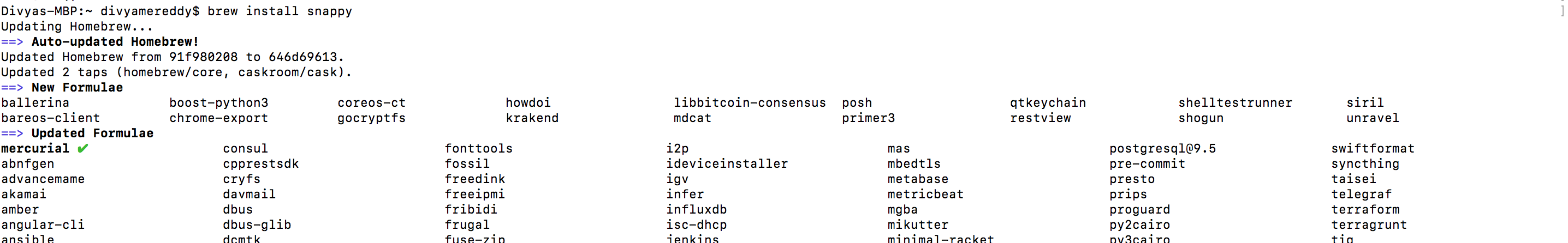
Installation Snappy:



**python:**

**Strat server:**

#!/usr/bin/env python

# Licensed to the Apache Software Foundation (ASF) under one

# or more contributor license agreements. See the NOTICE file

# distributed with this work for additional information

# regarding copyright ownership. The ASF licenses this file

# to you under the Apache License, Version 2.0 (the

# "License"); you may not use this file except in compliance

# with the License. You may obtain a copy of the License at

#

# http://www.apache.org/licenses/LICENSE-2.0

#

# Unless required by applicable law or agreed to in writing, software

# distributed under the License is distributed on an "AS IS" BASIS,

# WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.

# See the License for the specific language governing permissions and

# limitations under the License.

#from BaseHTTPServer import BaseHTTPRequestHandler, HTTPServer

import http.server

import socketserver

import avro.ipc as ipc

import avro.protocol as protocol

import avro.schema as schema

PROTOCOL = protocol.Parse(open("../avro/mail.avpr").read())

class MailResponder(ipc.Responder):

def \_\_init\_\_(self):

ipc.Responder.\_\_init\_\_(self, PROTOCOL)

def invoke(self, msg, req):

if msg.name == 'send':

message = req['message']

return ("Sent message to " + message['to']

+ " from " + message['from']

+ " with body " + message['body'])

else:

raise schema.AvroException("unexpected message:", msg.getname())

Handler = http.server.SimpleHTTPRequestHandler

class MailHandler(Handler):

def do\_POST(self):

self.responder = MailResponder()

call\_request\_reader = ipc.FramedReader(self.rfile)

call\_request = call\_request\_reader.read\_framed\_message()

resp\_body = self.responder.respond(call\_request)

self.send\_response(200)

self.send\_header('Content-Type', 'avro/binary')

self.end\_headers()

resp\_writer = ipc.FramedWriter(self.wfile)

resp\_writer.write\_framed\_message(resp\_body)

server\_addr = ('localhost', 9090)

if \_\_name\_\_ == '\_\_main\_\_':

server = socketserver.TCPServer(server\_addr, MailHandler)

server.allow\_reuse\_address = True

server.serve\_forever()

****

***Run hello world:***

import sys

import http.client as client

import avro.ipc as ipc

import avro.protocol as protocol

PROTOCOL = protocol.Parse(open("../avro/mail.avpr").read())

server\_addr = ('localhost', 9090)

class UsageError(Exception):

def \_\_init\_\_(self, value):

self.value = value

def \_\_str\_\_(self):

return repr(self.value)

if \_\_name\_\_ == '\_\_main\_\_':

if len(sys.argv) != 4:

raise UsageError('Usage: <to> <from> <body>')

# client code - attach to the server and send a message

client = ipc.HTTPTransceiver(server\_addr[0], server\_addr[1])

requestor = ipc.Requestor(PROTOCOL, client)

# fill in the Message record and send it

message = dict()

message['to'] = sys.argv[1]

message['from'] = sys.argv[2]

message['body'] = sys.argv[3]

params = dict()

params['message'] = message

print ("Result: " + requestor.request('send', params))

# cleanup

client.close()

