

Aeternity测试网络已经上线许久，这里简单介绍如何在测试网络创建账号，充值，转账，编译智能合约，发布智能合约，调用智能合约的步骤。

## 使用Python SDK创建账号

Aeternity 创建账号可以使用Python SDK，JS SDK或者Go SDK。这里举例Python SDK，下载SDK后执行：

```
>>python aecli
Commands:
  chain      Interact with the blockchain
  config     Print the client configuration
  contract   Compile contracts
  inspect    Get information on transactions, blocks,...
  oracle     Interact with oracles
  wallet     Handle wallet operations1
```

可以看到sdk包含的内容有公链数据查询，合约编译，账户区块查询，预言机，钱包创建等相关内容。我们选择钱包创建，执行：

```
>>python aecli wallet "E://ae//mykey" create
Wallet created
Wallet address_____ ak$2F9bmsy5FRgbs33dMGyStvxyBvoMfpVspmSCaAT7jAog1TDyam
Wallet path_____ E:\ae\mykey
```

 mykey	9/5/2018 10:21 AM	File	1 KB
 mykey.pub	9/5/2018 10:21 AM	Microsoft Office P...	1 KB

在wallet后面加上保存密钥文件的路径，加上create命令，输入密码后可以得到如上结果。

生成的账户其实是一个公钥私钥对，执行以上命令还会弹出输入密码，此密码是对生成的密钥进行加密，所以如果你用编辑器打开mykey只会看到乱码，每次转账的时候都需要输入密码，对私钥解密。

## 对生成的账户进行充值

生成的账户其实是不在链上有记录的，只有执行过交易才会有记录，所以如果发出转账会提示找不到此账户，因此账户的余额为0。[进入测试网充值](#)后，如图所示

## Faucet Aepp

ak\$2F9bmsy5FRgbs33dMGyStvxyBvoMfpVspmSCaAT7jAog1TDyam

Top up!

**Added 250 AET!**

Current Balance: 250

Transaction: [th\\$FBLJ5wn28G2GHU85f1TTK9xWFXJ7xTeyHgbNH53R6pDQS5oEb](#)

默认充值是250AE，充值后就会产生交易记录，可以看到这个交易ID，点击跳转到区块链交易浏览器详情，不过似乎还没开发好，看不到详情。我们使用Python SDK查看交易详情

```
transaction The transaction hash to inspect (eg: th$...)
```

```
E:\workspace\aepp-sdk-python-develop>python aecli inspect transaction th$FBLJ5wn28G2GHU85f1TTK9xWFXJ7xTeyHgbNH53R6pDQS5oEb
Block hash_____ bh$Rs4CgTY82ybN9yw7ya6reHcG13X88cPfr9m4K7u12otZkcbgZ
Block height_____ 4074
Signatures_____ sg$MkEGKBj1fQrN2VpJFW5bFBhp2K9uCqX9pc2tWriVVncfDvNcwRgSNcZiDkDeWa1BG5iDzHJMFc6AgsKtNPD1rRaQuMBfi
Sender account_____ ak$2alJ2Mk9YSmC1gioUq4PWRm3bsv887MbuRVwyv4KaUGoR1eiKi
Recipient account_____ ak$2F9bmsy5FRgbs33dMGyStvxyBvoMfpVspmSCaAT7jAog1TDyam
Amount_____ 250
TTL_____ 4124
```

## 测试网转账

我们的账户已经可以看到有余额是250AE了，下面对此账户进行转账

```
E:\workspace\aepp-sdk-python-develop>python aecli wallet "E://ae//mykey" spend ak$2nsbEHGpZDs2vhuJnbZDCKsFY1k4qyH18TongQqgWwHCwJaloS 1
=====dd
=====
Enter the wallet password []:
=====
Transaction posted to the chain
Transaction hash_____ th$KKBdgFpJu41jK4XULpMrpo78yJzqpAtdSrrZuesRcKYqn9ThX
Sender account_____ ak$2F9bmsy5FRgbs33dMGyStvxyBvoMfpVspmSCaAT7jAog1TDyam
Recipient account_____ ak$2nsbEHGpZDs2vhuJnbZDCKsFY1k4qyH18TongQqgWwHCwJaloS
```

忽略掉我的打印输入哈，指定了我们的密钥路径后，接上spend命令，转账到哪个账户的地址，转账余额。会提示输入密码，此密码就是解密我们密钥的密码输出上图，我们再看一下我们的账户余额

```
E:\workspace\aepp-sdk-python-develop>python aecli inspect account ak$2F9bmsy5FRgbs33dMGyStvxyBvoMfpVspmSCaAT7jAog1TDyam
Account balance_____ 248
```

