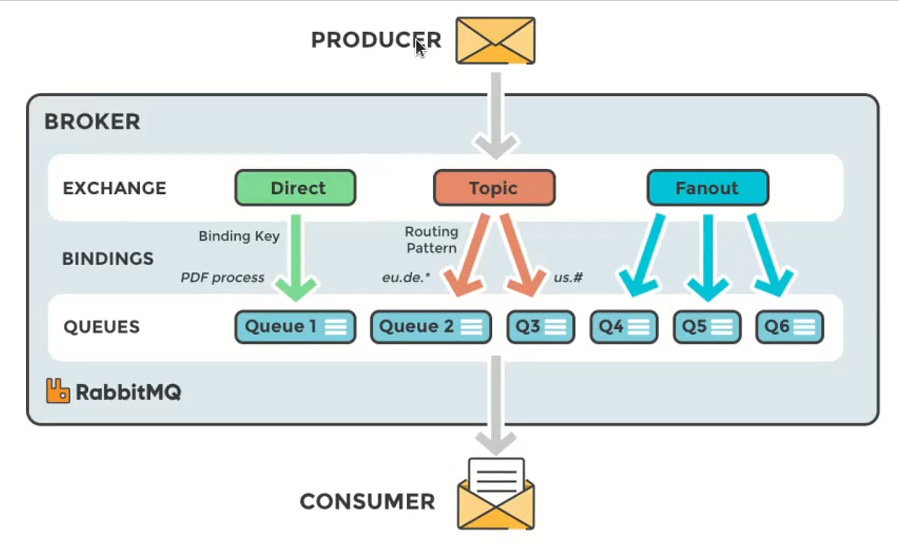
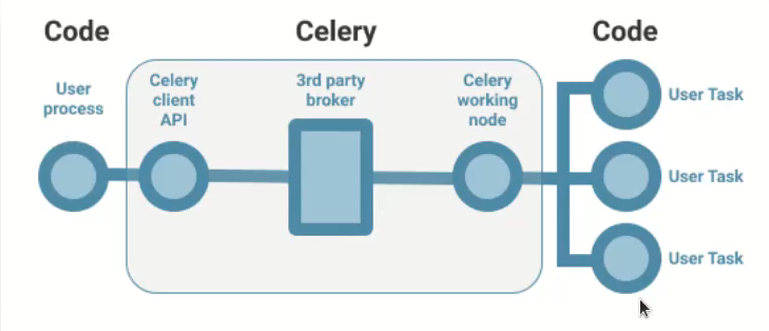
celery ابزاریست برای ساده تر کردن ارتباط بین producer و consumer با broker. برای نصب کردن celery میتونید از دستور pip install celery استفاده کنید. برای استفاده از celery حتما باید از یک broker استفاده کنید. پیشنهاد celery استفاده از rabbitmq به عنوان broker میباشد اما میتوانید از redis, couchdb, mongodb یا تمام دیتابیس هایی که از SqlAlchemy و django orm استفاده میکنند، استفاده کنید.





===========================================================

برای شروع کار با celery باید از کلاس Celery استفاده کنید و از این کلاس یک نمونه جدید ایجاد کنید. به هر نمونه ای که از کلاس Celery ایجاد میشود application یا بطور خلاصه app گفته میشود. تمامی appها باید یک اسم داشته باشند که پیشنهاد میشود همنام ماژول باشد. بعد از ایجاد کردن app با استفاده از دکوریتور task میتوانید فانکشن های خود را به صورت async صدا زنید.

app = Celery('one', broker="amqp://localhost")

ــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــ

برای اینکه celery متوجه شود که functionی که قرار است از آن استفاده کند کدام است کافیست در بالای فانکشن کد زیر را قرار دهید:

@app.task

def add(x, y):

الان celery متوجه می‌شود که باید حواسش به این فانکشن باشد و کارهارو صف بندی کند

ــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــ

با دستور زیر میتوانیم پروژه ای که مربوط به celery میباشد را اجرا کنیم

celery -A one worker

توجه شود که one همان one.py میباشد

ــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــ

هنگامیکه دستور بالا را اجرا میکنید یکسری پیغام ها نمایش داده می‌شوند در قسمتی از این پیغام ها خط زیر وجود دارد

**app:         one:0x7fa0a9ea4370**

oneی که جلویapp نوشته شده است همان اسمی است که هنگام ساخت یک inctanceی از کلاس Celery به آن دادید

ـــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــ

در قسمت دیگری از خروجی عبارت زیر را مشاهده میکنید:

**transport:   amqp://guest:**[**\*\*@localhost**](mailto:**@localhost)**:5672//**

amqp که نشان دهنده rabbitmq میباشد و \*\*:quest نشان دهنده username و پسورد میباشد که در config اولیه rabbitmq یوزرنیم و پسوردی quest و quest میباشد لازم نیست قرار دهید

ــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــ

**concurrency :** میزان کاری است که سلری میتواند بطور همزمان انجام دهد

ـــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــ

آپشنی که در worker وجود دار l میباشد با این آپشن به celery میگویید که چه نوع loglevelی را برای من نمایش بده:

celery -A one worker -l info

الان یعنی تمام logهایی که levelشان از info بالاتر هستند را نمایش بده

===========================================================

در celery به طور کلی دو متد apply\_async و delay برای صدا زدن workerها وجود دارد. هر دوی این متدها یک کار را انجام میدهند اما با استفاده از apply\_async شما میتوانید یکسری آپشن های بیشتری را برای صدا زدن wrokerها استفاده کنید. مثلا:

countdown=10 ==> یعنی تو 10 ثانیه صبر کن بعد دستورات را ارسال کن

expires=120 ==> بعد از 120 ثانیه اگر تسک اجرا نشد آن را از صف حذف کن

و نحوه ی ارسال اطلاعات نیز متفاوت میباشد

مثال:

add.apply\_async(args=[5, 9])  
add.deley(5, 9)

===========================================================

flower یک ابزار گرافیکی برای مدیریت کردن اجزای celery مثل taskها و workerها است. flower در مرورگر اجرا میشود.

$ celery flower -A proj --broker=amqp://guest:guest@localhost:5672//

===========================================================

محیط های گرافیکی به دلیل مصرف منابع زیاد آنچنان مورد پسند برنامه نویس ها نیستند به همین دلیل امکان این وجود دارد که شما بتوانید اجزای مختلف celery را با استفاده از command line مدیریت کنید