Chapter 6 Transaction

Transaction on browser(1/5)

Transaction View information about a bitcoin transaction

0627052b6f28912f2703066a912ea577f2ce4da4caa5a5fbd8a57286c345c2f2

1Cdid9KFAaatwczBwBttQcwXYCpvK8h7FK (0.1 BTC - Output)

1GdK9UzpHBzqzX2A9JFP3Di4weBwqgmoQA - (Unspent) 0.015 BTC

1Cdid9KFAaatwczBwBttQcwXYCpvK8h7FK - (Unspent) 0.0845 BTC

97 Confirmations | 0.0995 BTC

Summary	
Size	258 (bytes)
Received Time	2013-12-27 23:03:05
Included In Blocks	277316 (2013-12-27 23:11:54 +9 minutes)

Inputs and Outputs	
Total Input	0.1 BTC
Total Output	0.0995 BTC
Fees	0.0005 BTC
Estimated BTC Transacted	0.015 BTC

Transaction on browser(2/5)

Summary

Size	258 (bytes)
Fee Rate	0.001937984496124031 BTC per kB
Received Time	Dec 28, 2013 7:11:54 AM
Mined Time	Dec 28, 2013 7:11:54 AM
Included in Block	0000000000000001b6b9a13b095e96db41c4a928b97ef2d944a9b31b2cc7bdc4

Details

0627052b6f28912f2703066a912ea577f2ce4da4caa5a5fbd8a57286c345c2f2 mined Dec 28, 2013 7:11:54 AM

1Cdid9KFAaatwczBwBttQcwXYCpvK8h7FK	0.1 BTC	1GdK9UzpHBzqzX2A9JFP3Di4weBwqgmoQA	0.015 BTC (U)
		1Cdid9KFAaatwczBwBttQcwXYCpvK8h7FK	0.0845 BTC (U)

FEE: 0.0005 BTC

259642 CONFIRMATIONS 0.0995 BTC

Transaction on browser(3/5)

Transaction View information about a bitcoin transaction

0627052b6f28912f2703066a912ea577f2ce4da4caa5a5fbd8a57286c345c2f2

1Cdid9KFAaatwczBwBttQcwXYCpvK8h7FK ($ 628.66 - Output)	→	1GdK9UzpHBzqzX2A9JFP3Di4weBwqgmoQA - (Unspent) $ 94.30 1Cdid9KFAaatwczBwBttQcwXYCpvK8h7FK - (Unspent) $ 531.22 $ 625.52

Summary	
Size	258 (bytes)
Weight	1032
Received Time	2013-12-27 23:03:05
Included In Blocks	277316 (2013-12-27 23:11:54 + 9 minutes)
Confirmations	259642
Visualize	View Tree Chart

Inputs and Outputs	
Total Input	$ 628.66
Total Output	$ 625.52
Fees	$ 3.14
Fee per byte	193.798 sat/B
Fee per weight unit	48.45 sat/WU
Estimated BTC Transacted	$ 94.30
Scripts	Hide scripts & coinbase

Transaction on browser(4/5)

Transaction on browser(5/5)

Summary

Height	277316	Input	0.10000000 BTC
Confirmations	259643	Output	0.09950000 BTC
Timestamp	2013-12-28 07:11:54	Sigops	8
Size (rawtx)	258 Bytes	Fees	0.00050000 BTC
Virtual Size	258 Bytes	Fees Rate (BTC / kVB)	0.00193798 BTC
Weight	1,032		

Input (1)	0.10000000 BTC	Output (2)	0.09950000 BTC
1Cdid9KFAaatwczBwBttQcwXYCpvK8h7FK	0.10000000	1GdK9UzpHBzqzX2A9JFP3Di4weBwqgmoQA	0.01500000
		1Cdid9KFAaatwczBwBttQcwXYCpvK8h7FK	0.08450000

259,643 Confirmations

Bitcoind-cli command

./bitcoin-cli getrawtransaction "4c0de15…2c3c25c0“

>>>0100…0000(Hex)

./bitcoin-cli decoderawtransaction 0100…0000

>>> "txid": "4c0de1532f0cebb80ebb3a4e787bcd7f512789ab70109307195e697b2c3c25c0",


```
"hash": "3a522ffc32a4d7c845dea08ab38fdc017eff9329febee131036ce6f71d7d6640",
"version": 1,
"size": 216,
"vsize": 134,
"locktime": 0,
"vin": [
 {
  "txid": "da17232cfaa1da9edda2e3aded4d7ffc41b1c8976d2b364bc5c1064148ae00ee",
  "vout": 1,
  "scriptSig": {
   "asm": "0014646abc1ac4bb8b780bc0c7e316629dfc18b8cfea",
   "hex": "160014646abc1ac4bb8b780bc0c7e316629dfc18b8cfea"
  },
  "txinwitness": [
   "3045022100ef58e75476f619b6cb4b60588a5d25aab54c29677749c3c33ef4928f0fe6f8c0022060a558b713232917ae623c84e9e615d8848c36abb272ce61df363351ff9ffef701",
   "0371104ae352bba2855cc0f0f88f27044b0795e65d01c18b32c7debf8bba09cc62"
  ],
  "sequence": 4294967295
 }
],
"vout": [
 {
  "value": 0.00645280,
  "n": 0,
  "scriptPubKey": {
   "asm": "OP_HASH160 4d3a3cc3b216b19bd8e6a9d16086b8620826ead9 OP_EQUAL",
   "hex": "a9144d3a3cc3b216b19bd8e6a9d16086b8620826ead987",
   "reqSigs": 1,
   "type": "scripthash",
   "addresses": [
    "38jMiiZs2C5n5MPkyc5pSA7wwW6H4p6hPa"
   ]
  }
 }
]
}
```

