java离线签名工具

一.获得jar包二种方式

1.下载最新版本的jar包

```
网址：https://github.com/elastos/Elastos.ELA.Utilities.Java/releases

下载：Elastos.ELA.Utilities.Java_v0.1.*.jar
```

2.下载源代码，编译jar包

```
网址：https://github.com/elastos/Elastos.ELA.Utilities.Java
```

- 编译jar包

```
File -> Project Structure -> Artifacts -> + -> JAR -> From modules
with
1、-> Main Class
2、-> extract to the target JAR
3、-> META-INF PATH  (C:\DNA\src\ela_tool\src\main\resources)
4、ok ->  Include in project build -> Apply ->ok
```

- 删除签名jar包依赖，必须使用

```
命令：  zip -d Elastos.ELA.Utilities.Java 'META-INF/*.SF' 'META-
INF/*.RSA' 'META-INF/*SF'
```

二.启动jar包

```
建议java版本：1.8

启动命令：java -cp Elastos.ELA.Utilities.v0.1.*.Java
org.elastos.elaweb.HttpServer
```

```
web服务默认端口：8989，可修改
```

三.创建交易二种方式（自动获取utxo和手动获取utxo）

1.自动获取utxo,也称非离线签名

特点：

- 构造交易方便，不用计算utxo
- 不用计算找零地址金额，拿到从小到大排序的utxo，根据输出(outputs)金额，自动计算找零金额

参数说明：

- java程序金额为最小单位1塞拉(1 ela = 100000000 sela(1亿塞拉))，只能是正整数
- java-config.json 文件需要放在java程序同级目录，目的是连接节点获取utxo
- Host：节点程序所在的服务器ip和rpc端口
- Fee：双方规定的交易费，一笔交易的单个输出或多个输出交易费是一样的
- Confirmation:区块确认交易的次数，即区块数;建议16个确认数
- PrivateKey：输入(inputs)转账需要地址的私钥,java程序内部获取utxo
- Outputs：充值地址及金额
- ChangeAddress：转账后找零的地址，找零金额java程序自动处理
- 输入金额小于输出金额，提示金额不足

- **接口名：genRawTransactionByPrivateKey**

java-config.json

```
{
  "Host": "127.0.0.1:11336",
  "Fee":5000,
  "Confirmation":16
}
```

Request

```
{
    "method":"genRawTransactionByPrivateKey",
    "id":0,
    "params":[
        {
            "Transactions":[
                {
                    "PrivateKeys":[
                        {

"privateKey":"5FA927E5664E563F019F50DCD4D7E2D9404F2D5D49E31F9482912
E23D6D7B9EB"
                        },
                        {

"privateKey":"4C573939323F11BCDB57B61CCE095D4B1E55E986F9944F8807214
1F3DFA883A3"
                        }
                    ],
                    "Outputs":[
                        {

"address":"Eazj14ifau5eH1SP5F8MJRuiSsPMiGbJV1",
                            "amount":28900000
                        },
                        {

"address":"EQSpUzE4XYJhBSx5j7Tf2cteaKdFdixfVB",
                            "amount":60000000
                        }
                    ],

"ChangeAddress":"Edi5WWMFBsEL2qgggrFhnJe1HTjDnw447H"
                }
            ]
        }
    ]
}
```

Response

```
{
    "Action": "genRawTransaction",
    "Desc": "SUCCESS",
```

```
    "Result": {
        "rawTx":
"02000100133232235E5F9463B37EDE39688**********8E7456FDE2554E77E1D9A1
AB3360562F1D6FF4BAC",
        "txHash":
"A32203B48C740552AF0CDB1E77ECCEBE147C5CDA51B2BD80BA9C59662CDCD322"
    }
}
```

2.手动获取utxo,也称离线签名

特点：

- 离线签名，保障账户安全

参数说明：

- java程序金额为最小单位1塞拉(1 ela = 100000000 sela(1亿塞拉))，只能是正整数
- 需要计算找零地址余额，找零余额=inputs-outputs-fee，将找零地址和余额写在outputs最后一行
- txid：地址的可用余额所在的交易,下面接口返回的信息txid写入这里
- index：可用余额所在交易中的序号, 下面接口返回的信息vout为index
- address：outputs的address为转出地址
- privateKey：地址对应的私钥
- amount：转出的金额,long类型

