NEM это просто

https://nem.page.link/simplenem

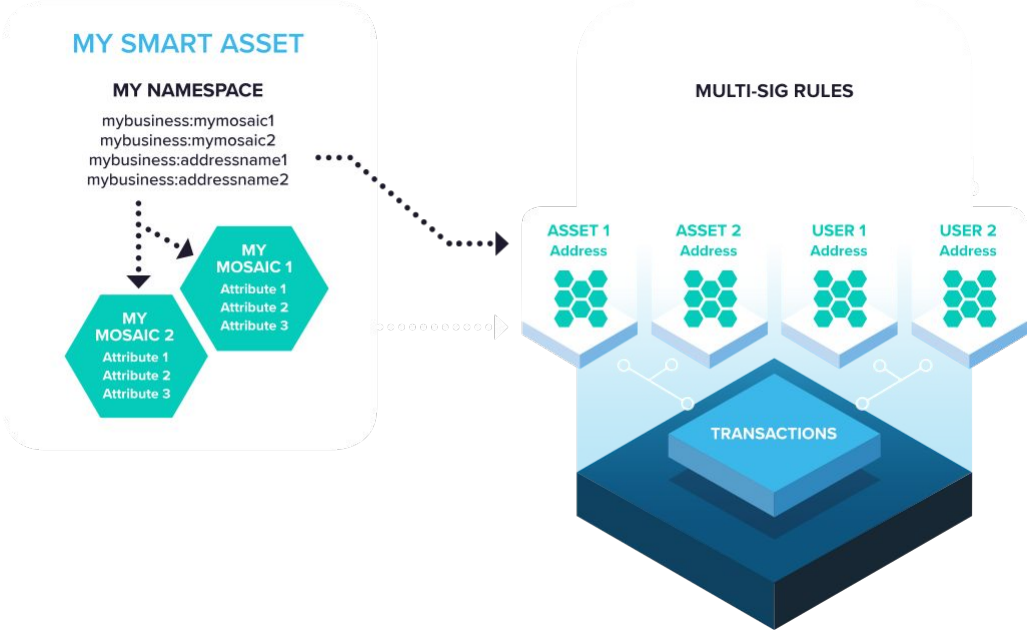
Что такое NEM?

The Smart Asset Blockchain

NEM - написанная с нуля платформа
для создания блокчейн-приложений,
это не просто криптовалюта

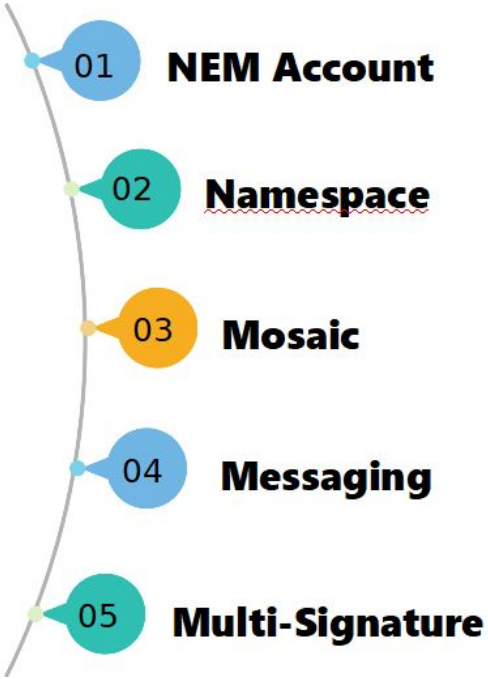
Взаимодействие с NEM

NEM предоставляет вам прямой доступ к специальным протестированным on-blockchain плагинам, посредством API. Вам не придется изучать новый язык и его особенности.



NEM как конструктор

Ингредиенты NEM:



NEM Аккаунты

1. Адреса в NEM начинаются с N - Mainnet, M - Mijin, T - Testnet и имеют длину 40 символов
2. Второй символ всегда A, B, C, или D, поэтому адреса которые начинаются с NE или N4 невалидны.
3. Цифры 0, 1, 8, 9 не могут быть в адресе

Берегите свои приватные ключи!

NEM Аккаунты

NEM аккаунт это не кошелек!

Public key: You need to make a transaction to get a public key

Публичный ключ будет записан в блокчейн только после того как аккаунт совершит исходящую транзакцию.

https://nem-toolchain.github.io/

- Generate new accounts for mijin, mainnet, testnet.
- Show accounts details for any given private key.
- Find vanity address by a given list of prefixes.

```
root@kali:~# nem --chain mainnet account vanity --skip-estimate ______-______-______-______-______-______-KIEV
----
Address: NASJMR-V2GP4I-PCRZVY-77MNDY-G7CBFP-3QXHRY-KIEV
Public key: 9304634d3c28964205e98bf079e82a3d63c1ddb609a99ce315acbccaab79f92a
Private key: a9ff7138d2935aa69ebd432a4e2ccb41cd380dd49a37fe9ebe4b709dc97fa796
```

NEM Пространства имен

Namespace позволяет иметь уникальное именование в системе по аналогии с системой доменов в интернет

1. Максимальная длина корневого неймспейса 16 символов (ochenkrutoyprojekt нельзя 18 букв)
2. Максимальная длина вложенных 64 символа
3. До 3х уровней вложенности (например scam.token.cash можно, scam.token.cash.buynow нельзя)
4. Разрешенные символы: a, b, c, ..., z, 0, 1, 2, ..., 9, _ , -
5. Обязано начинаться на букву (-coin или 1coin нельзя)
6. Зарезервированные пространства имен: nem, user, account, org, com, biz, net, edu, mil, gov, info

Namespace alias

Только владелец пространства имен может создавать мозаики.

NEM Мозаики

Мозаика является жетоном (token) для представления какого-либо определенного объекта из реального мира. Например: игровая валюта, балы лояльности, фишки казино, ценные бумаги, документ, подтверждения права голосования и как реальные голоса.

1. Максимальная длина имени 32 символа
2. Максимальная длина описания мозаики 512
3. Разрешенные символы: a, b, c, ..., z, 0, 1, 2, ..., 9, ‘, _ , -

Переопределение свойств мозаики возможно если весь выпуск находится на аккаунте эмитента.
Стоимость создания 10 xem.
Мозаики отправляются пачками (bags) в одной транзакции может быть отправлено до 10 уникальных мозаик

NEM Сообщения

1. Обычные сообщения
2. Зашифрованные сообщения (используются private и public key отправителя и получателя)
3. Hex сообщения

В БД сообщения хранятся в hex-виде

hex2utf8(‘476f6f64206c75636b21’) = Good luck!

Hex сообщения начинаются со спецсимвола. Максимальная длина 1024 байта.

В Nanowallet мультиподписные аккаунты не могут отправлять зашифрованные сообщения, так как соподписанты не могут их расшифровать, не зная private key мультиподписного аккаунта.
Так же вы не можете отправить зашифрованное сообщение на аккаунт без известного публичного ключа.

NEM Мультиподписи

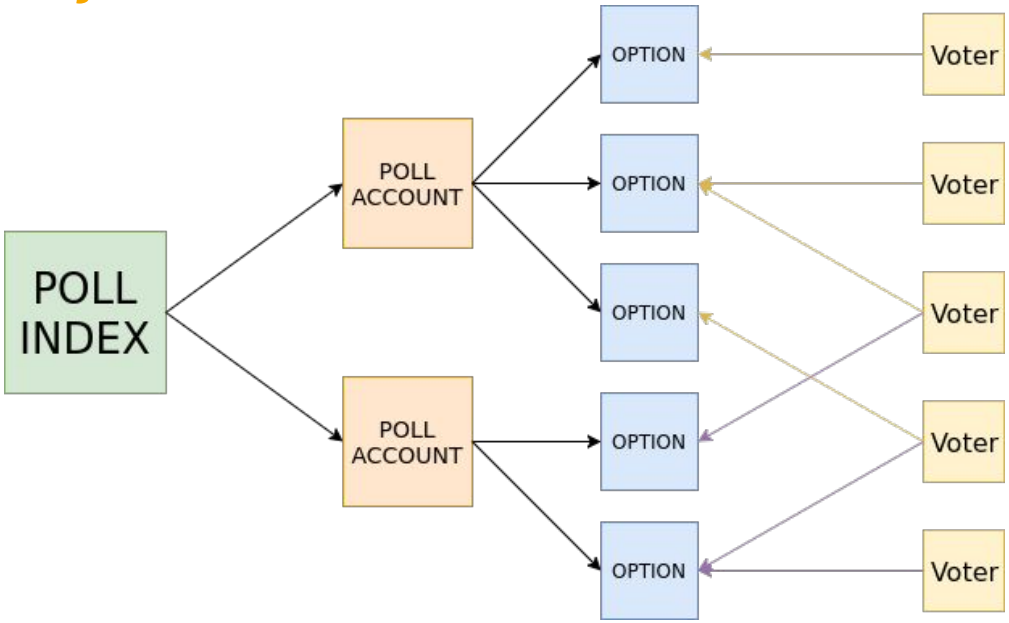
1. До 32 соподписантов
2. Соподписант не может быть мультиподписным аккаунтом (1 уровень)
3. Мультиподисной аккаунт можно сделать обычным, если удалить всех соподписантов.