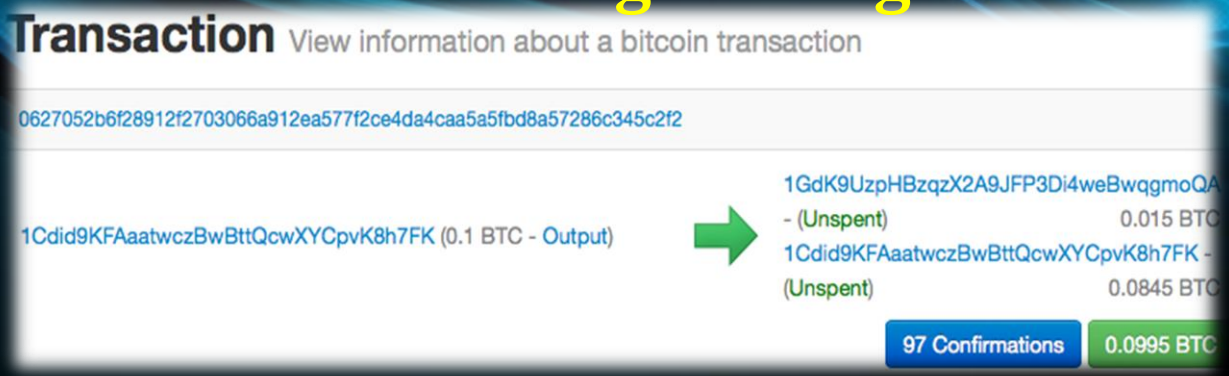
Transaction encoder


```
"vin":
[
  {
    "txid": "7957a35fe64f80d234d76d83a2a8f1a0d8149a41d81de548f0a65a8a999f6f18",
    "vout": 0,
    "scriptSig" :
"3045022100884d142d86652a3f47ba4746ec719bbfbd040a570b1deccbb6498c75c4ae24cb02204b9f039ff08df09cbe9f6addac960298cad530a863ea8f53982c09db8f6e381301410484ecc0d46f1918b30928fa0e4ed99f16a0fb4fde0735e7ade8416ab9fe423cc5412336376789d172787ec3457eee41c04f4938de5cc17b4a10fa336a8d752adf",
    "sequence": 4294967295
  }
]
"vout": [
  {
    "value": 0.01500000
    “scriptPubKey”: “OP_DUP OP_HASH160 ab68025513c3dbd2f7b92a94e0581f5d50f654e7 OP_EQUALVERIFY OP_CHECKSIG"
  },
  {
    "value": 0.08450000,
    “scriptPubKey”: “OP_DUP OP_HASH160  7f9b1a7fb68d60c536c2fd8aeaa53a8f3cc025a8  OP_EQUALVERIFY OP_CHECKSIG"
  },
]
```



```
01000000011 86f9f998a5aa6f048e51dd8419a14d8a0f1 a8a2836dd73
4d2804fe65fa35779000000008b483045022100884d142d86652a3f47
ba4746ec719bbfbd040a570b1 deccbb6498c75c4ae24cb02204b9f039
ff08df09cbe9f6addac960298cad530a863ea8f53982c09db8f6e3813
01410484ecc0d46f1918b30928fa0e4ed99f16a0fb4fde0735e7ade84
16ab9fe423cc5412336376789d172787ec3457eee41 c04f4938de5cc1
7b4a10fa336a8d752adffffffff0260e3160000000000001976a914ab6
8025513c3dbd2f7b92a94e0581f5d50f654e788acd0ef800000000000
1976a9147f9b1a7fb68d60c536c2fd8aeaa53a8f3cc025a888ac00000000
```

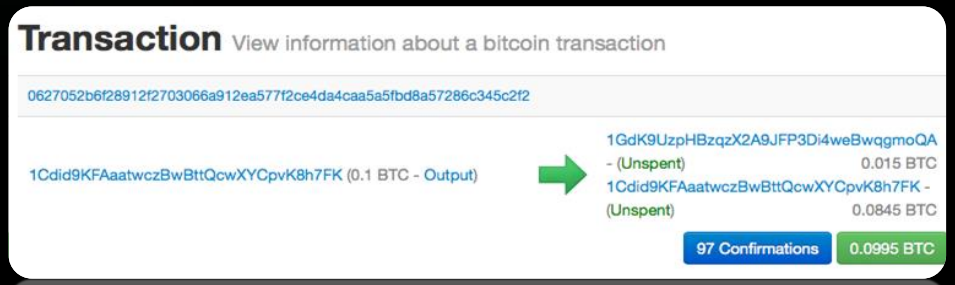
Something missing?




```
{
 "txid": "0627052b6f28912f2703066a912ea577f2ce4da4caa5a5fbd8a57286c345c2f2",
 "version": 1,
 "locktime": 0,
 "vin": [
  {
   "txid":"7957a35fe64f80d234d76d83a2a8f1a0d8149a41d81de548f0a65a8a999f6f18",
   "vout": 0,
   "scriptSig":
"3045022100884d142d86652a3f47ba4746ec719bbfbd040a570b1deccbb6498c75c4ae24cb02204b9f039ff08df09cbe9f6addac960298cad530a8
63ea8f53982c09db8f6e38130141 0484ecc0d46f1918b30928fa0e4ed99f16a0fb4fde0735e7ade8416ab9fe423cc5412336376789d172787ec3457
eee41c04f4938de5cc17b4a10fa336a8d752adf",
   "sequence": 4294967295
  }
 ],
 "vout": [
  {
   "value": 0.01500000,
   "scriptPubKey": "OP_DUP OP_HASH160 ab68025513c3dbd2f7b92a94e0581f5d50f654e7 OP_EQUALVERIFY OP_CHECKSIG"
   (Hash160) 1GdK9UzpHBzqzX2A9JFP3Di4weBwqgmoQA
  },
  {
   "value": 0.08450000,
   "scriptPubKey": "OP_DUP OP_HASH160 7f9b1a7fb68d60c536c2fd8aeaa53a8f3cc025a8 OP_EQUALVERIFY OP_CHECKSIG",
  }
 ]
}
```

Json structure

{
"txid": "0627052b6f28912f2703066a912ea577f2ce4da4caa5a5fbd8a57286c345c2f2",
"version": 1,
"locktime": 0,(Chapter7)
"vin": [
{
“txid”:“7957a35fe64f80d234d76d83a2a8f1a0d8149a41d81de548f0a65a8a999f6f18”, (hash of transaction)
“vout”: 0, (標示來自前次交易的第n筆輸出UTXO被引用)
“scriptSig”:
“3045022100884d142d86652a3f47ba4746ec719bbfbd040a570b1deccbb6498c75c4ae24cb02204b9f039ff08df09cbe9f6addac960298cad530a863ea8f53982c09db8f6e381301400484ecc0d46f1918b30928fa0e4ed99f16a0fb4fde0735e7ade8416ab9fe423cc5412336376789d172787ec3457eee41c04f4938de5cc17b4a10fa336a8d752adf”, (滿足放置在UTXO上面的解鎖條件也就是簽名)(由Alice錢包創建)
"sequence": 4294967295
}
{
“txid”(如果引用的UTXO不只一個會在此繼續引用)
“vout”: 100
}
],
"vout": [
{
“value”: 0.01500000,(第一筆輸出由Alice至Bob)
“scriptPubKey”: “OP_DUP OP_HASH160 ab68025513c3dbd2f7b92a94e0581f5d50f654e7 OP_EQUALVERIFY OP_CHECKSIG“ (該UTXO解鎖腳本)
},
{
“value”: 0.08450000,(第二筆輸出由Alice至Alice錢包)
“scriptPubKey”: “OP_DUP OP_HASH160 7f9b1a7fb68d60c536c2fd8aeaa53a8f3cc025a8 OP_EQUALVERIFY OP_CHECKSIG”, (該UTXO解鎖腳本)
}
]
}



UTXO intruduction

- $1BTC = 10^8 satoshi$
- A transaction output can have an arbitrary (integer) value denominated as a multiple of satoshis.
- Although an output can have any arbitrary value, once created it is indivisible.
- An unspent output can only be consumed in its entirety by a transaction.