- **listunspent (通过地址获取utxo接口)**

```
获取txid、index:
request:

{
    "method":"listunspent",
    "params":{"addresses": ["8ZNizBf4KhhPjeJRGpox6rPcHE5Np6tFx3",
"EeEkSiRMZqg5rd9a2yPaWnvdPcikFtsrjE"]}
```

```
}

response:

{
    "error": null,
    "id": null,
    "jsonrpc": "2.0",
    "result": [
        {
            "assetid":
"a3d0eaa466df74983b5d7c543de6904f4c9418ead5ffd6d25814234a96db37b0",
            "txid":
"9132cf82a18d859d200c952aec548d7895e7b654fd1761d5d059b91edbad1768",
            "vout": 0,
            "address": "8ZNizBf4KhhPjeJRGpox6rPcHE5Np6tFx3",
            "amount": "33000000",
            "confirmations": 1102
        },
        {
            "assetid":
"a3d0eaa466df74983b5d7c543de6904f4c9418ead5ffd6d25814234a96db37b0",
            "txid":
"3edbcc839fd4f16c0b70869f2d477b56a006d31dc7a10d8cb49bd12628d6352e",
            "vout": 0,
            "address": "8ZNizBf4KhhPjeJRGpox6rPcHE5Np6tFx3",
            "amount": "0.01255707",
            "confirmations": 846
        }
    ]
}
```

- 接口名：genRawTransaction

Request

```
{
    "method":"genRawTransaction",
    "id":0,
    "params":[
        {
            "Transactions":[
                {
                    "UTXOInputs":[
                        {
```

```
"txid":"61c22a83bb96d958f473148fa64f3b2be02653c66ede506e83b82e522200d446",
                            "index":0,

"privateKey":"5FA927E5664E563F019F50DCD4D7E2D9404F2D5D49E31F9482912E23D6D7B9EB"
                        },
                        {

"txid":"a91b63ba6ffdb13379451895c51abd25c54678bc89268db6e6c3dcbb7bb07062",
                            "index":0,

"privateKey":"A65E9FB6735C5FD33F839036B15D2DA373E15AED38054B69386E322C6BE52994",

"address":"EgSph8GNaNSMwpv6UseAihsAc5sqSrA7ga"
                        }
                    ],
                    "Outputs":[
                        {

"address":"ERz34iKa4nGaGYVtVpRWQZnbavJEe6PRDt",
                            "amount":200
                        },
                        {

"address":"EKjeZEmLSXyyJ42xxjJP4QsKJYWwEXabuC",
                            "amount":240
                        }
                    ]
                }
            ]
        }
    ]
}
```


Response


```
{
    "Action":"genRawTransaction",
    "Desc":"SUCCESS",
    "Result":{

"rawTx":"0200010012343332383333AC482F4F********09131B13B648EEF428885
```

```
A5F8AFB44EE38FAC",

"txHash":"B14A65207B801E991292FED3A4CAB06E29D54A792115BC3D45B7F8235
C1A0CF6"
    }
}
```

四.发送交易

- 发送交易是节点rpc接口，java不提供发送交易接口
- sendrawtransaction

Request

```
post请求:  http://127.0.0.1:20336  (20336是节点默认端口)

{
  "method":"sendrawtransaction",
  "params": ["xxxxxx"]
}
```

Response

```
{

"result":"764691821f937fd566bcf533611a5e5b193008ea1ba1396f67b7b0da2
2717c02",
  "id": null,
  "jsonrpc": "2.0",
  "error": null
}
```

五.web rpc 接口

- decodeRawTransaction（反解析rawTransaction）

Request

```
{
    "method":"decodeRawTransaction",
    "id":0,
    "params":[
        {

"RawTransaction":"02000100142D37323733373**********54E77E1D9A1AB336
0562F1D6FF4BAC"
        }
    ]
}
```

