Install RabbitMQ on Mac OS X and test with simplest possible messaging app

```
# Install RabbitMQ on Mac OS X and test with simplest possible messaging app
_____________________________________________________________________
                                                                     |
dyn-128-104-18-177:opt khazelton$ brew update                        |
_____________________________________________________________________|
Updated Homebrew from 45bfd2b to 2c35d2c.
Updated 1 tap (homebrew/core).
...
_____________________________________________________________________
                                                                     |
dyn-128-104-18-177:opt khazelton$ brew install rabbitmq              |
_____________________________________________________________________|
==> Installing dependencies for rabbitmq: xz, openssl, jpeg, libpng, libtiff, wxmac, erlang
==> Installing rabbitmq dependency: xz
==> Downloading https://homebrew.bintray.com/bottles/xz-5.2.2.el_capitan.bottle.tar.gz
######################################################################## 100.0%
==> Pouring xz-5.2.2.el_capitan.bottle.tar.gz
  /usr/local/Cellar/xz/5.2.2: 91 files, 1.4M
==> Installing rabbitmq dependency: openssl
==> Downloading https://homebrew.bintray.com/bottles/openssl-1.0.2h_1.el_capitan.bottle.tar.gz
######################################################################## 100.0%
==> Pouring openssl-1.0.2h_1.el_capitan.bottle.tar.gz
==> Using the sandbox
==> Caveats
A CA file has been bootstrapped using certificates from the system
keychain. To add additional certificates, place .pem files in
  /usr/local/etc/openssl/certs
and run
  /usr/local/opt/openssl/bin/c_rehash
This formula is keg-only, which means it was not symlinked into /usr/local.
Apple has deprecated use of OpenSSL in favor of its own TLS and crypto libraries
Generally there are no consequences of this for you. If you build your
own software and it requires this formula, you''ll need to add to your
build variables:
    LDFLAGS:  -L/usr/local/opt/openssl/lib
    CPPFLAGS: -I/usr/local/opt/openssl/include
==> Summary
  /usr/local/Cellar/openssl/1.0.2h_1: 1,691 files, 12M
==> Installing rabbitmq dependency: jpeg
==> Downloading https://homebrew.bintray.com/bottles/jpeg-8d.el_capitan.bottle.2.tar.gz
######################################################################## 100.0%
==> Pouring jpeg-8d.el_capitan.bottle.2.tar.gz
  /usr/local/Cellar/jpeg/8d: 19 files, 713.8K
==> Installing rabbitmq dependency: libpng
==> Downloading https://homebrew.bintray.com/bottles/libpng-1.6.24.el_capitan.bottle.tar.gz
######################################################################## 100.0%
==> Pouring libpng-1.6.24.el_capitan.bottle.tar.gz
  /usr/local/Cellar/libpng/1.6.24: 26 files, 1.2M
==> Installing rabbitmq dependency: libtiff
==> Downloading https://homebrew.bintray.com/bottles/libtiff-4.0.6_2.el_capitan.bottle.tar.gz
######################################################################## 100.0%
==> Pouring libtiff-4.0.6_2.el_capitan.bottle.tar.gz
  /usr/local/Cellar/libtiff/4.0.6_2: 261 files, 3.4M
==> Installing rabbitmq dependency: wxmac
==> Downloading https://homebrew.bintray.com/bottles/wxmac-3.0.2_2.el_capitan.bottle.1.tar.gz
######################################################################## 100.0%
==> Pouring wxmac-3.0.2_2.el_capitan.bottle.1.tar.gz
  /usr/local/Cellar/wxmac/3.0.2_2: 809 files, 23.6M
==> Installing rabbitmq dependency: erlang
==> Downloading https://homebrew.bintray.com/bottles/erlang-19.0.2.el_capitan.bottle.tar.gz
######################################################################## 100.0%
==> Pouring erlang-19.0.2.el_capitan.bottle.tar.gz
==> Caveats
Man pages can be found in:
  /usr/local/opt/erlang/lib/erlang/man
```

```
Access them with `erl -man`, or add this directory to MANPATH.
==> Summary
  /usr/local/Cellar/erlang/19.0.2: 7,292 files, 278.6M
==> Installing rabbitmq
==> Downloading https://www.rabbitmq.com/releases/rabbitmq-server/v3.6.4/rabbitmq-server-generic-unix-3.6.4.tar.
xz
######################################################################## 100.0%
==> /usr/bin/unzip -qq -j /usr/local/Cellar/rabbitmq/3.6.4/plugins/rabbitmq_management-3.6.4.ez
rabbitmq_management-3.6.4/priv/www/c
==> Caveats
Management Plugin enabled by default at http://localhost:15672
Bash completion has been installed to:
  /usr/local/etc/bash_completion.d
To have launchd start rabbitmq now and restart at login:
  brew services start rabbitmq
Or, if you don''t want/need a background service you can just run:
  rabbitmq-server
==> Summary
  /usr/local/Cellar/rabbitmq/3.6.4: 186 files, 5.8M, built in 8 seconds
_____________________________________________________________________
                                                                     |
dyn-128-104-18-177:opt khazelton$ brew services start rabbitmq       |
_____________________________________________________________________|
==> Tapping homebrew/services
Cloning into '/usr/local/Library/Taps/homebrew/homebrew-services'...
remote: Counting objects: 7, done.
remote: Compressing objects: 100% (6/6), done.
remote: Total 7 (delta 0), reused 3 (delta 0), pack-reused 0
Unpacking objects: 100% (7/7), done.
Checking connectivity... done.
Tapped 0 formulae (32 files, 46.0K)
==> Successfully started `rabbitmq` (label: homebrew.mxcl.rabbitmq)
_____________________________________________________________________
                                                                     |
dyn-128-104-18-177:opt khazelton$ sudo pip install pika              |
_____________________________________________________________________|
Collecting pika
  Downloading pika-0.10.0-py2.py3-none-any.whl (92kB)
    100% || 102kB 1.9MB/s
Installing collected packages: pika
Successfully installed pika-0.10.0
_____________________________________________________________________
                                                                     |
dyn-128-104-18-177:opt khazelton$ cat send.py                        |
_____________________________________________________________________|
#!/usr/bin/env python
import pika
connection = pika.BlockingConnection(pika.ConnectionParameters(
               'localhost'))
channel = connection.channel()
channel.queue_declare(queue='hello')
channel.basic_publish(exchange='',
                      routing_key='hello',
                      body='Hello World!')
print(" [x] Sent 'Hello World!'")
connection.close()
_____________________________________________________________________
                                                                     |
dyn-128-104-18-177:opt khazelton$ python send.py                     |
_____________________________________________________________________|
 [x] Sent 'Hello World!'

 _____________________________________________________________________
                                                                     |
dyn-128-104-18-177:opt khazelton$ cat receive.py                     |
_____________________________________________________________________|
#!/usr/bin/env python
import pika
connection = pika.BlockingConnection(pika.ConnectionParameters(
        host='localhost'))
channel = connection.channel()
```

```
channel.queue_declare(queue='hello')
def callback(ch, method, properties, body):
    print(" [x] Received %r" % body)
channel.basic_consume(callback,
                      queue='hello',
                      no_ack=True)
print(' [*] Waiting for messages. To exit press CTRL+C')
channel.start_consuming()
```

```
dyn-128-104-18-177:opt khazelton$ python receive.py
 [*] Waiting for messages. To exit press CTRL+C
 [x] Received 'Hello World!'
^C
KeyboardInterrupt
```