Scenario of transaction(1/3)

Vout:96

Vout:47

Vout:3

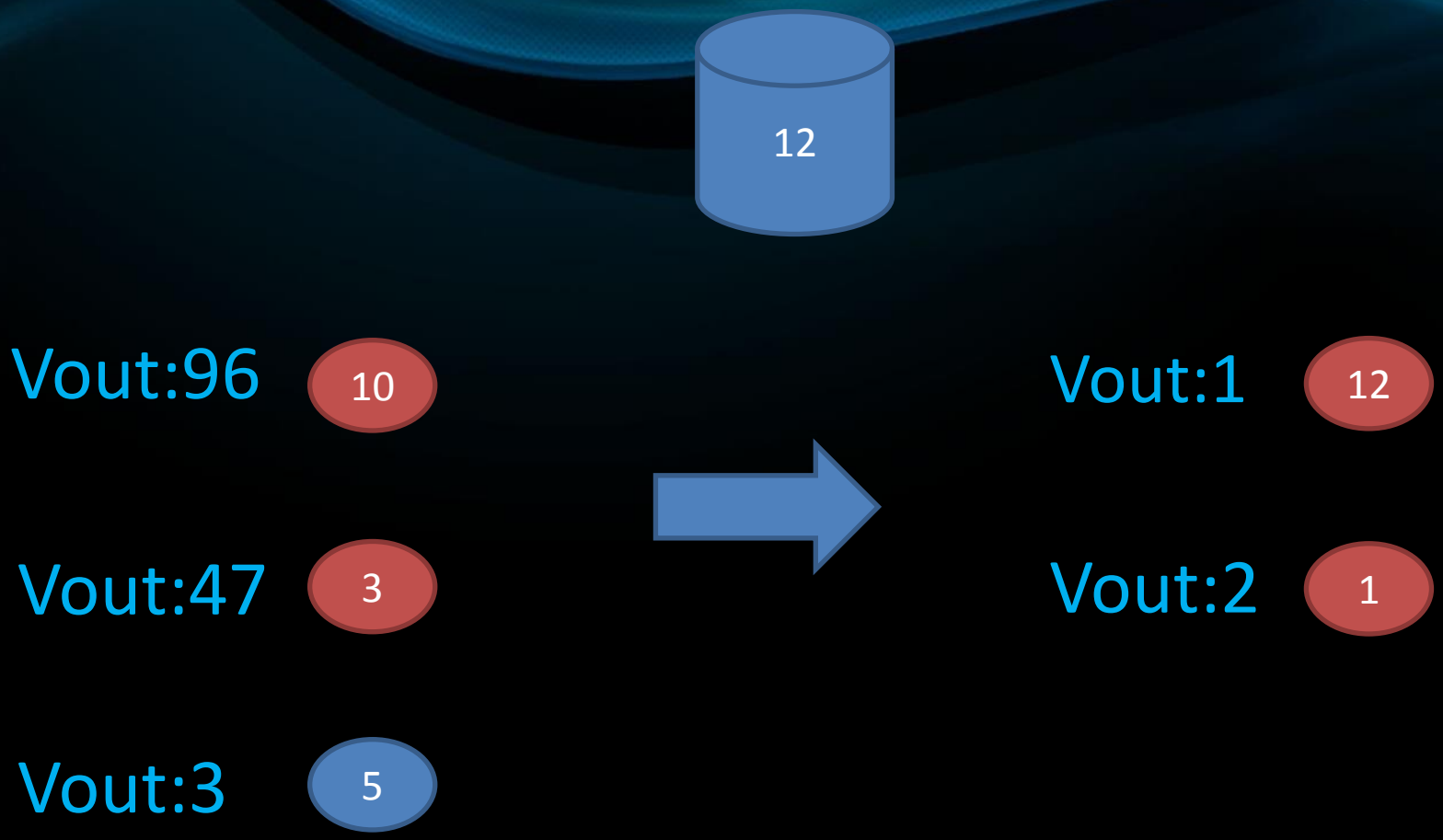
Scenario of transaction(2/3)

