# django-call-after-commit documentation

*Release*

**Thread**

May 29, 2014

## Contents

CHAPTER 1

# Call methods after the current database transaction has been committed

This project provides the ability for Django applications to mark methods to be called immediately after the current database transaction has committed.

This has two main use-cases:

1. You wish to perform an asynchronous task but your queue runner will not "see" the updated state of your database due to transaction isolation.

   For example, when registering a new user you might wish to download an avatar from a third-party provider without blocking the current request. In this scenario, your queue job may non-deterministically race and start executing without the new user actually being "on disk" yet and thus not appearing to exist, resulting in a failed job that would have run without issue milliseconds later.

2. You wish to execute potentially dangerous side-effects only after you are sure that you have really committed to a particular version of "truth".

   For example, if you send emails within a transaction and that transaction fails to commit, the sent email will still have been sent, causing potential confusion if the email only "makes sense"–or works at all–if the database was changed permanently. Worst still, these emails may be duplicated later if your application's logic relied on a "sent" flag being changed as part of the failed transaction - it will appear to your project that the email is yet to be sent. This behaviour can be compounded in cronjobs, resulting in 1000s of duplicate emails.

## 1.1 Installation

1. Add `django_call_after_commit.middleware.CallAfterCommitMiddleware` to `MIDDLEWARE_CLASSES`. It should go after `django.middleware.transaction.TransactionMiddleware` if you are using an older version of Django.

## 1.2 Usage

Instead of calling your method using:

```
some_method(arg1, arg2)
```

use:

```
from django_call_after_commit import call_after_commit

call_after_commit(some_method, arg1, arg2)
```

It is save to use `call_after_commit` outside of a "request" context - your method will simply be called immediately.

## 1.3 Configuration

(None)

## 1.4 Links

**Homepage/documentation:** https://django-call-after-commit.readthedocs.org/

**View/download code** https://github.com/thread/django-call-after-commit

**File a bug** https://github.com/thread/django-call-after-commit/issues

Figure 1.1: See more open source projects from Thread.com