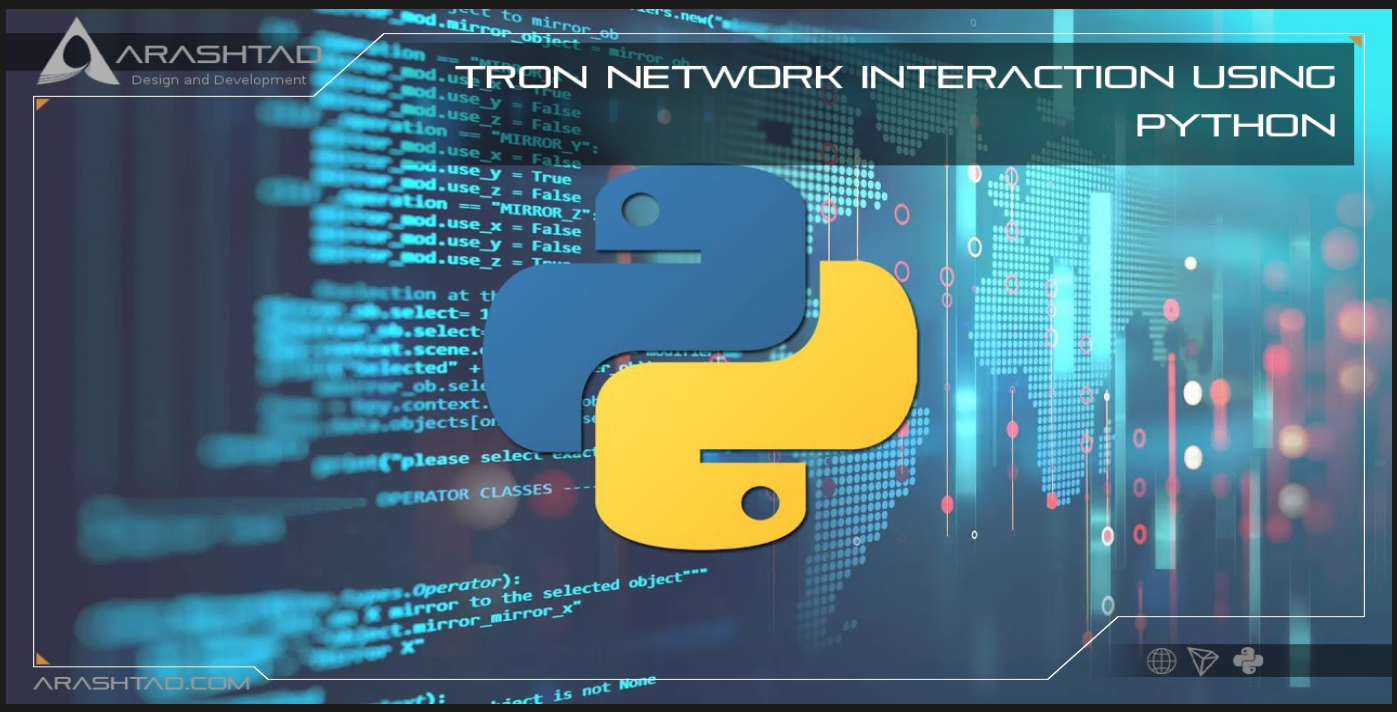
ARASHTAD

Title	A QUICK GUIDE TO TRON NETWORK INTERACTION USING PYTHON.
Description	Intro
Date	2022 14 September
Author	Arashtad
Author URI	https://Arashtad.com



In our Previous article about sending TRC-20 Usdt we explained a little about the Tron network itself, why it is essential and how to send TRC-20 Usdt on Tron network. To make this happen, we first created an Account on shasta testnet. Then we asked for some testnet usdt from the Tron testnet faucet on Tron Discord. Finally transferred our Usdt using the Tronweb JavaScript library. In this tutorial, we will create an account on nile testnet (you can do the same for shasta testnet). Ask for some nile test Tron tokens from the very same faucet we got our shasta Usdt test tokens and then transfer them to another account. Notice that you can do the same on the mainnet, with the difference that there is no faucet and you need to buy the real TRX tokens from an exchange or wallet.

INTRODUCTION TO TRONIX (TRX) TOKEN?

