

---

# **Store factory for simplekv**

**Feb 25, 2020**



---

## Contents

---

<b>1</b>	<b>Overview</b>	<b>3</b>
1.1	Store factory for simplekv . . . . .	3
<b>2</b>	<b>Reference</b>	<b>5</b>
2.1	storefact . . . . .	5
<b>3</b>	<b>Contributing</b>	<b>7</b>
3.1	Bug reports . . . . .	7
3.2	Documentation improvements . . . . .	7
3.3	Feature requests and feedback . . . . .	7
3.4	Development . . . . .	8
<b>4</b>	<b>Authors</b>	<b>9</b>
<b>5</b>	<b>Release Notes</b>	<b>11</b>
<b>6</b>	<b>Indices and tables</b>	<b>13</b>



Contents:



## 1.1 Store factory for simplekv

A factory for [simplekv](#)-Store-based storage classes. Takes configuration values and returns a [simplekv](#)-Store.

This allows one to easily deploy a blob-based store in production, but test with a filesystem-based store in development. The following [simplekv](#)-Stores are supported in storefact:

- DictStore
- RedisStore
- FilesystemStore
- BotoStore (Amazon S3)
- AzureBlockBlobStorage

Storefact is released as open source under the 3-clause BSD license.

### 1.1.1 Installation

```
pip install storefact
```

### 1.1.2 Usage

There are two possibilities to use storefact.

- 1) Use a dictionary with configuration data (e.g. loaded from an ini file)

```
from storefact import get_store

params = {
    'account_name': 'test',
    'account_key': 'XXXsome_azure_account_keyXXX',
    'container': 'my-azure-container',
}
store = get_store('azure', **params)
store.put(u'key', b'value')
assert store.get(u'key') == b'value'
```

- 2) Use an URL to specify the configuration

```
from storefact import get_store_from_url, get_store

store = get_store_from_url('azure://test:XXXsome_azure_account_keyXXX@my-azure-
↪container')
store.put(u'key', b'value')
assert store.get(u'key') == b'value'
```

### 1.1.3 Documentation

The documentation can be found on [readthedocs](#).

### 1.1.4 Development

To run the all tests run:

```
tox
```



## CHAPTER 2

---

Reference

---

### 2.1 storefact



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

### 3.1 Bug reports

When reporting a bug please include:

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

### 3.2 Documentation improvements

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

### 3.3 Feature requests and feedback

If you are proposing a feature:

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

## 3.4 Development

### 3.4.1 Pull Request Guidelines

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

For merging, you should:

1. Include passing tests (run `tox`).
2. Update documentation when there's new API, functionality etc.
3. Add a note to `CHANGELOG.rst` about the changes.
4. Add yourself to `AUTHORS.rst`.

### 3.4.2 Tips

To run a subset of tests:

```
tox -e envname -- py.test -k test_myfeature
```

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

```
detox
```

## CHAPTER 4

---

### Authors

---

- Felix Marcinowski - [felix.marczinowski@blue-yonder.com](mailto:felix.marczinowski@blue-yonder.com)
- Cornelius Riemenschneider - [cornelius.riemenschneider@blue-yonder.com](mailto:cornelius.riemenschneider@blue-yonder.com)



## CHAPTER 5

---

### Release Notes

---





## CHAPTER 6

---

### Indices and tables

---

- `genindex`
- `modindex`
- `search`