Vout:96

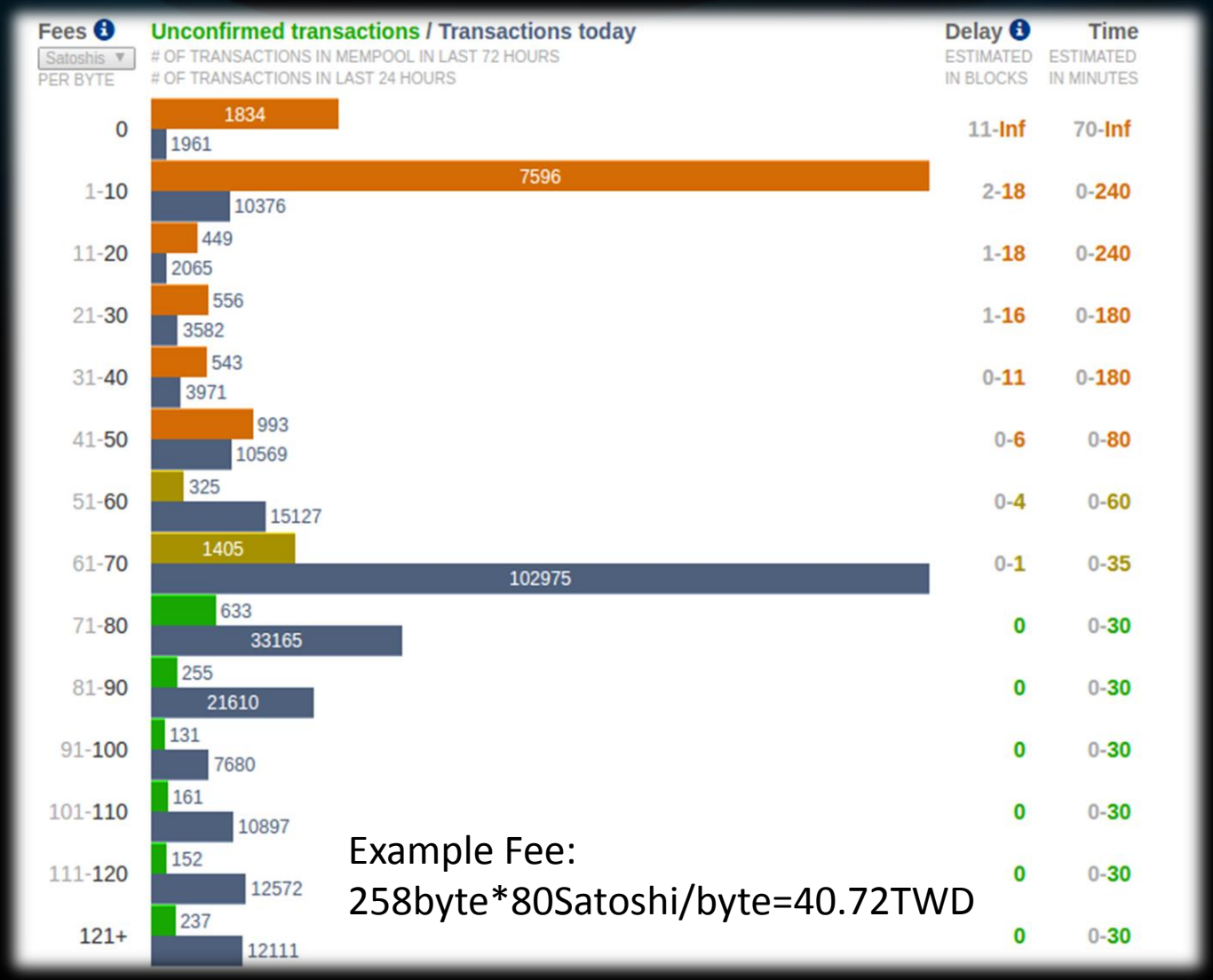
Vout:47

Vout:3

Scenario of transaction(3/3)



Transaction Fee



Example Fee:
258byte*80Satoshi/byte=40.72TWD

Verify process

Unlock script
<Cafe Signature> <Cafe Public Key>

P2PKH :pay to pub key hash

Lock script
OP_DUP OP_HASH160 <Cafe Public Key Hash> OP_EQUALVERIFY OP_CHECKSIG

input

1.Unlock script
2.引用UTXO(Lock script)