Tronix (TRX) more commonly known as Tron is a token that powers the blockchain with the same name. In order to use Tron-based applications, you must purchase Tronix. For example, if you want to play a Tron-based game or use a Tron-based service, you must purchase TRX. Owning TRX is also a prerequisite to taking part in Tron's consensus system, so you'll need TRX if you plan to stake coins and vote on the protocol's operations. In addition to adding TRX to their portfolios, traders may also be interested in Tron because its platform allows users to create custom applications and tokens. Those who are heavily invested in blockchain applications and tokens may want exposure to Tron. In addition to generating passive income through staking, they may also value the cryptocurrency's ability to offset the risks of holding it or owning it.

TRANSFERRING TRX ON TESTNET

Take the following steps in order to create a transaction on Tron testnet and be able to transfer some test TRX tokens from one account to another. Notice that once you learn how to work with the testnet, you will be able to transfer real tokens on the mainnet.

1. Tronpy Installation

If you haven't installed tronpy, the first step is to install it using the command below.

```
pip install tronpy
```

Notice that you can do the tasks we are doing in this tutorial using the node.js-based library (Tronweb). We have covered a complete tutorial on this subject in another article, where we transferred TRC-20 Usdt from one account to another.

2. Creating Two TRX Accounts:

The next step is to create 2 accounts for transferring TRX from one account to the other. To do this, create a new file called TronAccount.py. Notice that we want to create the test accounts on the nile test network. You can also try it with shasta. To create 2 accounts, copy the following codes in the TronAccount.py file. Then run the code twice. Notice that you need to copy the account address and private key and keep it somewhere safe. Especially when you are working with the mainnet.

```
from tronpy import Tron, Contract
from tronpy.keys import PrivateKey

client = Tron(network="nile")
wallet = client.generate_address()
print("Wallet address:  %s" % wallet['base58check_address'])
print("Private Key:  %s" % wallet['private_key'])

Account_addr = wallet['base58check_address']
```

In the above script, we have simply connected to the nile testnet and created a wallet. In order to connect to the mainnet, you need to simply write nothing inside of the Tron parenthesis.

```
#For connecting to the mainnet
client = Tron()
```

Now, let's run the following command to run the code for creating the sender's account:

```
python TronAccount.py
```

As a result, you will see the following wallet address with its private key:

```
Wallet address: TVmZ1RYSypWvRm3SQdr6d2uDNnkxdqWJBQ
Private Key:
fc507925fb449702e987ae4a963d91b557875b14f74207c3eecb4d688a111284
```

Run the code one more time to create the receiver's account. Make sure you save the address and private key of the two accounts somewhere you can access them later.

```
Wallet address: TRfPqCRLcW9bSmsuw68SDg8Xj5e4d3rugK
Private Key:
ed7e9f9c9e0d3192a873044b7e4cf260fb824d6688dad8ac391bbaa30d1d47b1
```

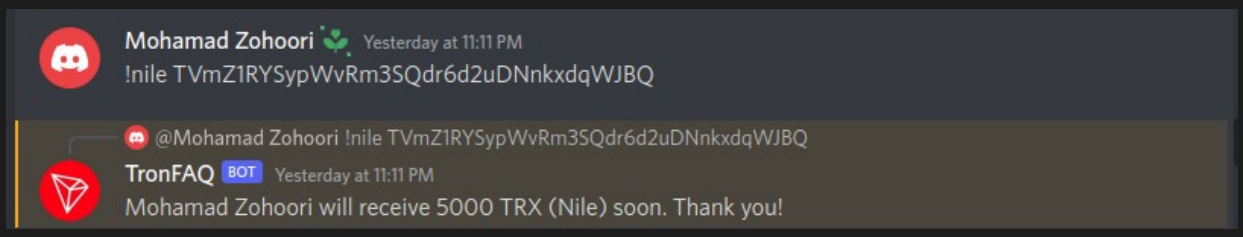
3. Getting test TRX from Tron Faucet:
Now, as we are working with the testnet, we need some test TRX tokens from its faucet. To get some of these tokens, you can create an account on Discord social media and ask for them in the faucet section of the tron community. Next, you should enter the following message inside the message box of the faucet:

```
!nile address_of_your_account
```

Replace the address of your account with the address_of_your_account. If you want shasta test TRX tokens, simply write:

```
!shasta address_of_your_account
```

You can see a sample request message and the response from the robot:



4. Checking the Balance of the account:
To make sure that you have received the tokens, you can use the script below to check it out. Notice that the Tron Faucet usually sends the TRX test tokens after 24 hours. So, don't worry if you face the error saying that the account is not listed on-chain. The main reason for that is the account only gets activated if it has received some tokens from other accounts.

```
from tronpy import Tron

client = Tron(network="nile")
balance =
client.get_account_balance(str('TVmZ1RYSypWvRm3SQdr6d2uDNnkxdqWJBQ'))
print (balance)
```