M из N подписей позволяют гибко настраивать параметры безопасности аккаунта

После создания мультиподписного аккаунта, приватный ключ не может быть использован для подписания транзакций этого аккаунта, все транзакции инициируются только соподписантами. Все комиссии оплачиваются с мультиподписного аккаунта, поэтому соподписанты могут иметь 0 xem на счету и выполнять транзакции.

Голосование

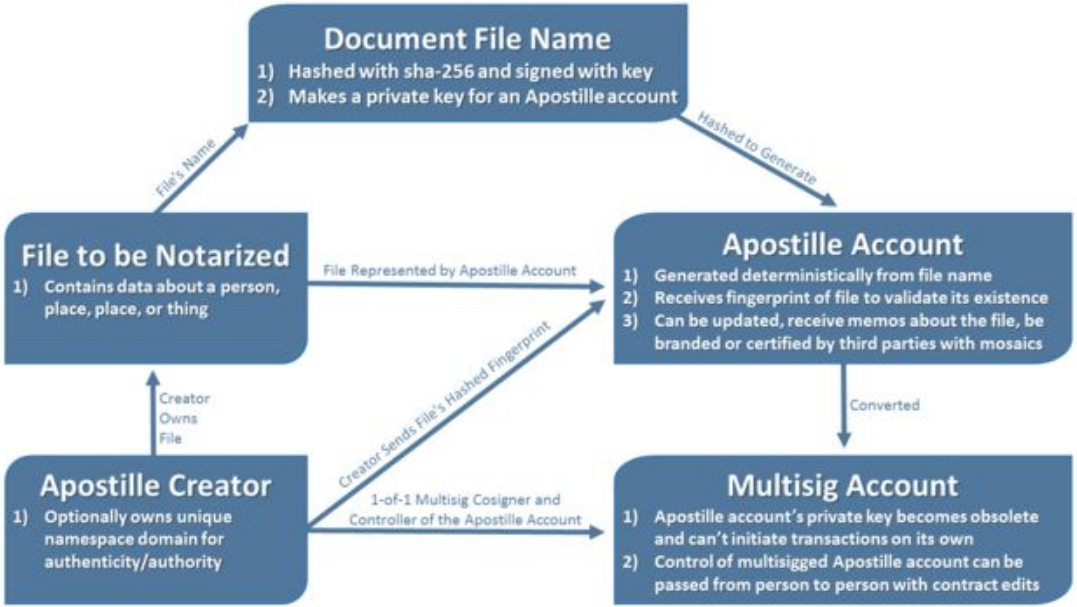
аккаунты+сообщения



https://github.com/shierve/NEM-voting-specifications

NEM Апостиль

аккаунты+сообщения+мультиподпись

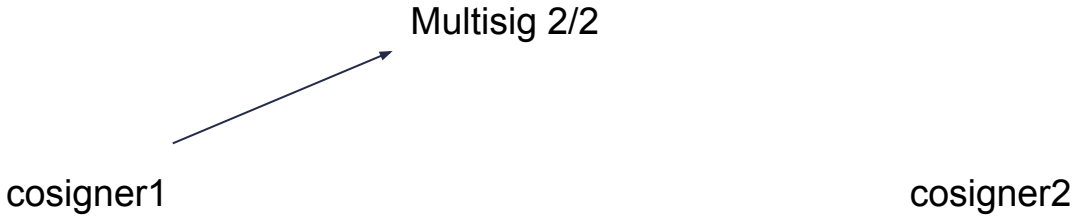


https://blog.nem.io/apostille-tutorial/

Неподтвержденные транзакции

А обязательно ли все писать в блокчейн?

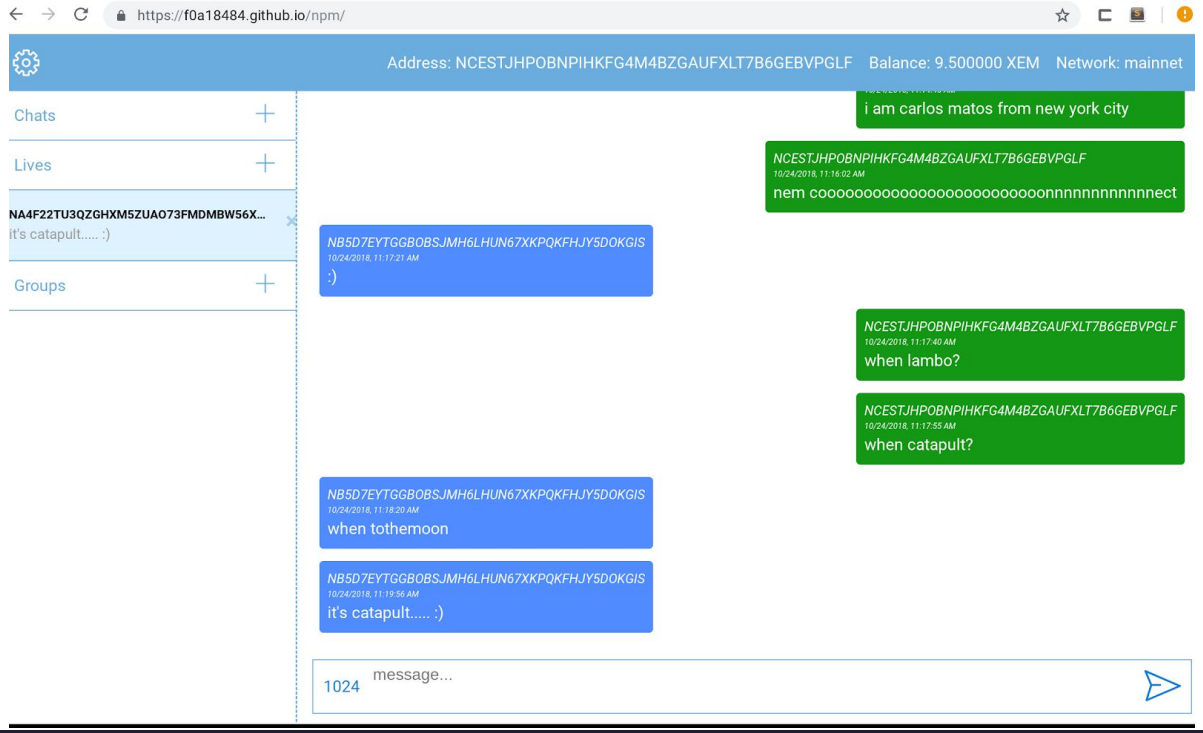
Сила неподтвержденных транзакций в том что они бесплатны. Расходы мультиподписного аккаунта кешируются. Ноды считают сколько неподтвержденных транзакций сделал аккаунт и сколько комиссий он должен до отмены транзакции.



Cosigner1 инициирует транзакции, а Cosigner2 их не подписывает

Snapchat на NEM?

сообщения+мультиподпись



https://f0a18484.github.io/npm/

Snapchat на NEM?

сообщения+мультиподпись

Регистрация = Создание мультиподисного 2/2 аккаунта

Lives - общий адрес на который отправляются незашифрованные сообщения.

Chat - 2 аккаунта отправляют зашифрованные сообщения друг другу.

Group - работает так же как и lives, каждый участник знает приватный ключ общего адреса для расшифровки сообщений.

Время жизни транзакции 10 секунд.
Открытый исходный код.

Пропускная способность

NEM NIS1 tps 2 транзакции в секунду

В 1 блоке может быть до 120 транзакций

Кеш неподтвержденных транзакций имеет 1000 слотов

Если в кеше неподтвержденных транзакции занято менее 120 слотов, то любой инициатор может отправить транзакцию.

Ограничение спам фильтра:
Каждый последующий слот предоставляется инициатору с большей комиссией или важностью.
Величина комиссии и важности возрастает экспоненциально, поэтому занять все слоты практически невозможно.