True of False

定出花費該UTXO的條件
公鑰
地址

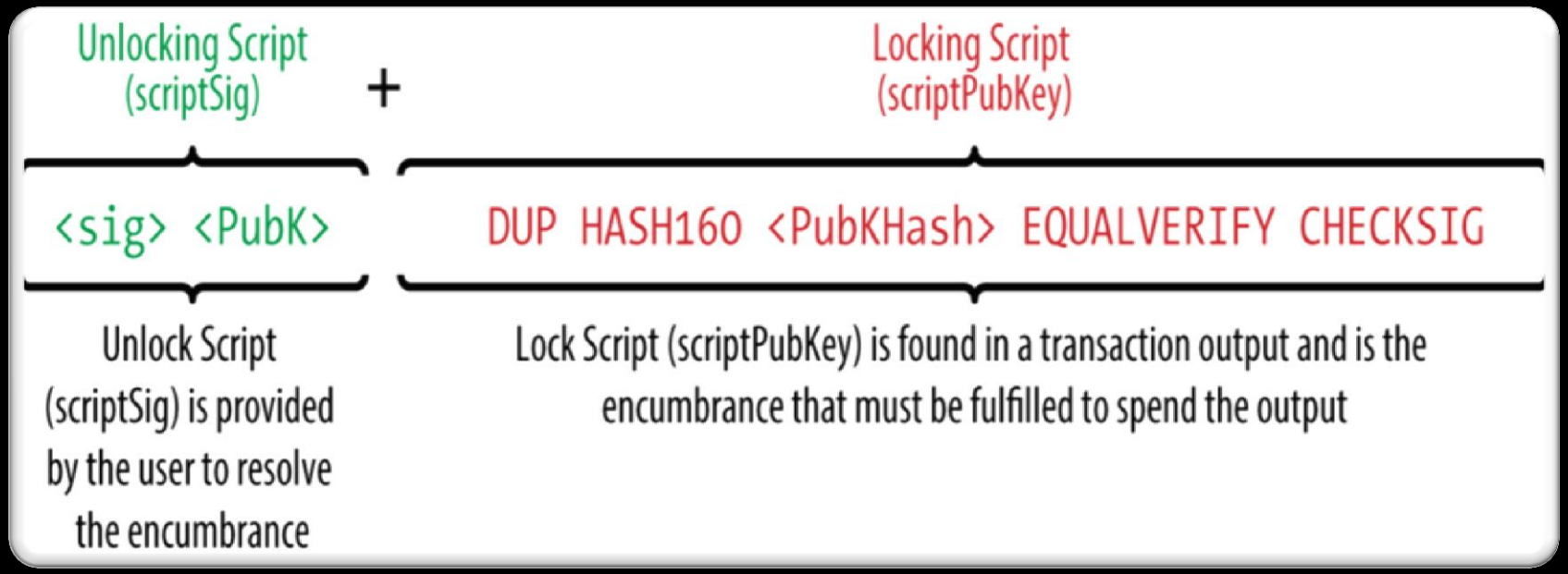
數位簽章(由私鑰產生)
ScriptSig

Concat

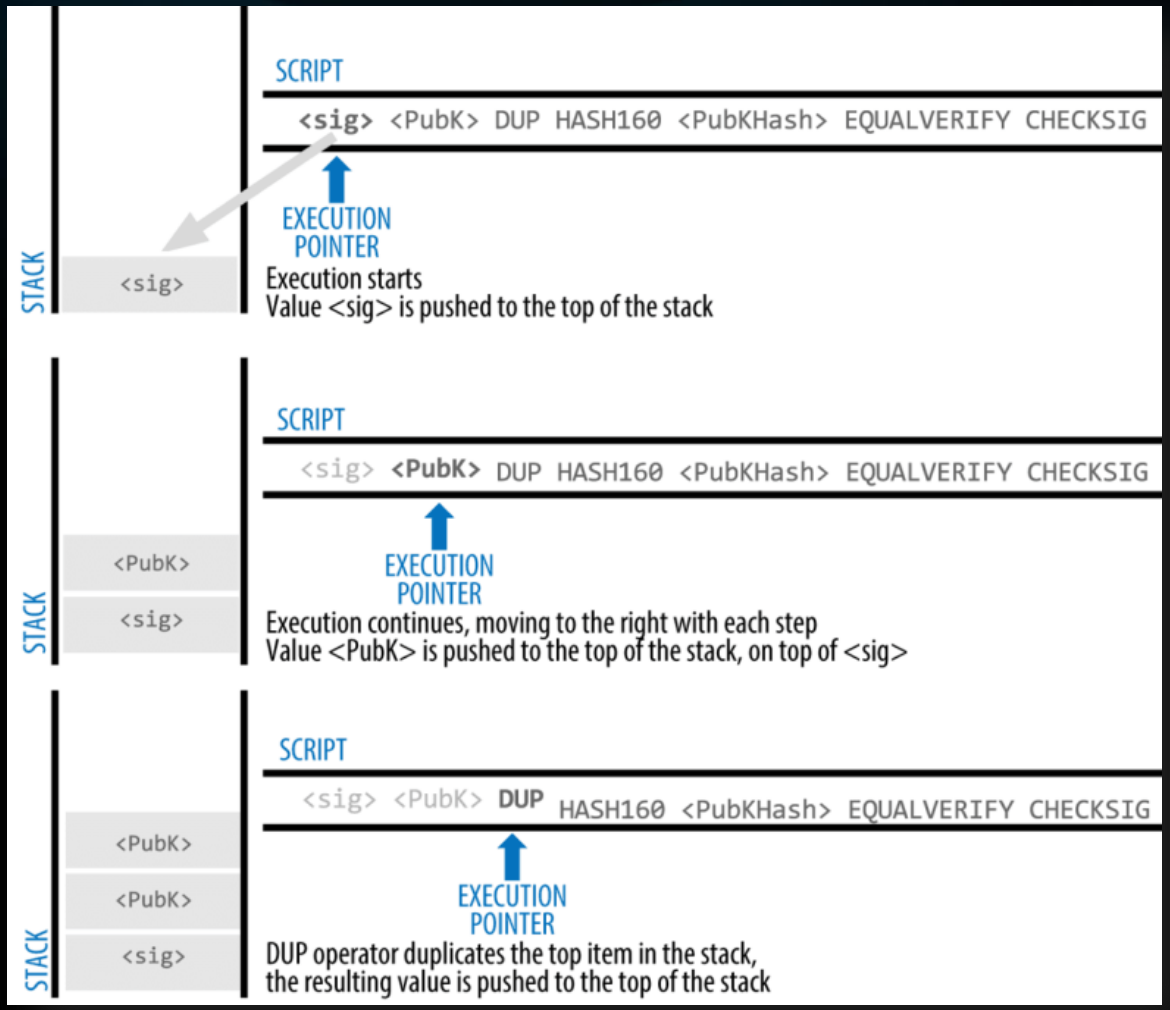
Unlock script
<Cafe Signature> <Cafe Public Key>

P2PKH :pay to pub key hash

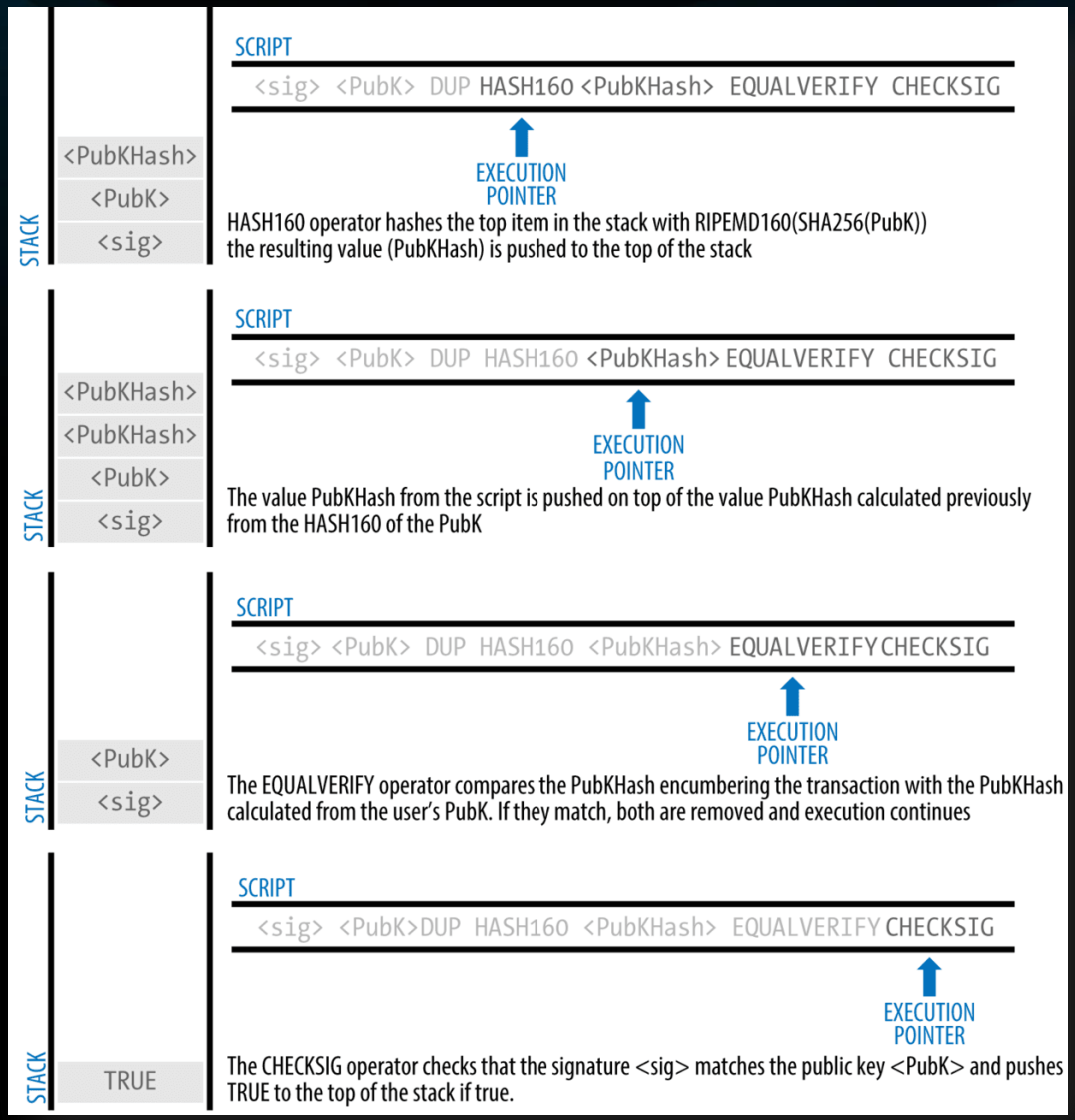
Lock script
OP_DUP OP_HASH160 <Cafe Public Key Hash> OP_EQUALVERIFY OP_CHECKSIG