Response

```
{
    "Action":"decodeRawTransaction",
    "Desc":"SUCCESS",
    "Result":{
        "UTXOInputs":[
            {

"Txid":"22BADE15481F1AF8240993207E1DF61144A7776E6087994D240917A887F
72052"
            }
        ],
        "Outputs":[
            {
                "Address":"Eazj14ifau5eH1SP5F8MJRuiSsPMiGbJV1",
                "Value":2999000000000000
            }
        ]
    }
}
```

- **genPrivateKey（生成私钥）**

Request

```
{
    "method":"genPrivateKey",
    "id":0,
```

```
    "params":[

    ]
}
```

Response

```
{
    "Action":"genPrivateKey",
    "Desc":"SUCCESS",

"Result":"94F2D1492963E991EA2878C55754293A627277108C2205C7F0EBC592896726D8"
}
```

- **genPublicKey（生成公钥）**

Request

```
{
    "method":"genPublicKey",
    "id":0,
    "params":[
        {

"PrivateKey":"4EA80EDBFC783A19FAC1072D15893AC7A20B4EDE1402FD57DE76D02EA61E28E4"
        }
    ]
}
```

Response

```
{
    "Action":"genPublicKey",
    "Desc":"SUCCESS",

"Result":"03B462F4DB3F67A6A71E51BF3034A183022F092E8E6ED0C91F139E4871F5BA0B57"
}
```

- genAddress（生成地址）

Request

```
{
    "method":"genAddress",
    "id":0,
    "params":[
        {

"PrivateKey":"4EA80EDBFC783A19FAC1072D15893AC7A20B4EDE1402FD57DE76D
02EA61E28E4"
        }
    ]
}
```

Response

```
{
    "Action":"genAddress",
    "Desc":"SUCCESS",
    "Result":"EPUhMEA8RVxqMEvxGDtC95Cwmm1gjtcsB3"
}
```

- gen_priv_pub_addr（生成私钥、公钥、地址）

Request

```
{
    "method":"gen_priv_pub_addr",
    "id":0,
    "params":[

    ]
}
```

Response

```
{
    "Action":"genAddress",
    "Desc":"SUCCESS",
    "Result":{

"PrivateKey":"579750E68061727B023FD0AB8A5ABFEE9FC00491220BA2C824024
63E5AF3E84A",

"PublicKey":"0278421F86F850D73A458680EEA36B49679CD09BE3F0D56E969AF8
F0761E94BC46",
        "Address":"EZ4u7ewRX3LhUCJYZGENpRVPbeCWU2AdXQ"
    }
}
```

- **checkAddress (检查地址)**

Request

检查地址支持**map**格式和数组格式

```
{
    "method":"checkAddress",
    "id":0,
    "params":[
        {
            "Addresses":[
                {
                    "address":"EXgtxGg4ep6vM6uCqWuxkP9KG4AGFyufZz"
                },
                {
                    "address":"1C1mCxRukix1KfegAY5zQQJV7samAciZpv"
                },
                {
                    "address":"8Frmgg4KMudMEPc5Wow5tYXH8XBgctT8QT"
                },
                {
                    "address":"XQd1DCi6H62NQdWZQhJCRnrPn7sF9CTjaU"
                }
                ]
        }
    ]
}
```

```
or

{
    "method":"checkAddress",
    "id":0,
    "params":[
        {
            "Addresses":
["EXgtxGg4ep6vM6uCqWuxkP9KG4AGFyufZz","1C1mCxRukix1KfegAY5zQQJV7sam
AciZpv","8Frmgg4KMudMEPc5Wow5tYXH8XBgctT8QT","XQd1DCi6H62NQdWZQhJCR
nrPn7sF9CTjaU"]
        }
    ]
}
```

Response


```
{
    "Action": "checkAddress",
    "Desc": "SUCCESS",
    "Result": {
        "EXgtxGg4ep6vM6uCqWuxkP9KG4AGFyufZz": true,
        "1C1mCxRukix1KfegAY5zQQJV7samAciZpv": false,
        "8Frmgg4KMudMEPc5Wow5tYXH8XBgctT8QT": true,
        "XQd1DCi6H62NQdWZQhJCRnrPn7sF9CTjaU": false
    }
}
```