经过我多次转账测试，发现每次转账测试网的手续费收取1AE，这个应该会调整

## 智能合约编译

Python SDK也提供了编译部署智能合约的功能，不过这里介绍[在线IDE](#)进行编译，合约使用的是一个叫Sophia 的语言。

Modify Settings

## Test contracts (client connected to <https://sdk-testnet.aepps.com>)

**Sophia Contract's Code:**

```
1 contract Identity =
2   type state = ()
3   function main(x : int) = x
```

**Byte Code**  
Deployed, and mined at this address:  
[ct\\$PzpYsHcA858UNDJLnfmQznnNKPwxmatXwFrWG6GjNhr1rScgm](#)  

Function: init

Arguments: 0

Deposit	Gas Price	Amount	Fee	Gas Limit
1	1	1	1	40000000

Deploy

**← Call Static Function**  

Function: main

Arguments: (1)

Call Static

直接使用测试样例，点击编译后成字节码，再点击下面的部署按钮就可以部署到测试网络了。

**Byte Code**  
Deployed, and mined at this address:  
[ct\\$PzpYsHcA858UNDJLnfmQznnNKPwxmatXwFrWG6GjNhr1rScgm](#)  

Function: init

Arguments: 0

Deposit	Gas Price	Amount	Fee	Gas Limit
1	1	1	1	40000000

Deploy

**← Call Static Function**  

Function: main

Arguments: (1)

Call Static

**↑ Call Function**  

Deposit	Gas Price	Amount	Fee	Gas Limit
1	1	1	1	40000000

Function: function

Arguments: arguments

Return Type: int

Call Function

点击部署后会生成一个合约地址，我这里的合约地址为

ct\$PzpYsHcA858UNDJLnfmQznnNKPwxmatXwFrWG6GjNhr1rScgm

使用区块链浏览器进行查看合约和交易

Dashboard

Generations

average rate 1 per

15s

last key block mined

1m 33sago

Generations					
4194	key-hash <b>bh\$CSvQy_fKpay3</b>	6Micro Blocks	6Transaction(s)	mined by <b>ak\$2TgWw_9uSwSj</b>	time 1m 33s
4193	key-hash <b>bh\$2NtBe_TKtquS</b>	1Micro Blocks	1Transaction(s)	mined by <b>ak\$2TgWw_9uSwSj</b>	time 1m 42s
4192	key-hash <b>bh\$BbpTM_WRzXpy</b>	3Micro Blocks	3Transaction(s)	mined by <b>ak\$2jirk_1cn5ew</b>	time 2m 03s
4191	key-hash <b>bh\$25Me8_wEyo6h</b>	0Micro Blocks	0Transaction(s)	mined by <b>ak\$2jirk_1cn5ew</b>	time 2m 06s
4190	key-hash <b>bh\$LB3KT_KwQKxp</b>	2Micro Blocks	2Transaction(s)	mined by <b>ak\$2jirk_1cn5ew</b>	time 2m 18s
4189	key-hash <b>bh\$UaudM_MCLVrK</b>	2Micro Blocks	2Transaction(s)	mined by <b>ak\$BpwWt_JTnPdn</b>	time 2m 30s
4188	key-hash <b>bh\$dkZij_CfQJ8K</b>	9Micro Blocks	8Transaction(s)	mined by <b>ak\$2jirk_1cn5ew</b>	time 3m 25s
4187	key-hash <b>bh\$WVsoD_uikCby</b>	0Micro Blocks	0Transaction(s)	mined by <b>ak\$2jirk_1cn5ew</b>	time 3m 26s
4186	key-hash <b>bh\$CzwjG_gdn4zU</b>	1Micro Blocks	1Transaction(s)	mined by <b>ak\$2jirk_1cn5ew</b>	time 3m 34s
4185	key-hash <b>bh\$21zRm_ke6bBf</b>	2Micro Blocks	2Transaction(s)	mined by <b>ak\$BpwWt_JTnPdn</b>	time 3m 52s

此浏览器可以查看我们以上执行动作的过程，包括充值交易，转账交易，合约创建。SDK并没有提供查询合约的方法，浏览器也不是很完善，所以刚才创建的合约我就不去找了。

## 智能合约调用

1 Call Function

Deposit

Gas Price

Amount

Fee

Gas Limit

1

1

1

1

40000000

Function

Arguments

Return Type

main

1

int

Call Result

Gas Used: 524

---

Result:

1

Call Function

这里使用IDE调用，传入方法名字和参数就可以了，更加复杂的合约例子可以看github的例子。

## 在线IDE更改账户

Settings

Host

Private Key

Public Key

https://sdk-testnet.aepps.com

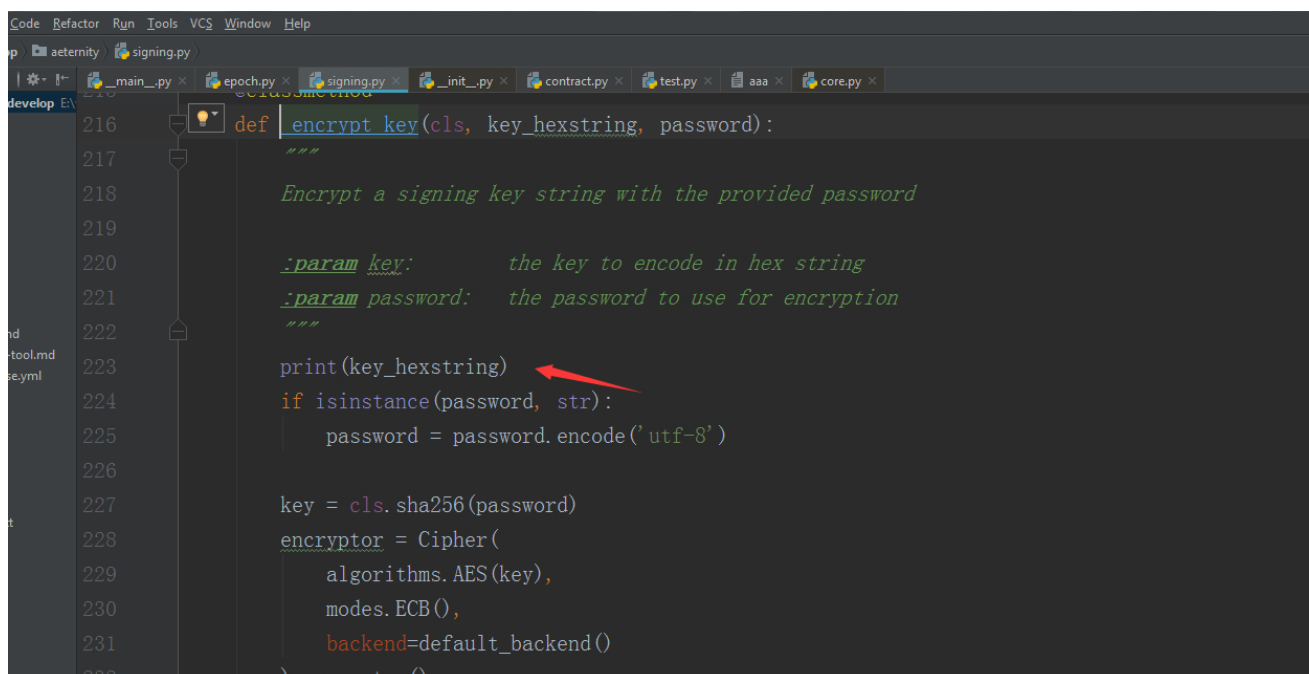
1749dd02ba1238d801b58f353e5e5b338191ecd23fz

ak\$2nsbEHGpZDs2vhuJnbZDCKsFY1k4qyH18TongC

Save

## Test contracts (client connected to https://sdk-testnet.aepps.com)

如果需要更改账户的话，在线IDE的头部可以输入你的账户的密钥，此外，这个密钥是一个十六进制的字符串，用Python SDK生成的是一个已经加密的的密钥文件，所以我们可以



```
216 def encrypt_key(cls, key_hexstring, password):
217     """
218     Encrypt a signing key string with the provided password
219
220     :param key: the key to encode in hex string
221     :param password: the password to use for encryption
222     """
223     print(key_hexstring)
224     if isinstance(password, str):
225         password = password.encode('utf-8')
226
227     key = cls.sha256(password)
228     encryptor = Cipher(
229         algorithms.AES(key),
230         modes.ECB(),
231         backend=default_backend()
232     )
233     encryptor =
```

如上图的位置增加一个输出，得到真正的私钥，后保存下来使用。

## 还需要改进的地方

- 区块链浏览器 不能根据交易ID查询，根据合约ID查询，交易详情不够完善，比如gas的使用量，交易费用等。
- 和以太坊IDE相比的话，缺少Debugger功能，还不够完善
- Python SDK缺少查询合约的功能

## 相关地址

python sdk地址: <https://github.com/aeternity/aepp-sdk-python>

测试网充值地址: <https://faucet.aepps.com/>

在线合约IDE: <https://contracts.aepps.com/>