Stack process(1/2)



Stack process(2/2)



假設世界上所有人都會乘法，沒有人會除法

有天Alice挑出了兩個數字123,456

由於Alice會乘法，於是計算出123x456=56088，並告訴Bob：123x???=50688

Bob想告訴Alice一個秘密67但不想讓別人知道，於是Bob自己先計算

123x222=27306

56088x222+67=12451603

Bob再告訴Alice

123x???=27306

56088x???+x=12451603

Alice可利用已知訊息(123x456=56088)計算出x

假設世界上所有人都會乘法，沒有人會除法

有天Alice挑出了兩個數字123(G),456(dA(Alice私鑰))

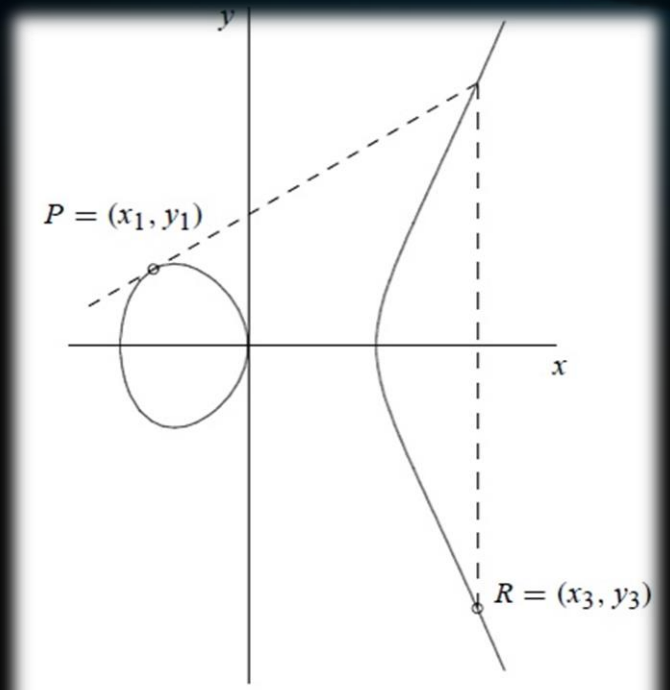
由於Alice會乘法，於是計算出123(G)x456(dA(Alice私鑰))=56088，並告訴Bob：123(G)x???=50688(Alice公鑰)

Bob想告訴Alice一個秘密67(Hash(交易訊息))但不想讓別人知道，於是Bob自己先計算
123(G)x222(Bob創的臨時私鑰)=27306(Bob臨時公鑰)
56088(Alice公鑰)x222(Bob創的臨時私鑰)+67(Hash(交易訊息))=12451603

Bob再告訴Alice
123x???=27306
56088x???+x=12451603

Alice可利用已知訊息(123x456=56088)計算出x

Why we use ECC



(b) Doubling: $P + P = R$.

- 取代原本加法運算，如果使用普通的加法運算，多次加法後(等同乘法)使用除法即可得私鑰
- 使用橢圓曲線加法運算，在多次加法後(等同乘法)不存在除法運算，僅能使用窮舉法取得私鑰
- 此橢圓曲線加法運算有封閉性，使得在求公鑰時運算難度為O(logn)，降低運算量

secp256k1

- 在secp256k1共識下規定(p,a,b,G,n,h)六個參數
- 橢圓曲線方程式為$y^2 = x^3 + ax + b$

(P為加解密時取的mod)

$P = 2^{256} - 2^{32} - 2^9 - 2^8 - 2^7 - 2^6 - 2^4 - 1$

$= FFFFFFFF\ FFFFFFFF\ FFFFFFFF\ FFFFFFFF\ FFFFFFFF\ FFFFFFFF\ FFFFFFFE\ FFFFFC2F$

(a為橢圓曲線係數)

$a = 0$

(b為橢圓曲線係數)

$b = 7$

(G為橢圓曲線進行加法運算時起點)

$G = 04\ 79BE667E\ F9DCBBAC\ 55A06295\ CE870B07\ 029BFCDB\ 2DCE28D9\ 59F2815B\ 16F81798$
$483ADA77\ 26A3C465\ 5DA4FBFC\ 0E1108A8\ FD17B448\ A6855419\ 9C47D08F\ FB10D4B8$

(n為使nG=0的最小正整數，隨機創建的私鑰必須小於此值)

$n = FFFFFFFF\ FFFFFFFF\ FFFFFFFF\ FFFFFFFE\ BAAEDCE6\ AF48A03B\ BFD25E8C\ D0364141$

(*h*待補充，目前不知道用途)

h= 01

Alice Sig

Unlock script
<Cafe Signature> <Cafe Public Key>

Lock script
OP_DUP OP_HASH160 <Cafe Public Key Hash> OP_EQUALVERIFY OP_CHECKSIG

簽名的驗證意味著只有生成此公鑰的私鑰的所有者，才能在此交易上產生此簽名。

Alice想要為交易簽名必須擁有以下資料：

1. 交易內容(m)
2. Alice私鑰(dA)
3. 創建臨時私鑰(k)(每次為新交易簽名時，需額外生新私鑰)
4. 使用臨時私鑰使用橢圓曲線計算臨時公鑰的x座標(R)
5. 計算$S = k^{-1}(Hash(m) + dA * R) \bmod p$
6. 將R,S以DER編碼，即簽名

