eBook Gratuit

APPRENEZ celery

eBook gratuit non affilié créé à partir des **contributeurs de Stack Overflow.**

#celery

Table des matières

À propos

You can share this PDF with anyone you feel could benefit from it, downloaded the latest version from: celery

It is an unofficial and free celery ebook created for educational purposes. All the content is extracted from Stack Overflow Documentation, which is written by many hardworking individuals at Stack Overflow. It is neither affiliated with Stack Overflow nor official celery.

Use the content presented in this book at your own risk; it is not guaranteed to be correct nor accurate, please send your feedback and corrections to info@zzzprojects.com

Chapitre 1: Commencer avec le céleri

Remarques

"Celery est une file d'attente de tâches / file d'attente de tâches asynchrone basée sur le passage de messages distribués." - http://www.celeryproject.org/

Le céleri est idéal pour les tâches d'arrière-plan asynchrones et programmées. Il est couramment utilisé pour les tâches de longue durée faisant partie d'une application Django ou Flask.

Examples

Installation ou configuration

Vous pouvez installer Celery via l'index de package Python (PyPI) ou à partir de sources.

Pour installer la dernière version en utilisant `pip` :

```
$ pip install celery
```

Pour installer avec `easy_install` :

```
$ easy_install celery
```

Téléchargement et installation depuis le source

Téléchargez la dernière version de Celery depuis http://pypi.python.org/pypi/celery/

Vous pouvez l'installer en procédant comme suit:

```
$ tar xvfz celery-0.0.0.tar.gz
$ cd celery-0.0.0
$ python setup.py build
# python setup.py install # as root
```

Céleri + Redis

Installation

Des dépendances supplémentaires sont requises pour le support Redis. Installez Celery et les dépendances en une seule fois en utilisant le bundle `celery[redis]` :

```
$ pip install -U celery[redis]
```

Configuration

Configurez l'emplacement de votre base de données Redis:

```
BROKER_URL = 'redis://localhost:6379/0'
```

L'URL doit avoir le format suivant:

```
redis://:password@hostname:port/db_number
```

Application

Créez le fichier tasks.py:

```
from celery import Celery

BROKER_URL = 'redis://localhost:6379/0'
app = Celery('tasks', broker=BROKER_URL)

@app.task
def add(x, y):
    return x + y
```

Le premier argument de `Celery` est le nom du module actuel. De cette façon, les noms peuvent être générés automatiquement. Le second argument est le mot clé `broker` qui spécifie l'URL du courtier de messages.

Exécution du serveur de travail celery

Exécutez le travailleur en exécutant avec l'argument worker:

```
$ celery -A tasks worker --loglevel=info
```

Appeler la tâche

Pour appeler la tâche, utilisez la méthode `delay()` .

```
>>> from tasks import add
>>> add.delay(4, 4)
```

L'appel d'une tâche renvoie une instance AsyncResult , qui peut vérifier l'état de la tâche, attendre la fin de la tâche ou obtenir sa valeur de retour. (Si la tâche a échoué, elle obtient l'exception et la traceback).

Garder les résultats

Pour garder une trace des états de la tâche, Celery doit stocker ou envoyer les états quelque part. Utilisez Redis comme moteur de résultat:

```
BROKER_URL = 'redis://localhost:6379/0'
BACKEND_URL = 'redis://localhost:6379/1'
app = Celery('tasks', broker=BROKER_URL, backend=BACKEND_URL)
```

Pour en savoir plus sur les backends de résultats, voir Résultat Backends .

Maintenant que le moteur de résultat est configuré, appelez à nouveau la tâche. Cette fois, maintenez l'instance AsyncResult renvoyée par la tâche:

```
>>> result = add.delay(4, 4)
```

La méthode `ready()` retourne si le traitement est terminé ou non:

```
>>> result.ready()
False
```

Il est possible d'attendre que le résultat soit complet, mais cela est rarement utilisé car il transforme l'appel asynchrone en un appel synchrone:

```
>>> result.get(timeout=1)
8
```

Basé sur le document officiel de céleri

Lire Commencer avec le céleri en ligne: https://riptutorial.com/fr/celery/topic/6987/commencer-avec-le-celeri

Chapitre 2: Surveillance du céleri avec fleur

Examples

En marche avec fleur

Flower est un outil basé sur le Web pour surveiller le céleri.

Pour installer Flower, nous pouvons utiliser `pip` comme suit:

```
pip install flower
```

Pour lancer Flower pour `proj` :

```
celery –A proj flower
```

Maintenant, la fleur est accessible dans:

```
http://localhost:5555
```

Lire Surveillance du céleri avec fleur en ligne:
https://riptutorial.com/fr/celery/topic/7477/surveillance-du-celeri-avec-fleur

Crédits

S. No	Chapitres	Contributeurs
1	Commencer avec le céleri	4444, Community, jamjar, jegesh, Majid
2	Surveillance du céleri avec fleur	ettanany