django-b2 Documentation

Release 0.7.0

Mirek Zvolský

Feb 26, 2022

Contents

Contents:

CHAPTER 1

django-b2

django backblaze b2 storage using b2sdk (b2sdk is official backblaze python library)

1.1 Documentation

The full documentation is at https://django-b2.readthedocs.io.

1.2 Quickstart

Install django-b2:

```
pip install django-b2
```

Add into your settings:

```
MEDIA_URL = '/media/'
DEFAULT_FILE_STORAGE = 'django_b2.storage.B2Storage'                  # if used␣
↪without django-tenant-schemas
# DEFAULT_FILE_STORAGE = 'django_b2.tenant_storage.TenantB2Storage'  # if used with␣
↪django-tenant-schemas
B2_APP_KEY_ID = 000xxxxxxxxxxx000000000n
B2_APP_KEY = keyvalue
B2_BUCKET_NAME = bucketname
# optional, see Usage (docs/usage.rst):
B2_FORCE_UNIQUE = False
# optional, see Usage (docs/usage.rst):
```

(continues on next page)

(continued from previous page)

```
MEDIA_ROOT = ..
B2_LOCAL_MEDIA = ..  # "", "M", "L", "ML"
```

Using outside of Django:

```
from django_b2.backblaze_b2 import BackBlazeB2
b2 = BackBlazeB2()
b2.authorize("production", application_key_id, application_key)
b2.set_bucket(bucket_name)
with open(filename, 'rb') as f:
    b2.upload_file(filename, f)
content = b2.download_file(filename)
with open(filename2, 'wb') as f:
    f.write(content)
```

1.3 Features

- Django media storage (with storage.py) or general python access to BackBlaze B2 (without usage of storage.py).
- Upload single file to B2 (call backblaze_b2.py as script; new in 0.2.0)
- Backup a postgres database to B2 (use script pgtob2.sh; new in 0.2.0)
- Optionally cache media files locally for immediate access or for long time faster access.

1.4 Running Tests

Does the code actually work?

```
source <YOURVIRTUALENV>/bin/activate
(myenv) $ pip install -r requirements_test.txt
(myenv) $ tox
```

1.5 Credits

Tools used in rendering this package:

- b2sdk
- cookiecutter
- cookiecutter-djangopackage

CHAPTER 2

Installation

For usage:

```
$ pip install django-b2
```

For development, with virtualenvwrapper installed:

```
$ git clone https://github.com/pyutil/django-b2.git
$ cd django-b2
$ mkvirtualenv django-b2 -p python3 -r requirements.txt
```

CHAPTER 3

Usage

Django:

Add this to Django settings

```
MEDIA_URL = '/media/'
DEFAULT_FILE_STORAGE = 'django_b2.storage.B2Storage'                    # if used␣
↪without django-tenant-schemas
# DEFAULT_FILE_STORAGE = 'django_b2.tenant_storage.TenantB2Storage'  # if used with␣
↪django-tenant-schemas
B2_APP_KEY_ID=000xxxxxxxxxx000000000n
B2_APP_KEY=keyvalue
B2_BUCKET_NAME=bucketname
# see bellow:
B2_FORCE_UNIQUE = False | True  # for v0.7, True is default
# optional, see bellow:
MEDIA_ROOT = ..
B2_LOCAL_MEDIA = ..  # "", "M", "L", "ML"
```

Large uploads:

Nginx large file uploads: You need at least modify /etc/nginx/nginx.conf, http section, add client... settings.

Read: https://vsoch.github.io/2018/django-nginx-upload/

Example: client_max_body_size 100M; client_body_buffer_size 100M; client_body_timeout 120;

Upload library which enforces name uniqueness

django_b2 saves each file into unique folder ie. as <uuid>/name. This should prevent problems with non unique names. If this doesn't work properly in your case especially if other tool makes the same thing you should turn this feature off. Example: django_drf_filepond creates similar uniquely named folders. In such case set

B2_FORCE_UNIQUE = False

We recommend set this always (False|True) because the default could change in the future.

Imagekit which need reopen the picture soon (Wagtail or so), local filesystem cache

If you upload an image and the imagekit want to reopen it immediately (to create thumbnails or so) it can fail because backblaze storage has the file not immediately accessible. We handle this that way that you can add B2_LOCAL_MEDIA into your settings.

B2_LOCAL_MEDIA='ML'

M means that the copy of file is saved to the local filesystem MEDIA_ROOT too and the file is opened from this local "cache". That way the files are soon and faster accessible. L means that a log files are written into <MEDIA_ROOT>/_log. With help of logs (files in _log directory) the local media files can be deleted: one hour later or 5 days later.

There is a script (django management command) b2_clear_local_media usable via cron, celery,.. It will delete local media files older than few hours (parameter: –hours) or days (–days). The script move the deleted logs into _log/history/ (you can prevent this with –no-history). The affected media files will later be served from backblaze.

Or without deleting of local media files you can simple have primary local media files and their backup on backblaze.

Without Django:

You can upload/download single files. This is not suitable for mass backups, see backblaze docs.

```
from django_b2.backblaze_b2 import BackBlazeB2
b2 = BackBlazeB2()
b2.authorize("production", application_key_id, application_key)
b2.set_bucket(bucket_name)
with open(filename, 'rb') as f:
    b2.upload_file(filename, f)
content = b2.download_file(filename)
with open(filename2, 'wb') as f:
    f.write(content)
```

Upload/backup single file to b2 from console:

This is implemented in backblaze_b2.py file if called as script. We don't implement it (yet) as the django management command. That means outside of Django: You can use this too. And in Django: No need (at 0.2.0) to add this package to INSTALLED_APPS.