Note：臨時私鑰用來生成臨時公鑰後，即可丟棄，臨時私鑰不會在網路上傳播

DER

Unlock script
<Cafe Signature> <Cafe Public Key>

Lock script
OP_DUP OP_HASH160 <Cafe Public Key Hash> OP_EQUALVERIFY OP_CHECKSIG

*Distinguished Encoding Rules(*DER)編碼規則

3045022100884d142d86652a3f47ba4746ec719bbfbd040a570b1deccbb6498c75c4ae24cb02204b9f039ff08df09cbe9f6addac960298cad530a863ea8f53982c09db8f6e3813

0x30 - 表示DER序列的開始
0x45 - 序列的長度（69位元組）整個序列的長度 (1+1+33+1+1+32=69)
0x02 - 序列是一個整數值
0x21 - 整數的長度（33位元組）
R-00884d142d86652a3f47ba4746ec719bbfbd040a570b1deccbb6498c75c4ae24cb
0x02 - 序列是一個整數值
0x20 - 整數的長度（32位元組）
S-4b9f039ff08df09cbe9f6addac960298cad530a863ea8f53982c09db8f6e3813

Bob verify

Unlock script
<Cafe Signature> <Cafe Public Key>

Lock script
OP_DUP OP_HASH160 <Cafe Public Key Hash> OP_EQUALVERIFY OP_CHECKSIG

Bob想要驗證簽名必須擁有以下資料：

1. 交易內容(m)(已知)
2. 簽名R,S值(經由DER解碼得知)
3. 橢圓曲線起點G(secp256k1定義為定值)
4. 計算$P = S^{-1} * Z * G + S^{-1} * R * Qa$
5. $P.x == R$

Signature and Verification

$$S = k^{-1}(Hash(m) + dA * R)\ mod\ p$$

*k*是臨時私鑰
*R*是臨時公鑰的x座標
*dA*是簽名私鑰
*m*是簽署的交易資料
*p*是橢圓曲線邊界限制

$$\mathrm{P} = S^{-1} * Hash(m) * G + S^{-1} * R * Qa$$

*R*是臨時公鑰的x座標
S是簽名值由第一式計算得出
*Qa*是Alice的公鑰
*m*是簽署的交易資料
*G*是橢圓曲線發生器點

Proof of equivalent

$$Z = Hash(m)$$
$$S = k^{-1}(Z + dA * R)\ mod\ p$$
$$P = S^{-1} * Z * G + S^{-1} * R * Qa$$
$$= S^{-1}(Z * G + R * Qa)$$
$$= \frac{Z * G + R * Qa}{k^{-1}(Z + dA * R)}, \text{where } Qa = dA * G$$
$$= \frac{Z * G + R * dA * G}{k^{-1}(Z + dA * R)}$$
$$= \frac{G(Z + R * dA)}{k^{-1}(Z + R * dA)}$$
$$= kG$$

判斷P(x)==R

k:臨時私鑰
R：臨時公鑰的x座標、簽名值
S：簽名值
dA：是簽名私鑰
m：交易資料
p：邊界
Qa：Alice的公鑰
G：橢圓曲線起點

Ephemeral (temporary) private public key pair

$$\begin{cases} S_1 = k^{-1}(Z_1 + dA * R) \\ S_2 = k^{-1}(Z_2 + dA * R) \end{cases}$$

R=k*G

G：常數
k：臨時私鑰
R：定值(臨時公鑰x座標)

Historical perspective(1/2)

Transaction View information about a bitcoin transaction

0627052b6f28912f2703066a912ea577f2ce4da4caa5a5fbd8a57286c345c2f2

1Cdid9KFAaatwczBwBttQcwXYCpvK8h7FK (0.1 BTC - Output)

1GdK9UzpHBzqzX2A9JFP3Di4weBwqgmoQA - (Unspent) 0.015 BTC
1Cdid9KFAaatwczBwBttQcwXYCpvK8h7FK - (Unspent) 0.0845 BTC

97 Confirmations | 0.0995 BTC

Summary	
Size	258 (bytes)
Received Time	2013-12-27 23:03:05
Included In Blocks	277316 (2013-12-27 23:11:54 +9 minutes)

Inputs and Outputs	
Total Input	0.1 BTC
Total Output	0.0995 BTC
Fees	0.0005 BTC
Estimated BTC Transacted	0.015 BTC

1. 發送者地址(左側)並不存在於鏈上，瀏覽器必須尋找該UTXO在上一筆交易中的輸出。
2. 在該輸出內是一個Locking Script，將UTXO鎖定到Alice的公鑰雜湊(P2PKH腳本)。
3. 將提取出的公鑰雜湊使用Base58Check編碼，以生成地址。

1. 發送者地址(右側)並不存在於鏈上，瀏覽器必須從每個輸出中提取鎖定腳本。
2. 將鎖定腳本識別為P2PKH腳本，從內部提取公鑰雜湊。
3. 將提取出的公鑰雜湊使用Base58Check編碼，以生成收件地址。

Historical perspective(2/2)

Bitcoin Address Addresses are identifiers which you use to send bitcoins to another person.

Summary	
Address	1GdK9UzpHBzqzX2A9JFP3Di4weBwqgmoQA
Hash 160	ab68025513c3dbd2f7b92a94e0581f5d50f654e7
Tools	Taint Analysis - Related Tags - Unspent Outputs

Transactions	
No. Transactions	25
Total Received	0.17579525 BTC
Final Balance	0.17579525 BTC

1. 區塊鏈中沒有餘額的概念。
2. 瀏覽器首先解碼比特幣地址的公鑰雜湊(160bit)
3. 瀏覽器搜索交易資料庫，找出使用此公鑰雜湊的P2PKH鎖定腳本
4. 總結所有輸出的值，瀏覽器可以產生接收的值
5. 同時瀏覽器還需要統計被花費的UTXO
6. 才能依此統計餘額
7. 如果瀏覽器未能同步，餘額可能會有錯誤

Transaction

$ bitcoin-cli sendtoaddress 1M72Sfpbz1BPpXFHz9m3CdqATR44Jvaydd 0.1
533ac3682be8723cca63f37a75178155c0b6e69d06606010d5cee1c0f7ccba97

rpcserver.h

```
extern UniValue sendtoaddress(const UniValue& params, bool fHelp); //發送比特幣到特定地址
```

sendtoaddress

wallet/rpcwallet.cpp

```
UniValue sendtoaddress(const UniValue& params, bool fHelp)
{
   if (!EnsureWalletIsAvailable(fHelp))                                    //確保錢包可使用
   if (fHelp || params.size() < 2 || params.size() > 5)                    //確定命令參數數量
   LOCK2(cs_main, pwalletMain->cs_wallet);                                 //錢包上鎖
   CBitcoinAddress address(params[0].get_str());                           //獲取目標地址
   if (!address.IsValid())                                                 //驗證地址是否有效
    CAmount nAmount = AmountFromValue(params[1]);                          //獲取轉帳金額
    if (nAmount <= 0)                                                      //金額數量判斷
    ….                                                                     //取得相關參數
    EnsureWalletIsUnlocked();                                              //確保錢包解密
    SendMoney(address.Get(), nAmount, fSubtractFeeFromAmount, wtx); //發送
    return wtx.GetHash().GetHex();                                         //取得hash
}
```