P2P NETWORK

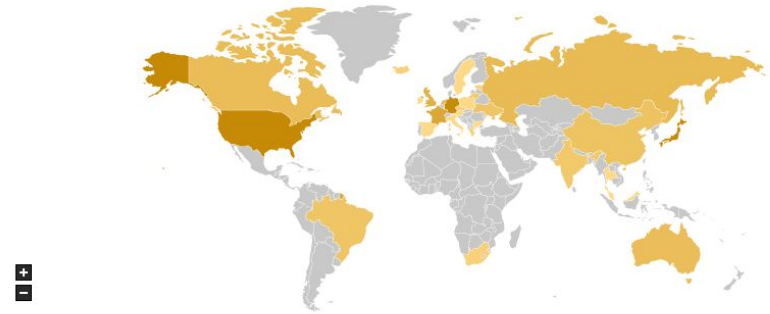
http://nodeexplorer.com

NODEEXPLORER

Welcome | API | Latest release: 0.6.96 | Sponsored by NEM! | Terms and Data Privacy Policy

1834844 Block Height (best block)	2 minutes Last Block (ago)	63 Average Block Time (in seconds)	971 999 999 Marketcap (1 XEM = 0.108$)
464 Nodes Online (now)	516 Nodes Online (average)	53 (Vultr Hold...) Node ISP (most used)	121 Node Average Response Time (ms)

FULL NODE (LOCATION)



Top Locations	Nodes
United States	105
Japan	100
Germany	98
France	27
Singapore	18
United Kingdom	18
Netherlands	13

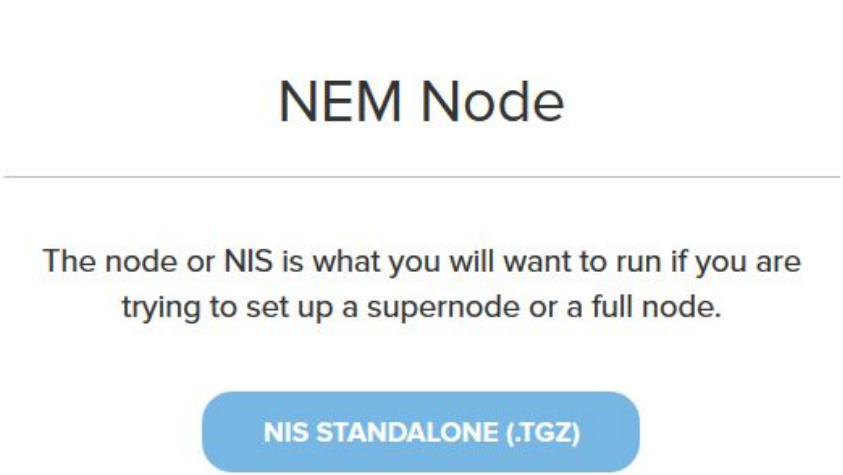
nodeexplorer.com

https://nodeexplorer.com/api_openapi_jsonp

nodeexplorer({"nodes":["104.238.161.61:7890", "153.122.86.201:7890", "45.77.170.104:7890", "199.217.118.114:7890", "62.109.20.138:7890", "163.44.170.40:7890", "191.232.233.187:7890", "206.189.237.252:7890", "47.254.128.67:7890", "173.212.251.246:7890",
"170.75.146.181:7890", "47.90.206.223:7890", "82.223.21.84:7890", "108.61.251.209:7890", "104.128.74.141:7890", "210.16.120.28:7890", "185.183.157.46:7890", "78.46.241.158:7890", "153.122.13.96:7890", "149.28.247.129:7890", "133.130.91.240:7890", "207.148.93.118:7890",
"193.29.57.226:7890", "138.201.155.196:7890", "159.89.146.5:7890", "185.183.157.131:7890", "222.230.111.179:7890", "163.44.170.172:7890", "170.75.153.141:7890", "191.232.214.107:7890", "150.95.205.30:7890", "160.16.201.189:7890", "47.91.78.2:7890",
"85.214.147.237:7890", "45.76.133.25:7890", "139.99.168.68:7890", "150.95.137.138:7890", "170.75.153.140:7890", "47.91.73.206:7890", "195.162.81.89:7890", "37.59.120.16:7890", "204.48.24.142:7890", "173.212.238.176:7890", "213.252.247.126:7890", "144.217.88.207:7890",
"133.130.123.154:7890", "107.161.126.132:7890", "213.136.80.93:7890", "82.202.162.62:7890", "153.122.13.112:7890", "149.28.197.154:7890", "199.247.5.101:7890", "171.100.233.237:7890", "66.42.62.98:7890", "133.130.123.50:7890", "5.189.155.137:7890",
"94.130.76.171:7890", "191.232.50.79:7890", "47.91.88.194:7890", "88.99.87.132:7890", "185.185.40.26:7890", "45.63.69.29:7890", "5.189.173.249:7890", "149.28.197.243:7890", "199.233.237.83:7890", "78.47.148.107:7890", "159.69.91.123:7890", "153.122.84.98:7890",
"88.198.116.20:7890", "150.95.153.229:7890", "213.252.247.125:7890", "217.182.66.115:7890", "47.91.78.211:7890", "51.38.177.8:7890", "176.223.141.72:7890", "138.201.158.14:7890", "153.122.13.99:7890", "150.95.175.183:7890", "153.126.160.85:7890", "188.68.38.88:7890",
"178.254.23.3:7890", "178.254.39.181:7890", "47.254.148.56:7890", "176.122.164.233:7890", "62.75.251.134:7890", "159.69.50.30:7890", "103.27.76.170:7890", "82.196.7.64:7890", "178.63.1.41:7890", "185.243.112.27:7890", "160.16.80.240:7890", "150.95.147.85:7890",
"133.130.122.209:7890", "45.32.196.216:7890", "95.82.223.13:7890", "185.122.58.130:7890", "188.166.30.239:7890", "107.191.50.14:7890", "153.122.13.178:7890", "95.130.254.153:7890", "78.47.64.34:7890", "109.69.61.3:7890", "45.124.66.70:7890", "193.70.0.150:7890",
"54.38.54.86:7890", "188.166.2.98:7890", "46.252.24.18:7890", "144.202.57.206:7890", "65.49.203.200:7890", "78.47.48.110:7890", "173.249.58.188:7890", "199.233.237.85:7890", "153.126.186.201:7890", "185.11.139.147:7890", "212.109.193.248:7890", "185.122.58.143:7890",
"78.46.241.159:7890", "149.28.198.184:7890", "89.250.240.60:7890", "185.117.22.168:7890", "37.120.172.158:7890", "45.77.174.157:7890", "80.241.222.232:7890", "160.16.97.178:7890", "154.127.52.243:7890", "138.201.154.227:7890", "213.126.47.206:7890",
"153.122.13.88:7890", "185.122.59.228:7890", "149.56.131.89:7890", "160.16.63.95:7890", "160.16.195.190:7890", "153.126.157.53:7890", "110.44.135.87:7890", "195.162.81.80:7890", "108.61.204.81:7890", "5.101.179.24:7890", "45.63.98.203:7890", "74.208.42.132:7890",
"51.254.36.253:7890", "45.124.64.143:7890", "185.162.248.183:7890", "88.86.222.147:7890", "121.50.43.168:7890", "5.189.145.181:7890", "160.16.126.235:7890", "45.32.247.110:7890", "13.59.29.47:7890", "150.95.157.157:7890", "88.198.116.18:7890", "217.182.66.114:7890",
"202.71.98.123:7890", "45.124.65.125:7890", "157.7.198.84:7890", "138.201.157.246:7890", "95.169.29.194:7890", "45.32.192.155:7890", "172.82.183.28:7890", "149.56.98.80:7890", "159.89.122.39:7890", "88.198.116.16:7890", "185.243.113.237:7890", "45.77.33.181:7890",
"153.122.13.97:7890", "153.126.183.39:7890", "159.89.146.73:7890", "138.201.157.248:7890", "78.47.250.109:7890", "207.148.99.87:7890", "213.136.86.202:7890", "173.249.34.67:7890", "139.59.71.243:7890", "157.7.131.206:7890", "150.95.128.62:7890", "149.56.103.225:7890",
"97.64.33.218:7890", "185.53.131.101:7890", "178.238.227.90:7890", "153.122.13.90:7890", "185.243.113.234:7890", "45.58.49.191:7890", "47.88.230.169:7890", "133.130.116.102:7890", "170.75.153.139:7890", "128.199.244.45:7890", "95.169.14.208:7890",
"185.122.58.113:7890", "144.202.98.59:7890", "45.63.27.20:7890", "157.7.197.62:7890", "95.163.109.156:7890", "74.208.244.49:7890", "138.201.154.165:7890", "14.202.176.236:7890", "108.61.96.212:7890", "160.16.198.74:7890", "54.199.183.90:7890", "170.75.146.179:7890",
"210.16.120.204:7890", "138.201.155.197:7890", "45.76.184.63:7890", "185.122.56.89:7890", "185.185.40.67:7890", "153.122.86.21:7890", "153.122.85.116:7890", "45.32.144.178:7890", "150.95.157.21:7890", "45.77.248.215:7890", "153.122.13.100:7890", "153.122.84.23:7890",
"104.128.74.139:7890", "185.162.249.8:7890", "153.122.84.113:7890", "47.88.174.110:7890", "163.44.168.183:7890", "5.45.110.242:7890", "103.207.68.56:7890", "79.137.34.6:7890", "108.61.166.217:7890", "47.91.77.101:7890", "178.128.239.230:7890", "45.77.187.43:7890",
"108.61.182.27:7890", "160.119.248.49:7890", "153.126.163.16:7890", "172.93.49.17:7890", "150.95.131.241:7890", "108.61.168.86:7890", "213.136.89.115:7890", "153.122.86.245:7890", "172.93.99.37:7890", "178.254.36.69:7890", "78.46.195.25:7890", "23.163.0.42:7890",
"133.130.121.238:7890", "78.47.156.244:7890", "104.251.212.131:7890", "157.7.135.123:7890", "153.122.13.92:7890", "217.182.66.112:7890", "54.37.75.182:7890", "65.49.211.62:7890", "153.122.13.52:7890", "37.235.49.13:7890", "104.128.68.102:7890", "69.171.69.92:7890",
"188.225.38.223:7890", "45.77.1.100:7890", "74.118.138.21:7890", "193.70.0.148:7890", "170.75.146.180:7890", "46.38.240.127:7890", "172.107.17.88:7890", "104.128.238.136:7890", "47.254.133.1:7890", "172.93.49.169:7890", "150.95.145.117:7890", "138.201.157.245:7890",
"74.120.168.156:7890", "216.189.147.206:7890", "193.29.58.186:7890", "178.254.31.95:7890", "153.122.53.148:7890", "104.41.59.146:7890", "153.122.13.89:7890", "160.16.100.162:7890", "153.122.60.153:7890", "86.7.197.213:7890", "138.201.157.249:7890",
"45.32.134.182:7890", "45.32.205.23:7890", "138.201.186.71:7890", "83.85.221.92:7890", "160.16.239.29:7890", "153.122.13.86:7890", "153.122.13.98:7890", "45.63.76.253:7890", "45.63.79.55:7890", "174.138.24.208:7890", "46.252.27.237:7890", "157.7.135.224:7890",
"153.122.84.36:7890", "172.93.49.166:7890", "138.68.240.171:7890", "133.130.115.129:7890", "88.198.116.17:7890", "62.75.171.41:7890", "45.76.120.189:7890", "89.107.56.153:7890", "104.128.68.205:7890", "153.126.160.251:7890", "138.201.118.129:7890",
"103.192.177.252:7890", "52.2.187.5:7890", "153.122.13.94:7890", "104.128.68.101:7890", "149.28.196.255:7890", "185.243.113.238:7890", "217.182.66.116:7890", "94.250.251.245:7890", "212.227.10.213:7890", "51.68.136.65:7890", "144.217.88.193:7890", "104.41.34.175:7890",
"163.44.175.223:7890", "133.130.121.211:7890", "78.47.64.36:7890", "107.161.126.131:7890", "74.118.138.61:7890", "138.201.155.194:7890", "185.185.41.53:7890", "150.95.136.134:7890", "153.122.115.170:7890", "5.189.173.196:7890", "78.47.148.106:7890",
"167.99.152.66:7890", "193.70.0.151:7890", "78.47.148.104:7890", "95.163.200.130:7890", "94.16.112.237:7890", "95.130.254.155:7890", "159.89.154.139:7890", "193.70.0.147:7890", "217.182.66.113:7890", "192.99.54.221:7890", "82.101.233.100:7890", "103.42.225.46:7890",
"217.182.74.161:7890", "45.76.174.212:7890", "82.202.167.195:7890", "160.16.140.161:7890", "5.189.142.133:7890", "104.156.237.208:7890", "51.141.28.95:7890", "160.119.248.48:7890", "45.63.78.67:7890", "45.76.178.172:7890", "45.63.107.13:7890", "52.42.99.254:7890",
"85.214.32.50:7890", "88.198.125.199:7890", "45.76.144.42:7890", "80.240.16.53:7890", "138.201.154.90:7890", "88.198.116.19:7890", "153.122.13.35:7890", "213.252.247.12:7890", "193.29.56.12:7890", "217.182.74.162:7890", "52.163.228.196:7890", "34.245.57.239:7890",
"153.122.85.149:7890", "153.122.84.183:7890", "47.91.75.45:7890", "193.70.0.149:7890", "153.122.13.168:7890", "13.76.243.137:7890", "153.122.13.93:7890", "199.233.237.84:7890", "209.58.177.181:7890", "173.212.247.199:7890", "108.175.5.146:7890", "62.75.163.236:7890",
"162.243.227.155:7890", "54.169.91.123:7890", "18.191.109.28:7890", "13.251.166.26:7890", "212.237.32.21:7890", "185.122.58.141:7890", "78.47.150.129:7890", "203.111.44.54:7890", "198.199.112.62:7890", "52.53.207.99:7890", "66.228.48.37:7890", "185.86.150.172:7890",
"45.32.250.173:7890", "150.95.133.243:7890", "216.189.157.228:7890", "47.91.89.114:7890", "183.181.34.30:7890", "157.7.196.200:7890", "45.77.31.211:7890", "153.122.84.34:7890", "34.208.76.79:7890", "54.37.136.53:7890", "45.76.199.119:7890", "47.254.133.181:7890",
"47.254.132.177:7890", "185.164.4.249:7890", "178.79.181.190:7890", "172.107.17.80:7890", "47.91.94.195:7890", "94.130.167.236:7890", "195.201.11.66:7890", "47.254.134.237:7890", "13.114.63.69:7890", "45.32.41.5:7890", "45.76.194.171:7890", "45.79.94.42:7890",
"47.91.94.68:7890", "65.132.7.227:7890", "78.47.156.251:7890", "54.37.19.3:7890", "153.122.13.80:7890", "178.63.18.3:7890", "207.154.254.48:7890", "78.47.156.246:7890", "153.122.84.12:7890", "45.124.64.16:7890", "13.76.172.76:7890", "148.251.255.171:7890"]})

