notes Documentation

1

leesea

6 19, 2017

Contents

CHAPTER 1

python

- int to binary

```
>>> bin(6)
'0b110'
>>> bin(6)[2:]
'110'
```

- binary to int

```
>>> int('11111111', 2)
255

>>> from bitstring import BitArray
>>> b = Bitarray(bin='11111111')
>>> b.uint
255
>>> b.int
-1
```

CHAPTER 2

import

python import

```
-- src
   |-- m1.py
   |-- mod2
   |     `-- m2.py
   |-- mod3
   |     `-- m3.py
   `-- main.py
```

- `main.py` import `m1.py`

```
import m1
```

- `main.py` import `m2.py`

  import, `ImportError: No module named ..`

```
>>> import mod2.m2
ImportError: No module named mod2.m2
```

  python __init__.py python package. mod2 `__init__.py`,

```
-- src
   |-- m1.py
   |-- mod2
   |     |-- __init__.py
   |     `-- m2.py
   |-- mod3
   |     |-- __init__.py
   |     `-- m3.py
   `-- main.py
```

  import

```
>>> import mod2.m2
```

```
>>> import mod2
>>> dir(mod2)
['__builtins__', '__doc__', '__file__', '__name__', '__package__', '__path__']
```

`dir` m2.py, import `mod2.__init__.py` `dir(mod2)` m2.py __init__.py import m2

- `m2` import `m3`

, `m2.py`

```
import sys
sys.append('..')
import m3
```

`main.py` `main.py` m2,

```
import mod1.m1
```

CHAPTER 3

python json

json

json json json

1. json

```
>>> import json

>>> json_str = '{"name": "lina", "age": 23}'
>>> json_obj = json.loads(json_str)

>>> print json_obj
{u'age': 23, u'name': u'lina'}
```

2. json

```
import json

with open('json_file_path') as fp:
    json_obj = json.load(fp)

# or use loads
json_obj = json.loads(open('json_file_path').read())
```

json

1. json

```
import json

json_obj = {u'age': 23, u'name': u'lina'}
json_str = json.dumps(json_obj)
```

CHAPTER 4

try

python try..except..else try..except else

except else except try except except else try else

CHAPTER 5

shell

grep

sed

awk

- `export SCRIPT_HOME=$(dirname $(readlink -f $0))`

CHAPTER 6

openstack

cloud-init

cloud-init cloud-init

cloud-init

- local: cloud-init-local.service
- init: cloud-init.service
- config: cloud-config.service
- final: cloud-final.service

cloud-init `/etc/cloud/cloud.cfg`

1. local

   cloud-init

   `/usr/bin/cloud-init init --local`
2. init
3. config
4. final

cloud-init

write-mime-multipart cloud-init

```
$ cat my-boothook.txt
#!/bin/sh
echo "Hello World!"
echo "This will run as soon as possible in the boot sequence"

$ cat my-user-script.txt
#!/usr/bin/perl
print "This is a user script (rc.local)\n"

$ cat my-include.txt
# these urls will be read pulled in if they were part of user-data
# comments are allowed.  The format is one url per line
http://www.ubuntu.com/robots.txt
http://www.w3schools.com/html/lastpage.htm

$ cat my-upstart-job.txt
description "a test upstart job"
start on stopped rc RUNLEVEL=[2345]
console output
task
script
echo "====BEGIN======="
echo "HELLO From an Upstart Job"
echo "=====END========"
end script

$ cat my-cloudconfig.txt
#cloud-config
ssh_import_id: [smoser]
apt_sources:
 - source: "ppa:smoser/ppa"

$ write-mime-multipart --output=combined-userdata.txt \
    my-boothook.txt:text/cloud-boothook \
    my-include.txt:text/x-include-url \
    my-upstart-job.txt:text/upstart-job \
    my-user-script.txt:text/x-shellscript \
    my-cloudconfig.txt
```

User Data

1. Gzip Compressed Content
2. Mime Multi Part archive
3. User-Data Script

   `#! Content-Type:  text/x-shellscript`

   python, shell, perl `user data` rc.local
4. Include File

   `#include Content-Type:  text/x-include-url`

   `includ file` url, cloud-init url
5. Cloud Config Data

   `#cloud-config Content-Type:  text/cloud-config`

6. Upstart Job

   `#upstart-job Content-Type: text/upstart-job`

   /etc/init job

7. Cloud Boothook

   `#cloud-boothook Content-Type: text/cloud-boothook`

   `boothook` */var/lib/cloud* `boothook`

8. Part Handler

BFV

ipxe

ipxe pxe

- boot from a web server via HTTP
- boot from an iSCSI SAN
- boot from a Fibre Channel SAN via FCoE
- boot from an AoE SAN
- boot from a wireless network
- boot from a wide-area network
- boot from an Infiniband network
- control the boot process with a script

ipxe

ipxe

- ipxe PXE ROM
- ipxe

ipxe : http://boot.ipxe.org/ipxe.iso ipxe

```
git clone git://git.ipxe.org/ipxe.git
cd ipxe/src
make
```

