Flask-Cachual Documentation

Release 0.2.0

Alex Landau

January 17, 2017

Contents

Flask-Cachual is a Flask extension for the Cachual library. It allows you to cache your function's return values with a simple decorator:

```
from flask_cachual import cached

@cached(ttl=300) # 5 minutes
def get_user_email(user_id):
    ...
```

It's that easy!

CHAPTER 1

Installation

Install the extension with pip:

```
$ pip install flask-cachual
```

CHAPTER 2

Usage

You can initialize *Cachual* directly:

```
from flask import Flask
from flask_cachual import Cachual

app = Flask()
Cachual(app)
```

Or, you can defer initialization if you are utilizing the application factory pattern, using the *init_app()* method:

```
from flask import Flask
from flask_cachual import Cachual

cachual = Cachual()

def create_app():
    app = Flask()
    cachual.init_app(app)
```

That's it! Now, anywhere in your application code, you can use the *cached* decorator on any function whose value you want to cache. For example:

```
@app.route('/<userId>/email')
def user_email(userId):
    return jsonify({'email': get_user_email(userId)})

@cached(ttl=300)
def get_user_email(userId):
    ...
```

Whenever the decorated method is called, your app will check the configured cache for the result. If there's a cache hit, it's immediately returned; otherwise the function will execute as normal, the value will be stored in the cache with a unique key, and the value will be returned. Make sure you read Cachual's documentation carefully to understand what values can be cached, how it works, and how to deal with more complex data types.

Speaking of configuration...

CHAPTER 3

Configuration

You need to specify two configuration values in your application's config:

`CACHUAL_TYPE`	The type of cache you want to use. Currently only two values are supported: `redis` which corresponds to `RedisCache` and `memcached` which corresponds to `MemcachedCache`.
`CACHUAL_ARGS`	Dictionary of arguments to initialize the the Cachual cache with. If `None`, the cache's defaults will be used.

CHAPTER 4

API Documentation

class flask_cachual.**Cachual**(*app=None*)

This object ties the Flask application object to the Cachual library by setting the cachual_cache attribute of the application instance to the Cachual cache as specified by the application's configuration.

Parameters **app** (*Flask*) – The Flask application object to initialize.

init_app(*app*)

Configure the application to use Cachual. Based on the application's configuration, this will instantiate the cache and give the application object access to it via the cachual_cache attribute. See *Configuration*.

Parameters **app** (*Flask*) – The Flask application object to initialize.

flask_cachual.**cached**(*ttl=None*, *pack=None*, *unpack=None*, *use_class_for_self=False*)

Functions decorated with this will have their value cached via the Cachual library. This is basically just a proxy to Cachual's cached() decorator. It ensures that the correct cache is used based on the current_app context.

f

C

F

I