NIS1

https://nem.io/downloads/



https://www.java.com/en/download/

NIS1

config.properties

nem.protocol = http
nem.host = 127.0.0.1
nem.httpPort = 7890
nem.httpsPort = 7891
nem.websocketPort = 7778

```
# Denotes the name of the network to join. Available are:
# testnet - the test network that uses test coins and where new features are tested before they are released to the main network.
# mainnet - the main network that uses real coins.
nem.network = mainnet
```

```
# Maximum number of unlocked accounts. Meaning: maximum number of accounts that are allowed to use this NIS for harvesting
# Keep the value within a reasonable range, a too large value an cause problems for all harvesting accounts.
nis.unlockedLimit = 4
```

NIS1

config.properties

```
# Optional features supported by the local node (pipe-separated).
# TRANSACTION_HASH_LOOKUP: transactions can be retrieved by supplying the transaction hash.
# HISTORICAL_ACCOUNT_DATA: historical account data can be retrieved.
nis.optionalFeatures = TRANSACTION_HASH_LOOKUP
```

nix.runNis.sh || runNis.bat

По умолчанию

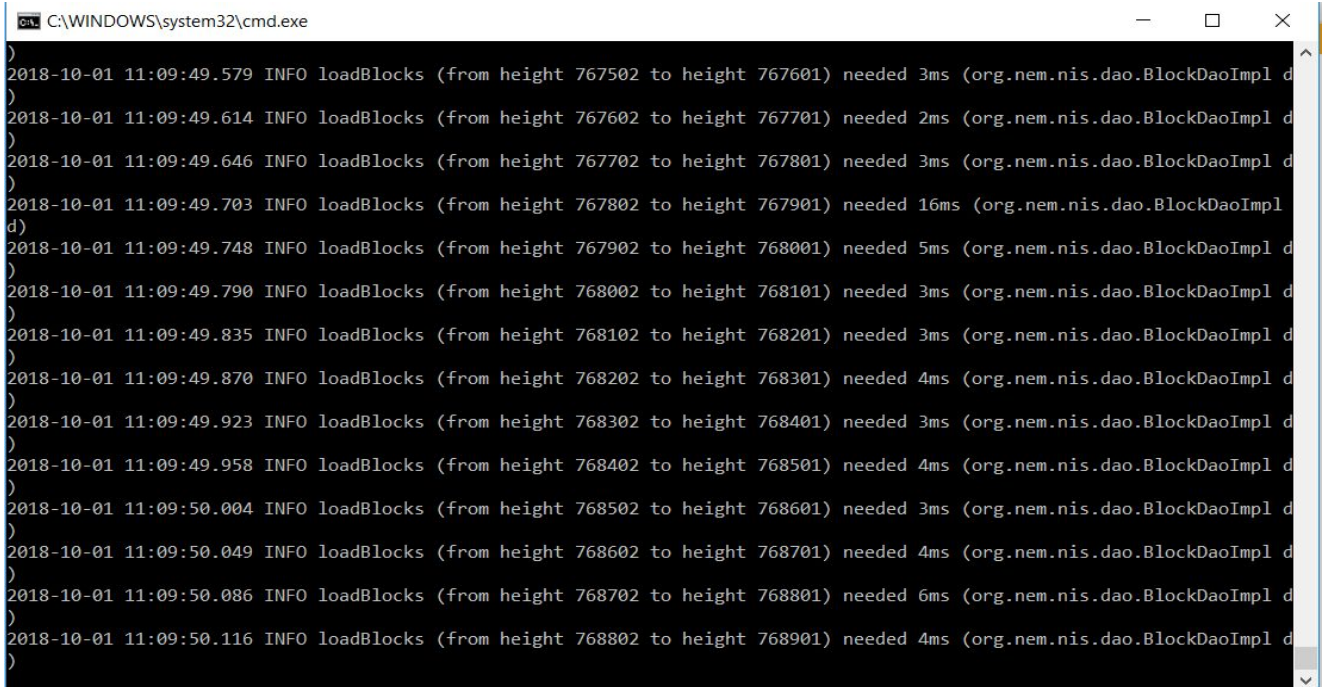
java -Xms512M -Xmx1G -cp ".:./*:../libs/*" org.nem.deploy.CommonStarter

