1. `conda env list`
2. `activate emma`
3. `pip install celery`
4. `pip install redis`
5. 在当前目录下编写tasks.py

```
from celery import Celery
broker = "redis://127.0.0.1:6379/0"
backend = "redis://127.0.0.1:6379/0"
app = Celery("tasks", broker=broker, backend=backend)
@app.task
def add(x, y):
return x+y
```

1. `celery -A tasks worker --loglevel=info` 启动Celery Worker开始监听并执行任务
   其中 `-A` 表示的是Celery APP的名称，指的是tasks.py，tasks是APP的名称，worker是一个执行任务角色，loglevel=info记录日志类型默认是info，这个命令启动了一个worker，用来执行程序中add这个加法任务

```
[2019-02-18 21:54:53,370: INFO/MainProcess] mingle: all alone
[2019-02-18 21:54:53,380: INFO/MainProcess] celery@DESKTOP-
P278GU7 ready.
```

可以看到Celery正常工作在名称为DESKTOP-P278GU7的主机上，当前APP是tasks，运输工具是在程序中设置的中间人redis://127.0.0.1:6379/0
此时重新打开一个终端，执行Python，进入Python交互界面，用delay()方法调用任务

1. 调用任务

```
from tasks import add
add.delay(6, 6)
```

1. 结果

```
[2019-02-18 22:03:27,252: INFO/MainProcess] Received task:
tasks.add[211c316c-d9ea-4301-81e9-a21212a3992e]
[2019-02-18 22:03:27,256: ERROR/MainProcess] Task handler raised
error: ValueError('not enough values to unpack (expected 3, got
0)',)
Traceback (most recent call last):
  File "c:\users\lab\anaconda3\envs\emma\lib\site-
packages\billiard\pool.py", line 358, in workloop
    result = (True, prepare_result(fun(*args, **kwargs)))
  File "c:\users\lab\anaconda3\envs\emma\lib\site-
packages\celery\app\trace.py", line 537, in _fast_trace_task
    tasks, accept, hostname = _loc
ValueError: not enough values to unpack (expected 3, got 0)
```

第一行表明worker收到一个任务：tasks.add

参考链接

error

```
tasks, accept, hostname = _loc
ValueError: not enough values to unpack (expected 3, got 0)
```

解决方法：

```
在两个命令行终端输入：
set FORKED_BY_MULTIPROCESSING = 1
```

如下：

```
(emma) E:\emma>set FORKED_BY_MULTIPROCESSING=1
```

```
(emma) E:\emma>celery -A tasks worker --loglevel=info

 -------------- celery@DESKTOP-P278GU7 v4.2.1 (windowlicker)
---- **** -----
--- * *** * -- Windows-10-10.0.17134-SP0 2019-02-19 14:57:37
-- * - **** ---
- ** ---------- [config]
- ** ---------- .> app: tasks:0x20a4efdfcf8
- ** ---------- .> transport: redis://127.0.0.1:6379/0
- ** ---------- .> results: redis://127.0.0.1:6379/0
- *** --- * --- .> concurrency: 8 (prefork)
-- ******* ---- .> task events: OFF (enable -E to monitor tasks
in this worker)
--- ***** -----
 -------------- [queues]
                .> celery exchange=celery(direct) key=celery


[tasks]
  . tasks.add

[2019-02-19 14:57:37,234: INFO/MainProcess] Connected to
redis://127.0.0.1:6379/0
[2019-02-19 14:57:37,268: INFO/MainProcess] mingle: searching for
neighbors
[2019-02-19 14:57:38,115: INFO/SpawnPoolWorker-3] child process
7124 calling self.run()
[2019-02-19 14:57:38,115: INFO/SpawnPoolWorker-1] child process
15932 calling self.run()
[2019-02-19 14:57:38,115: INFO/SpawnPoolWorker-4] child process
18320 calling self.run()
[2019-02-19 14:57:38,115: INFO/SpawnPoolWorker-6] child process
17664 calling self.run()
[2019-02-19 14:57:38,131: INFO/SpawnPoolWorker-5] child process
4384 calling self.run()
[2019-02-19 14:57:38,134: INFO/SpawnPoolWorker-8] child process
14528 calling self.run()
[2019-02-19 14:57:38,134: INFO/SpawnPoolWorker-7] child process
7600 calling self.run()
[2019-02-19 14:57:38,134: INFO/SpawnPoolWorker-2] child process
13768 calling self.run()
```

```
[2019-02-19 14:57:38,316: INFO/MainProcess] mingle: all alone
[2019-02-19 14:57:38,338: INFO/MainProcess] celery@DESKTOP-P278GU7 ready.
[2019-02-19 14:58:08,367: INFO/MainProcess] Received task: tasks.add[eedda814-3404-4c16-8e9f-5a97c9ff4bd0]
[2019-02-19 14:58:08,378: INFO/SpawnPoolWorker-3] Task tasks.add[eedda814-3404-4c16-8e9f-5a97c9ff4bd0] succeeded in 0.015000000013969839s: 4
```

异步端

```

(emma) E:\emma>set FORKED_BY_MULTIPROCESSING=1

(emma) E:\emma>python
Python 3.6.2 |Continuum Analytics, Inc.| (default, Jul 20 2017, 12:30:02) [MSC v.1900 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> from tasks import add
>>> add.delay(2,2)
<AsyncResult: eedda814-3404-4c16-8e9f-5a97c9ff4bd0>
>>>
```

或者新建环境变量`FORKED_BY_MULTIPROCESSING`，将其值设置为1.