To run the code use the following command:

```
python Balance.py
```

As you can see, we have received 5000 TRX test tokens:

```
5000
```

5. Transferring TRX :
Now it is time to transfer our tokens from the account that we have some test TRX in it, to the second account. Notice that we should have 3 data including the account address (public key) plus the private key of the sender and the account address of the receiver. The general format of the code is like below. You should enter the mentioned 3 data and also the amount that you want to send in this general format.

```
from tronpy import Tron
from tronpy.keys import PrivateKey

# half Tron & one Tron
HALF_TRON = 500000
ONE_TRON = 1000000

# your wallet information
WALLET_ADDRESS = <'The sender's address'>
PRIVATE_KEY = <'The sender's private key'>

# connect to the Testnet Tron blockchain
client = Tron(network="nile")

# sending some 'amount' of nile test Tron to another nile test address

def send_tron(amount, wallet):
    try:
        priv_key = PrivateKey(bytes.fromhex(PRIVATE_KEY))

        # create transaction and broadcast it
        txn = (
            client.trx.transfer(WALLET_ADDRESS, amount, wallet )
            .memo("Transaction Description")
            .build()
            .inspect()
            .sign(priv_key)
            .broadcast()
        `

5000


    # return the exception
    except Exception as ex:
        print("exception")
        print(ex)
        return ex


recipient_address = <'The recipient address'>
amount =
send_tron(recipient_address, amount)
```

Specifically, for our case with the addresses we have created and 1 TRX that we want to send, the script will become like this.

```
from tronpy import Tron
from tronpy.keys import PrivateKey

# half Tron & one Tron
HALF_TRON = 500000
ONE_TRON = 1000000

# your wallet information
WALLET_ADDRESS = 'TVmZ1RYSypWvRm3SQdr6d2uDNnkxdqWJBQ'
PRIVATE_KEY =
'fc507925fb449702e987ae4a963d91b557875b14f74207c3eecb4d688a111284'

# connect to the Testnet Tron blockchain
client = Tron(network="nile")

# sending some 'amount' of nile test Tron to another nile test address

def send_tron(amount, wallet):
    try:
        priv_key = PrivateKey(bytes.fromhex(PRIVATE_KEY))

        # create transaction and broadcast it
        txn = (
            client.trx.transfer(WALLET_ADDRESS, amount, wallet )
            .memo("Transaction Description")
            .build()
            .inspect()
            .sign(priv_key)
            .broadcast()
        )
        # wait until the transaction is sent through and then return the details
        print(txn.txid)

    # return the exception
    except Exception as ex:
        print("exception")
        print(ex)


recipient_address = 'TRfPqCRLcW9bSmsuw68SDg8Xj5e4d3rugK'
amount = ONE_TRON
send_tron(recipient_address, amount)
```

```
python transfer.py
```

And we should see a result like this, containing all the details of the transaction such as transaction ID (txID), and so on:

```
{'raw_data': {'contract': [{'parameter': {'type_url':
'type.googleapis.com/protocol.TransferContract',
'value': {'amount': 1000000,
'owner_address': '41d92e397bd7ca5c67f20f19ef55cf164dba34da8a',
'to_address': '41ac2390cdace69a6f06863fb6c2d115ffdbde4c5f'}},
'type': 'TransferContract'}],
'data': '5472616e73616374696f6e204465736372697074696f6e',
'expiration': 1662829800365,
'ref_block_bytes': '909f',
'ref_block_hash': 'c878845edbd25388',
'timestamp': 1662829740365},
'signature': [],
'txID': '77c43f401b212aca3cb02874b0d3bd1b2198b1f1bb709eb14c65073397e6d3b8'}
77c43f401b212aca3cb02874b0d3bd1b2198b1f1bb709eb14c65073397e6d3b8
```

To check whether the transaction has been successful or not, we can check the balance of the sender. To do so, you should enter the address inside the script. Then run the command below:

```
python transfer.py
```

As you can see 1 TRX has been reduced from the balance plus the 0.1 TRX gas fee.

```
4998.9
```

WRAPPING UP:

In this tutorial-based article, you learned how to install tronpy, create a TRX account using it, get some test tokens from the Tron faucet, and in the end, learned how to check the balance of the accounts and transfer the test tokens from the sender to the receiver. This was one key tutorial because you learned how to create and sign a transaction on the Tron network as well. This helps you to interact with smart contracts and also deploy your smart contract on this popular network with the benefit of paying fewer gas fees compared to the Ethereum blockchain.

★ ★ ★

Follow Arashtad on Social Media