Рекомендуется

java -Xms1G -Xmx2G -cp ".;./*;../libs/*" org.nem.deploy.CommonStarter

NIS1

Синхронизация



NIS1

http://bob.nem.ninja/

Index of /

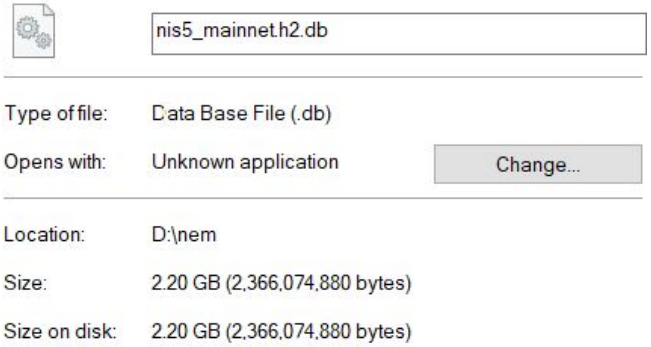
Name	Last Modified	Size	Type
Parent Directory/		-	Directory
.well-known/	2017-Dec-19 12:10:51	-	Directory
beta-testnet/	2017-Dec-22 19:01:04	-	Directory
bin/	2016-Jan-29 12:06:37	-	Directory
docs/	2017-Mar-20 20:37:38	-	Directory
installer/	2018-Aug-10 19:04:24	-	Directory
last-ncc/	2017-Jul-30 16:40:10	-	Directory
org.nem.core/	2018-Aug-10 18:51:08	-	Directory
README.txt	2014-Jul-28 23:15:37	1.8K	text/plain
lightwallet-standalone-1.8-cdn-nginx1.8.zip	2016-May-16 20:33:09	1.3M	application/zip
lightwallet-standalone-1.8-cdn-nginx1.8.zip.sig	2016-May-16 20:33:11	0.2K	application/pgp-signature
nis-0.6.93.tgz	2017-Jul-30 16:35:18	25.1M	application/x-gtar-compressed
nis-0.6.93.tgz.sig	2017-Jul-30 16:35:18	0.1K	application/pgp-signature
nis-0.6.95.tgz	2017-Dec-22 19:01:23	26.1M	application/x-gtar-compressed
nis-0.6.95.tgz.sig	2018-Aug-10 19:02:13	0.1K	application/pgp-signature
nis-0.6.96.tgz	2018-Aug-10 18:51:06	26.1M	application/x-gtar-compressed
nis-0.6.96.tgz.sig	2018-Aug-13 11:03:29	0.1K	application/pgp-signature
nis5_mainnet.h2-1453k.db.zip	2018-Jan-10 09:17:18	529.2M	application/zip
nis5_mainnet.h2-1453k.db.zip.sig	2018-Jan-10 10:41:49	0.1K	application/pgp-signature
nis5_mainnet.h2-1680k.db.zip	2018-Jun-18 18:54:20	699.5M	application/zip
nis5_mainnet.h2-1680k.db.zip.sig	2018-Jun-19 12:15:52	0.1K	application/pgp-signature
servant_0_0_4.zip	2016-Jun-01 11:02:45	10.0M	application/zip
servant_0_0_4.zip.sig	2016-Jun-01 11:03:31	0.2K	application/pgp-signature
test.json	2015-Apr-21 22:32:29	0.1K	application/json
version.json	2018-Aug-10 18:51:08	0.1K	application/json
version.txt	2018-Aug-10 18:51:08	0.1K	text/plain
version_p.js	2018-Aug-10 18:51:08	0.1K	application/javascript

Где блокчейн?

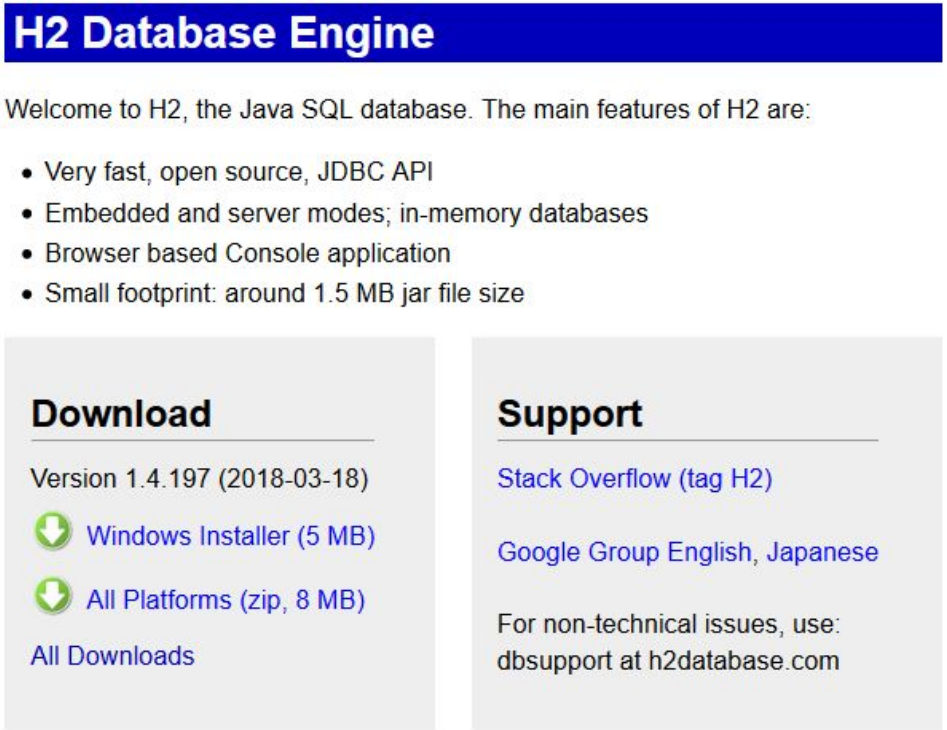
h2 database

config.properties

```
5  # In order to be able to start the NEM software (NIS and also NCC) from a removeable media (e.g. USB drive),
6  # the path, where the database, logs and other files will be saved, can be configured.
7  # %h - is the home directory of the user (also if the entry is missing) (example: nem.folder = %h/nem)
8  # . - is the working directory (example: nem.folder = ./nem)
9  nem.folder = %h/nem
```



А что же там внутри?



H2 Database Engine

Welcome to H2, the Java SQL database. The main features of H2 are:

- Very fast, open source, JDBC API
- Embedded and server modes; in-memory databases
- Browser based Console application
- Small footprint: around 1.5 MB jar file size

Download

Version 1.4.197 (2018-03-18)

Windows Installer (5 MB)

All Platforms (zip, 8 MB)

All Downloads

Support

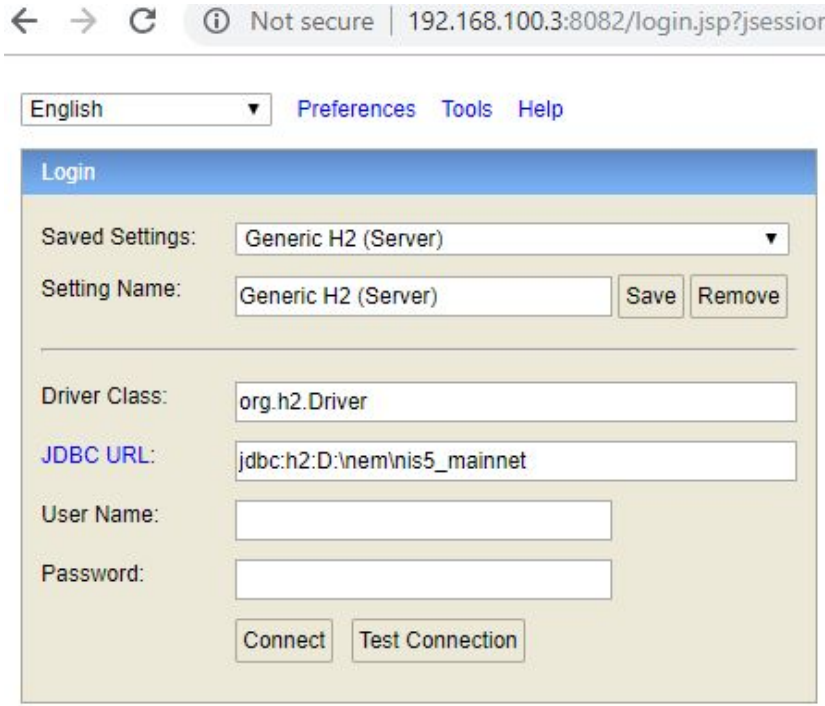
Stack Overflow (tag H2)

