EBook Gratis

APRENDIZAJE celery

Free unaffiliated eBook created from
Stack Overflow contributors.

#celery

Tabla de contenido

Acerca de

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

Capítulo 1: Empezando con el apio

Observaciones

"El apio es una cola de tareas / cola de tareas asíncronas basada en el paso de mensajes distribuidos". - http://www.celeryproject.org/

El apio es ideal para tareas de fondo asíncronas y programadas. Se usa comúnmente para tareas de larga duración que forman parte de una aplicación Django o Flask.

Examples

Instalación o configuración

Puede instalar Celery a través del Índice de paquetes de Python (PyPI) o desde la fuente.

Para instalar la última versión usando `pip` :

```
$ pip install celery
```

Para instalar utilizando `easy_install` :

```
$ easy_install celery
```

Descargando e instalando desde la fuente

Descargue la última versión de Celery desde http://pypi.python.org/pypi/celery/

Puedes instalarlo haciendo lo siguiente:

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

Apio + Redis

Instalación

Se requieren dependencias adicionales para el soporte de Redis. Instale tanto el apio como las dependencias de una sola vez utilizando el paquete de `celery[redis]` :

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

Configuración

Configure la ubicación de su base de datos Redis:

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

La URL debe estar en el formato de:

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

Solicitud

Crea el archivo task.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
```

El primer argumento de `Celery` es el nombre del módulo actual. De esta manera los nombres se pueden generar automáticamente. El segundo argumento es la palabra clave de `broker` que especifica la URL del intermediario de mensajes.

Ejecutando el servidor de apio trabajador

Ejecuta el trabajador ejecutando con el argumento del trabajador:

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

Llamando a la tarea

Para llamar a la tarea, use el método `delay()` .

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

La llamada a una tarea devuelve una instancia de AsyncResult , que puede verificar el estado de la tarea, esperar a que la tarea finalice o obtener su valor de retorno. (Si la tarea falla, obtiene la excepción y el rastreo).

Manteniendo resultados

Para mantener un registro de los estados de la tarea, Celery necesita almacenar o enviar los estados a algún lugar. Utilice Redis como el resultado backend:

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

Para leer más sobre los backends del resultado, por favor vea Backends del resultado .

Ahora, con el resultado configurado, vuelva a llamar a la tarea. Esta vez retén la instancia AsyncResult devuelta de la tarea:

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

El método `ready()` devuelve si la tarea ha terminado de procesarse o no:

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

Es posible esperar a que se complete el resultado, pero esto rara vez se usa, ya que convierte la llamada asíncrona en una síncrona:

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

Basado en el documento oficial de apio.

Lea Empezando con el apio en línea: https://riptutorial.com/es/celery/topic/6987/empezando-con-el-apio

Capítulo 2: Monitoreo de apio con flor.

Examples

En marcha con Flower

Flower es una herramienta basada en web para monitorear el apio.

Para instalar Flower, podemos usar `pip` siguiente manera:

```
pip install flower
```

Para ejecutar Flower para `proj` :

```
celery -A proj flower
```

Ahora, la flor es accesible en:

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

Lea Monitoreo de apio con flor. en línea: https://riptutorial.com/es/celery/topic/7477/monitoreo-de-apio-con-flor-

Creditos

S. No	Capítulos	Contributors
1	Empezando con el apio	4444, Community, jamjar, jegesh, Majid
2	Monitoreo de apio con flor.	ettanany