAG].get(self._key_filter(
    key), 0) > self._max_age
114                 if cache_old:
115                     logger.debug("The cached value is old, cached value will be ignored")
116             if self._key_filter(key) not in self._cached_props[self.DATA_TAG] or cache_old:
117                 val = self._source_instance.get(key, None)
118                 logger.debug("%s Loading property for '%s' from source instance (%s)", self.
    LOG_PREFIX, key, repr(val))
119                 if val or self._store_none_value:
120                     tm = int(time.time())
121                     logger.debug("Storing value=%s with timestamp=%d to chache", val, tm)
122                     self._cached_props[self.DATA_TAG][self._key_filter(key)] = val
123                     self._cached_props[self.AGE_TAG][self._key_filter(key)] = tm
124                     self._save_cache()
125                 else:
126                     return val
127             else:
128                 logger.debug("%s Providing property for '%s' from cache (%s)", self.LOG_PREFIX,
129                              key, repr(self._cached_props[self.DATA_TAG].get(self._key_filter(key
    ), default)))
130             return self._cached_props[self.DATA_TAG].get(self._key_filter(key), default)
131         else:
132             logger.info("%s Key '%s' is not in cached_keys. Uncached data will be returned.",
    self.LOG_PREFIX, key)
133             return self._source_instance.get(key, default)
134 
135     def keys(self):
136         """
137         Method to get the available keys (from :data:`source_instance`).
138         """
139         return self._source_instance.keys()
140 
141     def _data_version(self):
142         if self._cached_props is None:
143             return None
144         else:
145             return self._cached_props.get(self.DATA_VERSION_TAG, None)
146 
147     def _storage_version(self):
148         if self._cached_props is None:
149             return None
150         else:
151             return self._cached_props.get(self.STORAGE_VERSION_TAG, None)
152 
153     def _init_cache(self):
154         load_cache = self._load_cache()
155         uid = self._source_instance.uid() != self._uid()
156         try:
157             data_version = self._source_instance.data_version() > self._data_version()
158         except TypeError:
159             data_version = True
160         try:
161             storage_version = self._storage_version() != self.STORAGE_VERSION
162         except TypeError:
163             storage_version = True
164         #
```

```
165         if not load_cache or uid or data_version or storage_version:
166             if self._uid() is not None and uid:
167                 logger.debug("%s Source uid changed, ignoring previous cache data", self.
    LOG_PREFIX)
168             if self._data_version() is not None and data_version:
169                 logger.debug("%s Data version increased, ignoring previous cache data", self.
    LOG_PREFIX)
170             if storage_version:
171                 logger.debug("%s Storage version changed, ignoring previous cache data", self.
    LOG_PREFIX)
172             self._cached_props = {self.AGE_TAG: {}, self.DATA_TAG: {}}
173             if self._load_all_on_init:
174                 self._load_source()
175             self._cached_props[self.UID_TAG] = self._source_instance.uid()
176             self._cached_props[self.DATA_VERSION_TAG] = self._source_instance.data_version()
177             self._cached_props[self.STORAGE_VERSION_TAG] = self.STORAGE_VERSION
178             self._save_cache()
179 
180     def _load_cache(self):
181         if os.path.exists(self._cache_filename):
182             with open(self._cache_filename, 'rb') as fh:
183                 self._cached_props = pickle.load(fh)
184             logger.info('%s Loading properties from cache (%s)', self.LOG_PREFIX, self.
    _cache_filename)
185             return True
186         else:
187             logger.debug('%s Cache file does not exists (yet).', self.LOG_PREFIX)
188         return False
189 
190     def _key_filter(self, key):
191         return key
192 
193     def _load_source(self):
194         logger.debug('%s Loading all data from source - %s', self.LOG_PREFIX, repr(self.
    _source_instance.keys()))
195         for key in self._source_instance.keys():
196             val = self._source_instance.get(key)
197             self._cached_props[self.DATA_TAG][self._key_filter(key)] = val
198             self._cached_props[self.AGE_TAG][self._key_filter(key)] = int(time.time())
199 
200     def _save_cache(self):
201         with open(self._cache_filename, 'wb') as fh:
202             pickle.dump(self._cached_props, fh)
203             logger.info('%s cache-file stored (%s)', self.LOG_PREFIX, self._cache_filename)
204         if self._callback_on_data_storage is not None:
205             self._callback_on_data_storage()
206 
207     def _uid(self):
208         if self._cached_props is None:
209             return None
210         else:
211             return self._cached_props.get(self.UID_TAG, None)
212 
213 
214 class property_cache_json(property_cache_pickle):
215     """
216     Class to cache properties, which take longer on initialising than reading a file in json
    format. See also parent :py:class:`property_cache_pickle`
217 
218     :param source_instance: The source instance holding the data
219     :type source_instance: instance
220     :param cache_filename: File name, where the properties are stored as cache
```

```
221     :type cache_filename: str
222     :param load_all_on_init: Optionally init behaviour control parameter. True will load all
        available properties from source on init, False not.
223
224     .. warning::
225         * This class uses json. You should **only** use keys of type string!
226         * Unicode types are transfered to strings
227
228     .. note:: source_instance needs to have at least the following methods: uid(), keys(),
        data_version(), get()
229
230                 * uid(): returns the unique id of the source.
231                 * keys(): returns a list of all available keys.
232                 * 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).
233                 * get(key, default): returns the property for a key. If key does not exists,
        default will be returned.
234
235     Reasons for updating the complete data set:
236
237     * UID of source_instance has changed (in comparison to the cached value).
238     * data_version is increased
239
240     **Example:**
241
242     .. literalinclude:: caching/_examples_/property_cache_json.py
243
244     Will result on the first execution to the following output (with a long execution time):
245
246     .. literalinclude:: caching/_examples_/property_cache_json_1.log
247
248     With every following execution (slow for getting "two" which is not cached – see
        implementation):
249
250     .. literalinclude:: caching/_examples_/property_cache_json_2.log
251     """
252     LOG_PREFIX = 'JsonCache:'
253
254     def _load_cache(self):
255         if os.path.exists(self._cache_filename):
256             with open(self._cache_filename, 'r') as fh:
257                 self._cached_props = json.load(fh)
258             logger.info('%s Loading properties from cache (%s)', self.LOG_PREFIX, self.
        _cache_filename)
259             return True
260         else:
261             logger.debug('%s Cache file does not exists (yet).', self.LOG_PREFIX)
262         return False
263
264     def _save_cache(self):
265         with open(self._cache_filename, 'w') as fh:
266             json.dump(self._cached_props, fh, sort_keys=True, indent=4)
267             logger.info('%s cache-file stored (%s)', self.LOG_PREFIX, self._cache_filename)
268         if self._callback_on_data_storage is not None:
269             self._callback_on_data_storage()
```