Google Group English, Japanese

For non-technical issues, use:
dbsupport at h2database.com

www.h2database.com

Подключение к БД



Python jaydebeapi

https://pypi.org/project/JayDeBeApi/

The JayDeBeApi module allows you to connect from Python code to databases using Java JDBC

Аккаунты

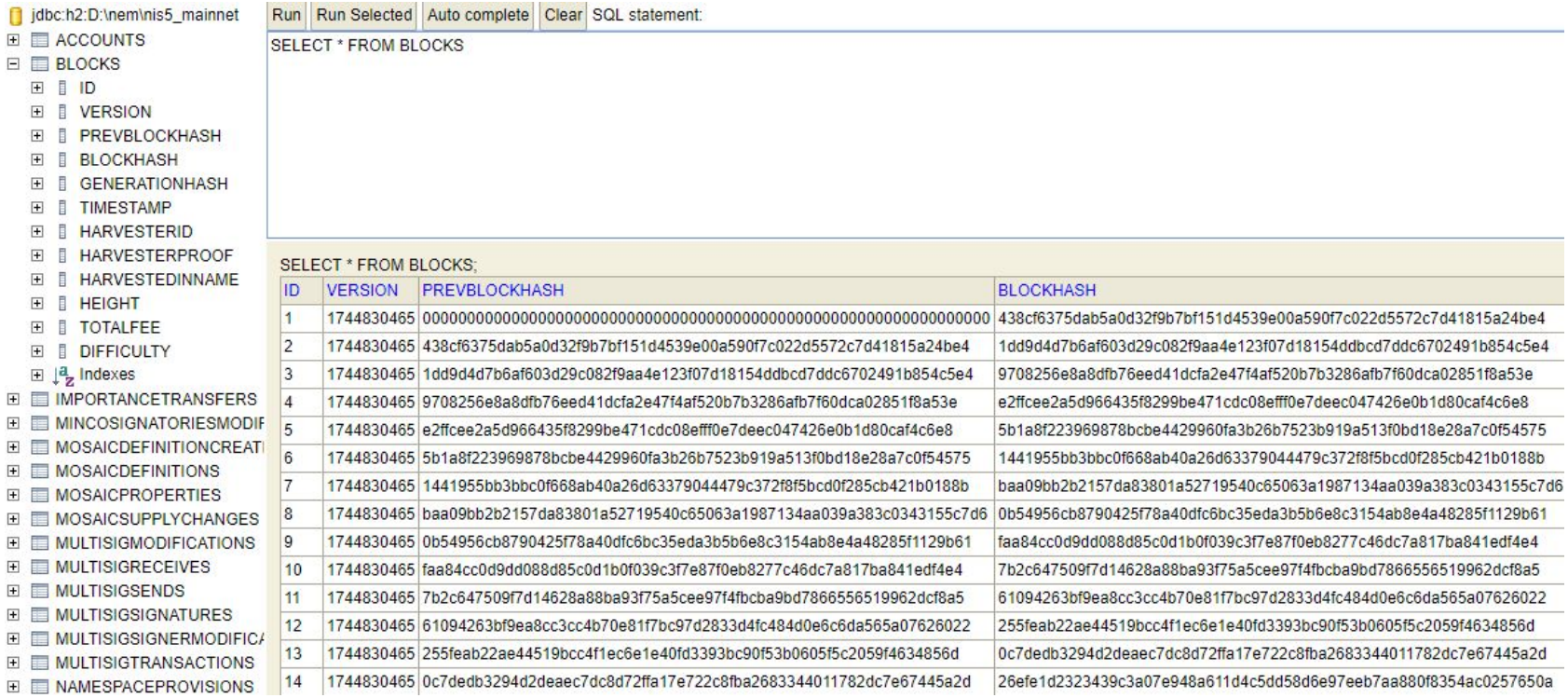
- ACCOUNTS
  - ID
  - PRINTABLEKEY
  - PUBLICKEY
  - Indexes
- BLOCKS
- IMPORTANCETRANSFERS
- MINCOSIGNATORIESMODIF
- MOSAICDEFINITIONCREAT
- MOSAICDEFINITIONS
- MOSAICPROPERTIES
- MOSAICSUPPLYCHANGES
- MULTISIGMODIFICATIONS
- MULTISIGRECEIVES
- MULTISIGSENDS
- MULTISIGSIGNATURES
- MULTISIGSIGNERMODIFICA
- MULTISIGTRANSACTIONS
- NAMESPACEPROVISIONS
- NAMESPACES
- TRANSFERREDMOSAICS
- TRANSFERS

```
SELECT * FROM ACCOUNTS
```

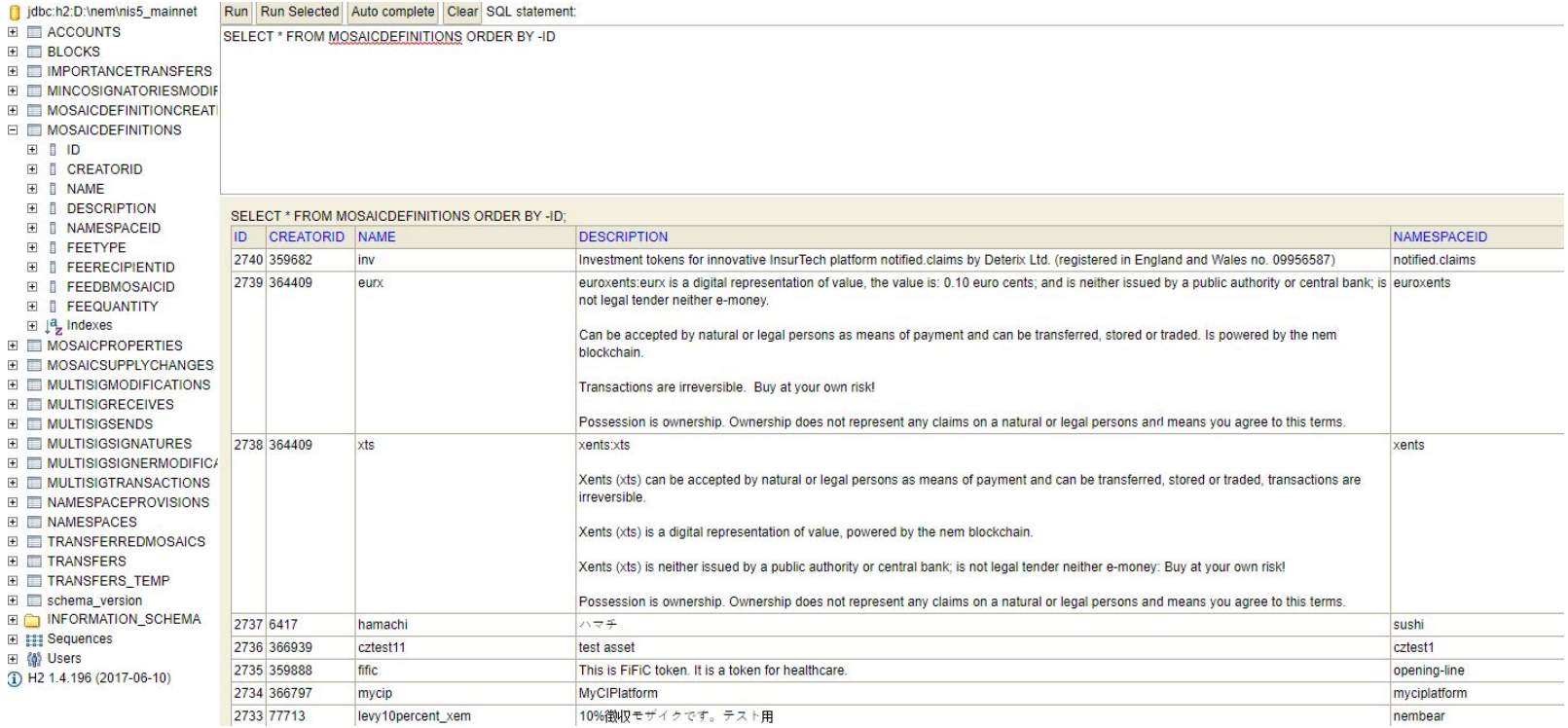
SELECT * FROM ACCOUNTS;