You can describe the target bucket in environment variables or in the .ini file. For details:

```
python (path/)backblaze_b2.py --help
```

Backup the postgres database:

Once django-b2 is installed, pgtob2.sh script is available in the virtual environment. Write 'which pgtob2.sh' for its location. Read comments inside the script for more info.

More (this is not well tested):

django-tenants instead of django-tenant-schemas? Maybe it could work but the mixin is inside the django-tenant-schemas package. Add django-tenant-schemas into requirements too but don't add it into INSTALLED_APPS/SHARED_APPS.

storage=B2Storage() in models, turn it off during tests. See https://github.com/pyutil/django-b2/issues/4 To work as real B2Storage, 'B2Storage' string must be contained in settings.DEFAULT_FILE_STORAGE. Otherwise DEFAULT_FILE_STORAGE will be instantiated. This makes following in tests possible: @override_settings(DEFAULT_FILE_STORAGE='django.core.files.storage.FileSystemStorage')

CHAPTER 4

Contributing

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

You can contribute in many ways:

4.1 Types of Contributions

4.1.1 Report Bugs

Report bugs at https://github.com/pyutil/django-b2/issues.

If you are 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.

4.1.2 Fix Bugs

Look through the GitHub issues for bugs. Anything tagged with “bug” is open to whoever wants to implement it.

4.1.3 Implement Features

Look through the GitHub issues for features. Anything tagged with “feature” is open to whoever wants to implement it.

4.1.4 Write Documentation

django-b2 could always use more documentation, whether as part of the official django-b2 docs, in docstrings, or even on the web in blog posts, articles, and such.

4.1.5 Submit Feedback

The best way to send feedback is to file an issue at https://github.com/pyutil/django-b2/issues.

If you are proposing a feature:

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

4.2 Get Started!

Ready to contribute? Here's how to set up *django-b2* for local development. This is a standard workflow. You can find similar alternative description in github's README.md.

1. Fork the *django-b2* repo on GitHub.
2. Clone your fork locally:

```
$ git clone git@github.com:your_name_here/django-b2.git
```

3. Install your local copy into a virtualenv. Assuming you have virtualenvwrapper installed, this is how you set up your fork for local development:

```
$ mkvirtualenv django-b2
$ cd django-b2/
$ python setup.py develop
```

4. Create a branch for local development:

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

   Now you can make your changes locally.

5. When you're done making changes, check that your changes pass flake8 and the tests, including testing other Python versions with tox:

```
  $ flake8 django_b2 tests
  $ python setup.py test
  $ tox

To get flake8 and tox, just pip install them into your virtualenv: pip install -r␣
↪requirements_test.txt
```

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

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

7. Submit a pull request through the GitHub website.

4.3 Pull Request Guidelines

Before you submit a pull request, check that it meets these guidelines:

1. The pull request should include tests.
2. If the pull request adds functionality, the docs should be updated. Put your new functionality into a function with a docstring, and add the feature to the list in README.rst.
3. The pull request should work for new Python 3 versions and for PyPy (optional). Check tox.ini and make sure that the tests pass for all supported Python versions.

CHAPTER 5

Credits

5.1 Development Lead

- Mirek Zvolský <zvolsky@seznam.cz>

5.2 Contributors

None yet. Why not be the first?

CHAPTER 6

History

6.1 0.7.0 (2020-12-29)

- new Django setting B2_FORCE_UNIQUE and new underlaying constructor parameter BackBlazeB2(force_unique=True)

6.2 0.6.2 (2020-11-04)

- B2Storage() initializes as the settings.DEFAULT_FILE_STORAGE if 'B2Storage' string is not inside can be used together with @override_settings(DEFAULT_FILE_STORAGE='django.core.files.storage.FileSystemStorage') see https://github.com/pyutil/django-b2/issues/4

6.3 0.6.0 (2020-05-28)

- lazy loading, to avoid running code during collectstatic,.. - https://github.com/pyutil/django-b2/issues/3

6.4 0.5.5 (2020-05-24)

- bugfix: upload on Windows, thx Same Weaver, https://github.com/pyutil/django-b2/issues/2
- Linux abs filenames: leading "/" will be removed so we can use local abs names 1:1 to upload to b2 (in Windows: C:/.. is valid name)

6.5 0.5.0 (2020-02-17)

- can work with django-tenant schemas, tenant aware storage django_b2.tenant_storage.TenantB2Storage

6.6 0.4.0 (2020-02-10)

- older local media (see B2_LOCAL_MEDIA) can be cleared with management command b2_clear_local_media
- B2_LOCAL_CACHE setting renamed to B2_LOCAL_MEDIA, possible values changed to =”ML”

6.7 0.3.0 (2020-02-08) - do not use

- !! new B2_LOCAL_MEDIA setting was in 0.3.0 named incompatible as B2_LOCAL_CACHE=”FM”
- B2_LOCAL_MEDIA setting to make a local copy of files. So you can have local instances backuped on backblaze.
- B2_LOCAL_MEDIA prevents failures if the django application want immediately reopen the file (imagekits creating thumbnails, Wagtail is an example)

6.8 0.2.0 (2020-01-31)

- backblaze_b2.py can be called as script to upload single file.
- pgtob2.sh script to backup postgres database

6.9 0.1.5 (2020-01-02)

- No code change. Minor docs changes.

6.10 0.1.4 (2020-01-02)

- First release on PyPI.