Sendmoney(目的地址,金額,標誌,備註)

wallet/rpcwallet.cpp

```
static void SendMoney(const CTxDestination &address, CAmount nValue, bool
fSubtractFeeFromAmount, CWalletTx& wtxNew)
{
    CAmount curBalance = pwalletMain->GetBalance();                    //取得餘額
    if (nValue <= 0)                                                   //交易金額為正
    if (nValue > curBalance)                                           //確定錢包餘額
    CScript scriptPubKey = GetScriptForDestination(address);           //從地址拿公鑰
    CReserveKey reservekey(pwalletMain);                               //創建臨時密鑰對
    CAmount nFeeRequired;                                              //所需交易费
    std::string strError;                                              //錯誤訊息
    vector<CRecipient> vecSend;                                        //發送列表
    CRecipient recipient = {scriptPubKey, nValue, fSubtractFeeFromAmount}; //初始化接收者
    vecSend.push_back(recipient);                                      //加入發送列表
    if (!pwalletMain->CreateTransaction(vecSend, wtxNew, reservekey,
            nFeeRequired,nChangePosRet, strError)) {                   //創建交易
    if (!pwalletMain->CommitTransaction(wtxNew, reservekey))           //提交交易
}
```

CreateTransaction(1/2)

wallet/wallet.cpp

```
bool CWallet::CreateTransaction(const vector<CRecipient>& vecSend, CWalletTx& wtxNew, CReserveKey& reservekey, CAmount& nFeeRet,
                                int& nChangePosRet, std::string& strFailReason, const CCoinControl* coinControl, bool sign)
{
    CAmount nValue = 0;                                                    // 1.紀錄發送的總金額
    unsigned int nSubtractFeeFromAmount = 0;                               // 從發送金額減去的總交易費
    BOOST_FOREACH (const CRecipient& recipient, vecSend)                   // 累加總金額
    wtxNew.BindWallet(this);                                               // 交易綁定當下錢包
    txNew.nLockTime = chainActive.Height();                                // 設定交易所定時間
    {
        LOCK2(cs_main, cs_wallet);                                         // 錢包上鎖
        {
            nFeeRet = 0;
            while (true)                                                   //循環直到有足夠的交易費
            {
                txNew.vin.clear();                                         // 清空交易輸入列表
                txNew.vout.clear();                                        // 清空交易輸出列表
                wtxNew.fFromMe = true;                                     // 標記為自己發出的交易
                bool fFirst = true;                                        // 第一次循環標誌
                CAmount nValueToSelect = nValue;                           // 要發送的總金額
                BOOST_FOREACH (const CRecipient& recipient, vecSend)       // 搜尋發送列表
                {
                    CTxOut txout(recipient.nAmount, recipient.scriptPubKey);   // 建構交易對象

                    if (recipient.fSubtractFeeFromAmount)                  // 從金額減去交易費
                    {
                        txout.nValue -= nFeeRet / nSubtractFeeFromAmount;  // 減去平均要減去的交易費
                    }
                    txNew.vout.push_back(txout);                           // 加入交易輸出列表
                }

                set<pair<const CWalletTx*,unsigned int> > setCoins;        // UTXO集
                CAmount nValueIn = 0;                                      // 紀錄選擇UTXO總和
                BOOST_FOREACH(PAIRTYPE(const CWalletTx*, unsigned int) pcoin, setCoins) // 尋找UTXO集
```

CreateTransaction(2/2)

wallet/wallet.cpp

```
            {
                CAmount nCredit = pcoin.first->vout[pcoin.second].nValue;                    // 獲取錢包輸出金額
            }
            const CAmount nChange = nValueIn - nValueToSelect;                               // 找零
            if (nChange > 0)                                                                 // 大於0表示存在找零
            {
                CScript scriptChange;                                                        // 創建找零腳本
                …
                BOOST_FOREACH(const PAIRTYPE(const CWalletTx*,unsigned int)& coin, setCoins)  //搜索UTXO集合
                txNew.vin.push_back(CTxIn(coin.first->GetHash(),coin.second,CScript(),       //加入交易輸入列表
                                std::numeric_limits<unsigned int>::max()-1));
            int nIn = 0;                                                                     // 輸入索引
            CTransaction txNewConst(txNew);                                                  // 建構一筆不變的交易
            BOOST_FOREACH(const PAIRTYPE(const CWalletTx*,unsigned int)& coin, setCoins)     // 尋找UTXO集
            {
                bool signSuccess;                                                            // 簽名狀態
                const CScript& scriptPubKey = coin.first->vout[coin.second].scriptPubKey;    // 獲取腳本公鑰
                CScript& scriptSigRes = txNew.vin[nIn].scriptSig;                            // 獲取腳本簽名引用
                if (sign)
                    signSuccess = ProduceSignature(TransactionSignatureCreator(this, &txNewConst, nIn, SIGHASH_ALL), scriptPubKey,
scriptSigRes);                                                                               // 進行簽名
            unsigned int nBytes = ::GetSerializeSize(txNew, SER_NETWORK, PROTOCOL_VERSION);  // 取得序列化後長度
            *static_cast<CTransaction*>(&wtxNew) = CTransaction(txNew);                      //把交易嵌入CTransaction
        }
    }
  }
  return true;                                                                               // 創建成功
}
```

CommitTransaction(輸入,輸出,找零,簽名)

wallet/wallet.cpp

```
bool CWallet::CommitTransaction(CWalletTx& wtxNew, CReserveKey& reservekey)
{
    {
        LOCK2(cs_main, cs_wallet);                                  // 1.錢包上鎖
        LogPrintf("CommitTransaction:\n%s", wtxNew.ToString());     // 紀錄交易訊息
        if (fBroadcastTransactions)                                 // 若開啟了交易廣播
        {
            if (!wtxNew.AcceptToMemoryPool(false))                  // 3.將交易加進記憶體池中
            wtxNew.RelayWalletTransaction();                        // 4.中斷錢包交易
        }
    }
    return true;
}
```