ID	PRINTABLEKEY	PUBLICKEY
1	NANEMOABLAGR72AZ2RV3V4ZHDCXW25XQ73O7OBT5	8d07f90fb4bbe7715fa327c926770166a11be2e494a970605f2e12557f66c9b9
2	NCMARKECQXP3SQZSJPCBKOQWIXRRI7LIS4FTU4VZ	f41b99320549741c5cce42d9e4bb836d98c50ed5415d0c3c2912d1bb50e6a0e5
3	NADMEHCFJD45GPTDL4HZP2LJLZVAZRLYWYPNEMLY	1fbdbdde28daf828245e4533765726f0b7790e0b7146e2ce205df3e86366980b
4	NCOSIGIMCITYNRE7XUR77OREOYFLPAILCNUYLTPD	826cedee421ff66e708858c17815fcd831a4bb68e3d8956299334e9e24380ba8
5	NCYP7E7S67LB4G4Y4RJWE7REFBZI4EXJ73MRKIWW	719862cd7d0f4e875a6a0274c9a1738f38f40ad9944179006a54c34724c1274d
6	NDKICABB2EXCLIQ6DHWL66CK7QPYUGLMDVTLUIEB	43aa69177018fc3e2bdbeb259c81cddf24be50eef9c5386db51d82386c41475a
7	NANIBAXPVLBP37YXSGREVD77NXIFZML5FDE7F3ZN	f94e8702eb1943b23570b1b83be1b81536df35538978820e98bfce8f999e2d37
8	NCOPERAWEWCD4A34NP5UQCCKEX44MW4SL3QYJYS5	6f4dc5fa7b531f56543a4b567fefea2fcf080be71af73aca9925968eeee2ad0f
9	NALICEXDMHEAVGSWHLSXDMUMCCHTBJLPKPI63IFY	dd7d3ac741ce2a757311f88d9cebf63200e934aebdd85917ce981c572312375f

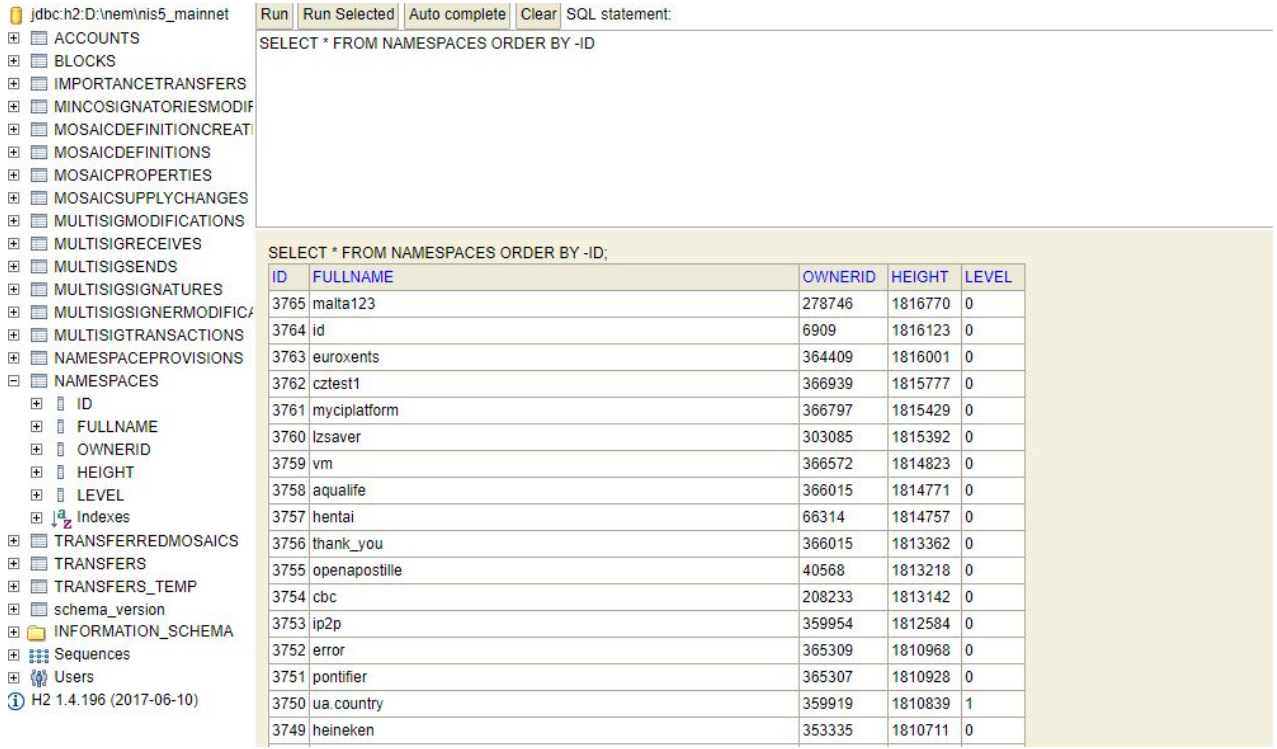
Блоки


jdbc:h2:D:\nem\nis5_mainnet
ACCOUNTS
BLOCKS
ID
VERSION
PREVBLOCKHASH
BLOCKHASH
GENERATIONHASH
TIMESTAMP
HARVESTERID
HARVESTERPROOF
HARVESTEDINNAME
HEIGHT
TOTALFEE
DIFFICULTY
Indexes
IMPORTANCETRANSFERS
MINCOSIGNATORIESMODIF
MOSAICDEFINITIONCREAT
MOSAICDEFINITIONS
MOSAICPROPERTIES
MOSAICSUPPLYCHANGES
MULTISIGMODIFICATIONS
MULTISIGRECEIVES
MULTISIGSENDS
MULTISIGSIGNATURES
MULTISIGSIGNERMODIFIC
MULTISIGTRANSACTIONS
NAMESPACEPROVISIONS
Run
Run Selected
Auto complete
Clear
SQL statement:
SELECT * FROM BLOCKS
SELECT * FROM BLOCKS;
ID VERSION PREVBLOCKHASH BLOCKHASH
1 1744830465 0000000000000000000000000000000000000000000000000000000000000000 438cf6375dab5a0d32f9b7bf151d4539e00a590f7c022d5572c7d41815a24be4
2 1744830465 438cf6375dab5a0d32f9b7bf151d4539e00a590f7c022d5572c7d41815a24be4 1dd9d4d7b6af603d29c082f9aa4e123f07d18154ddbcd7ddc6702491b854c5e4
3 1744830465 1dd9d4d7b6af603d29c082f9aa4e123f07d18154ddbcd7ddc6702491b854c5e4 9708256e8a8dfb76eed41dcfa2e47f4af520b7b3286afb7f60dca02851f8a53e
4 1744830465 9708256e8a8dfb76eed41dcfa2e47f4af520b7b3286afb7f60dca02851f8a53e e2ffcee2a5d966435f8299be471cdc08efff0e7deec047426e0b1d80caf4c6e8
5 1744830465 e2ffcee2a5d966435f8299be471cdc08efff0e7deec047426e0b1d80caf4c6e8 5b1a8f223969878bcbe4429960fa3b26b7523b919a513f0bd18e28a7c0f54575
6 1744830465 5b1a8f223969878bcbe4429960fa3b26b7523b919a513f0bd18e28a7c0f54575 1441955bb3bbc0f668ab40a26d63379044479c372f8f5bcd0f285cb421b0188b
7 1744830465 1441955bb3bbc0f668ab40a26d63379044479c372f8f5bcd0f285cb421b0188b baa09bb2b2157da83801a52719540c65063a1987134aa039a383c0343155c7d6
8 1744830465 baa09bb2b2157da83801a52719540c65063a1987134aa039a383c0343155c7d6 0b54956cb8790425f78a40dfc6bc35eda3b5b6e8c3154ab8e4a48285f1129b61
9 1744830465 0b54956cb8790425f78a40dfc6bc35eda3b5b6e8c3154ab8e4a48285f1129b61 faa84cc0d9dd088d85c0d1b0f039c3f7e87f0eb8277c46dc7a817ba841edf4e4
10 1744830465 faa84cc0d9dd088d85c0d1b0f039c3f7e87f0eb8277c46dc7a817ba841edf4e4 7b2c647509f7d14628a88ba93f75a5cee97f4fbcba9bd7866556519962dcf8a5
11 1744830465 7b2c647509f7d14628a88ba93f75a5cee97f4fbcba9bd7866556519962dcf8a5 61094263bf9ea8cc3cc4b70e81f7bc97d2833d4fc484d0e6c6da565a07626022
12 1744830465 61094263bf9ea8cc3cc4b70e81f7bc97d2833d4fc484d0e6c6da565a07626022 255feab22ae44519bcc4f1ec6e1e40fd3393bc90f53b0605f5c2059f4634856d
13 1744830465 255feab22ae44519bcc4f1ec6e1e40fd3393bc90f53b0605f5c2059f4634856d 0c7dedb3294d2deaec7dc8d72ffa17e722c8fba2683344011782dc7e67445a2d
14 1744830465 0c7dedb3294d2deaec7dc8d72ffa17e722c8fba2683344011782dc7e67445a2d 26efe1d2323439c3a07e948a611d4c5dd58d6e97eeb7aa880f8354ac0257650a