1. iso

   ```
   make bin/ipxe.iso
   ```

2. usb

```
make bin/ipxe.usb
dd if=bin/ipxe.usb of=/dev/sdX
```

3. pxe rom

```
make bin/undionly.kpxe
```

4. PXE ROM

```
make bin/808610de.rom
```

ironic ipxe

1. http

```
mkdir -p /tftpboot
mkdir -p /httpboot
chown -R ironic /tftpboot
chown -R ironic /httpboot
```

2. tftp map file

```
echo 'r ^([^/]) /tftpboot/\1' > /tftpboot/map-file
echo 'r ^(/tftpboot/) /tftpboot/\2' >> /tftpboot/map-file
```

3. tftp http server

```
[pxe]
tftp_root=/tftpboot
tftp_server=192.168.0.2
ipxe_enabled=True

[deploy]
http_root=/httpboot
http_url=http://192.168.0.2:8080
```

4. http

   apacha nginx nginx

   - httpd

     ```
     $ sudo yum install httpd
     ```

   - httpd

     /etc/httpd/conf/httpd.conf

     ```
     <Directory />
         Options Indexes FollowSymLinks
         AllowOverride none
     </Directory>

     DocumentRoot "/httpboot"
     ```

   -

```
$ sudo systemctl restart httpd
```

5. ipxe

```
[pxe]
pxe_bootfile_name=undionly.kpxe
pxe_config_template = $pybasedir/drivers/modules/ipxe_config.template
```

6. iPXE

```
yum install ipxe-bootimgs
cp /usr/share/ipxe/{undionly.kpxe,ipxe.efi} /tftpboot
```

7. conductor

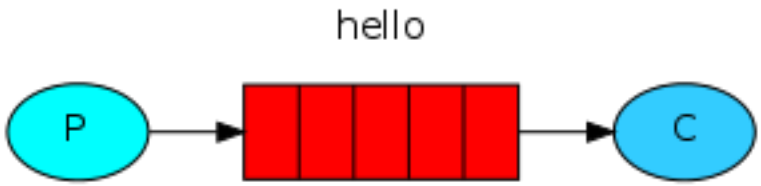
```
sudo systemctl restart openstack-ironic-conductor
```

rabbitmq

rabbitmq

-
-
- rabbitmq

```
hello world
```




```
#!/usr/bin/env python
import pika

connection = pika.BlockingConnection(pika.ConnectionParameters(host='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()
```

```
#!/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()
```

RPC

call

```
cctxt = self.client.prepare(topic=topic or self.topic, version='1.22')
cctxt.call(context, 'do_node_deploy', node_id=node_id,
           rebuild=rebuild, configdrive=configdrive)
```

cast

ironic cloud-init

bug: https://bugs.launchpad.net/ironic/+bug/1656854

commit: https://git.openstack.org/cgit/openstack/nova/commit/?id=28e6faecbf460f08f92d7c25876baf1709633e0c

CHAPTER 7

OS

SUSE

yast

- 163/Update; URL: http://mirrors.163.com/openSUSE/update/13.2/
- 163/Oss; URL: http://mirrors.163.com/openSUSE/distribution/13.2/repo/oss/
- 163/NON-Oss; URL: http://mirrors.163.com/openSUSE/distribution/13.2/repo/non-oss/

CHAPTER 8

SQL

sql

sql *OR AND*

```
$ SELECT * FROM World;
+-----------------+------------+------------+--------------+---------------+
| name            | continent  | area       | population   | gdp           |
+-----------------+------------+------------+--------------+---------------+
| Afghanistan     | Asia       | 652230     | 25500100     | 20343000      |
| Albania         | Europe     | 28748      | 2831741      | 12960000      |
| Algeria         | Africa     | 2381741    | 37100000     | 188681000     |
| Andorra         | Europe     | 468        | 78115        | 3712000       |
| Angola          | Africa     | 1246700    | 20609294     | 100990000     |
+-----------------+------------+------------+--------------+---------------+

$ SELECT name,population,area FROM World WHERE area > 3000000 OR population >␣
↪25000000;
+--------------+-------------+--------------+
| name         | population  | area         |
+--------------+-------------+--------------+
| Afghanistan  | 25500100    | 652230       |
| Algeria      | 37100000    | 2381741      |
+--------------+-------------+--------------+
```

```
$ SELECT * FROM Person;
+----+---------+
| Id | Email   |
```

```
+----+---------+
| 1  | a@b.com |
| 2  | c@d.com |
| 3  | a@b.com |
+----+---------+

$ SELECT Email FROM Person GROUP BY Email having count(Email) > 1;
+---------+
| Email   |
+---------+
| a@b.com |
+---------+
```

CHAPTER 9

English

- redundant:
- jargon:

CHAPTER 10

Indices and tables

- genindex
- modindex
- search