Определения мозаик

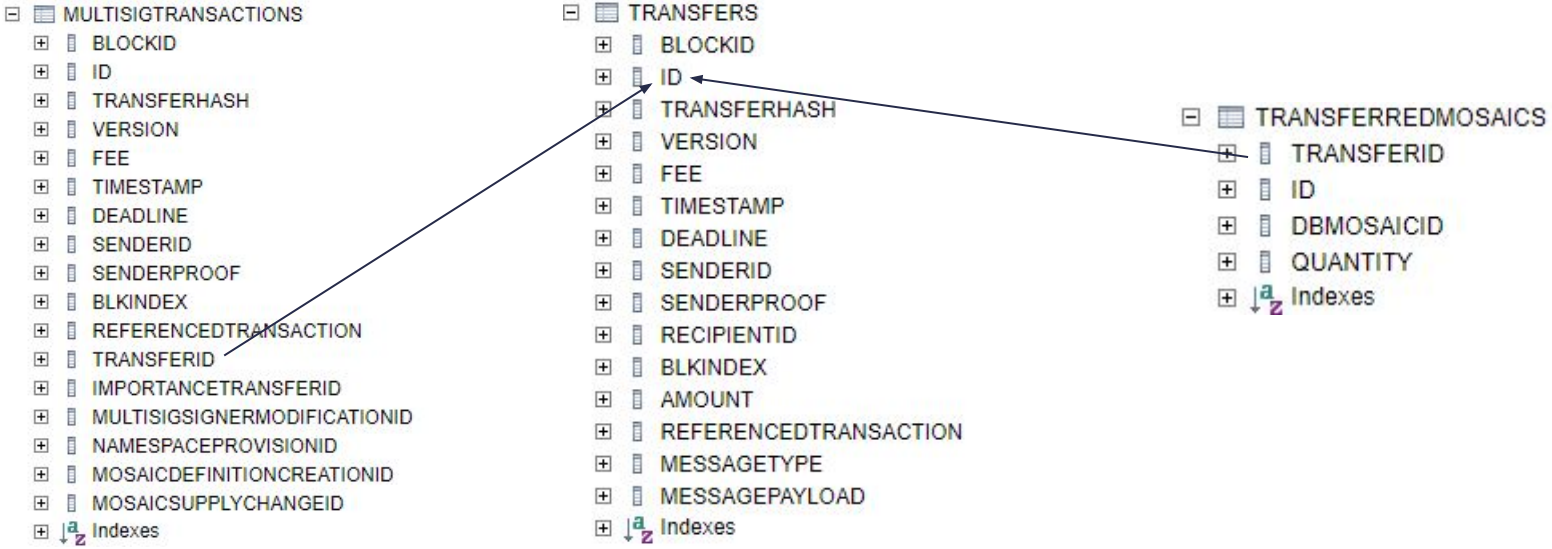


Просторанства имен



Транзакции

IMPORTANCETRANSFERS - передача важности (делегированный харвестинг)
MULTISIGTRANSACTIONS - мультиподписные транзакции
TRANSFERREDMOSAICS - передача мозаик
TRANSFERS - все транзакции, где есть получатель и отправитель



Пример 1

Получение списка всех аккаунтов когда-либо владевших ProximaX

1. Находим ID интересующей нас мозаики

SELECT * FROM MOSAICDEFINITIONS WHERE NAME='xpx';

ID	CREATORID	NAME	DESCRIPTION	NAMESPACEID	FEETYPE	FEERECIPIENTID	FEEDBMOSAICID	FEEQUANTITY
1605	191113	xpx	Official Proximax Mosaic - prx:xpx	prx	*null*	*null*	*null*	*null*

2. Собираем всех получателей соединяя таблицы TRANSFERS и TRANSFERREDMOSAICS

```
SELECT RECIPIENTID FROM TRANSFERS JOIN TRANSFERREDMOSAICS ON
TRANSFERREDMOSAICS.TRANSFERID=TRANSFERS.ID WHERE DBMOSAICID=1605
```

3. Соединяем полученное с таблицей ACCOUNTS, чтоб получить адреса

```
SELECT PRINTABLEKEY FROM TRANSFERS JOIN TRANSFERREDMOSAICS ON
TRANSFERREDMOSAICS.TRANSFERID=TRANSFERS.ID JOIN ACCOUNTS ON
ACCOUNTS.ID=TRANSFERS.RECIPIENTID WHERE DBMOSAICID=1605
```

Сохраняем в файл

Получение списка всех аккаунтов когда либо владевших ProximaX

```
CALL CSVWRITE('D:\xpx.txt','SELECT PRINTABLEKEY FROM TRANSFERS JOIN TRANSFERREDMOSAICS ON TRANSFERREDMOSAICS.TRANSFERID=TRANSFERS.ID JOIN ACCOUNTS ON ACCOUNTS.ID=TRANSFERS.RECIPIENTID WHERE DBMOSAICID=1605');
CSVWRITE('D:\xpx.txt', 'SELECT PRINTABLEKEY FROM TRANSFERS JOIN TRANSFERREDMOSAICS ON TRANSFERREDMOSAICS.TRANSFERID=TRANSFERS.ID JOIN ACCOUNTS ON ACCOUNTS.ID=TRANSFERS.RECIPIENTID WHERE DBMOSAICID=1605')
9323
(1 row, 170 ms)
```

```
"PRINTABLEKEY"
"NDJ74QHCGE4LV3Q7SQE2CN3SVLQ5X2QQHHRQNXPX"
"NBQXN7EJO47HBOY5LZHSW3XZZLO4C5BEKWN4MTS6"
"NDXMM5BH64U3PH4MYEMNKAC2YCTYVRZ7KZPXVOUF"
"NBQXN7EJO47HBOY5LZHSW3XZZLO4C5BEKWN4MTS6"
"NAEV6DH5V6MJB5MKEYO7EILINYTB3FHPPUFV4AQH"
"NBGYBNHDQYFBLP2QQCZHFMU3AOYIK5NBSFOQ636J"
"NCOQQRIKJCMKCOEISPDR37ZDA35JSHFEELEH7EVY"
"NAQ55SORFP3MTXLRQDOVSE32OQALCDNNGETE2I6J"
"NCMKQSD3EUU5OZVUD45RR6GKEMYUCAZ4CIDGUR6I"
"NCNR4TRV4ATJ2ESUI7LZZQ5CCWTMQZT3LJ25B4BG"
"NBHTRKUUXMDFYZB5WYZR37PN42LDX4YBPCI3GYIO"
"NBAJMGQRDYI27O63RYB7S5LHGA5X7MVQ34QLHVC5"
"NBXWMBP6TIJZYMHKNS3NM27KBU3CTIUOAQDMS43V"
"NA7RGKL5PK3KGQKPVXSGL4T2EFEGJMXED4NKXHGA"
"NDXNTF3WXVEV3OL4HX3SGCBS3UL7WTKIALX6I74H"
"NACUB6MB2P5SLFPMPC4ARAHY3GKZXOEC6CMMWWHJ"
"NA3NIFJO5VA65QCORIYU6YX7V2W7CEIB4ZKPZAXI"
"NCQLSQKG4TUQS62C7P2LHHHFQOQ5DSOMI5YZZJPP"
"NDCBKVR4ZNJNXFH3SLSH7UH2PV4EC5V2ECBZFTDW"
"NB37PHRDWPLFDI6UD4Q34UPUAF6MJEKRTS5O2JPP"
"NCZXL5YGL25ZQI37SV4NNCR7WPRF4O3UNC2GN4JY"
"ND4UJB55W2OZZ6ZIVFMA7HKJWAXFZ4BGLAEFKBZP"
"NCBGHHORJTGTG3EEQKLS2JCUHHQ6RFA5OH45Z5PN"
"NBFXMXKW4SA5UPWRCDHW7T6XGBWAIRJIMEEMASQD"
"NCILG74KNHVUKTU2GW5OHA7AQGM3AXM6XQELRPOM"
```

Пример 2

Сколько блоков собирают суперноды?

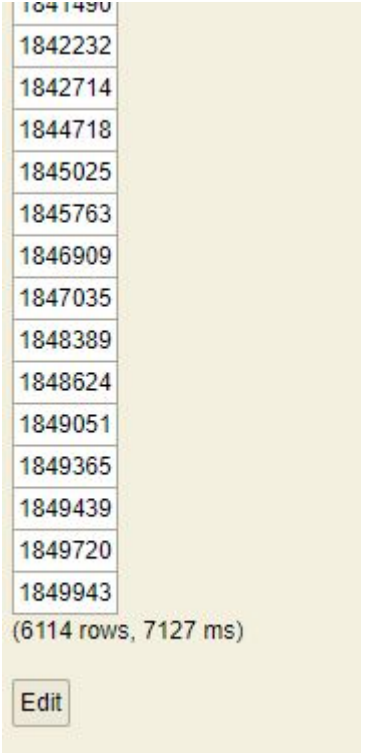
1. Найдем всех получателей SUPERNODE REWARDS программы за период от блока 1840000 до 1850000

```
SELECT DISTINCT RECIPIENTID FROM TRANSFERS WHERE
SENDERID='5203' AND BLOCKID>1840000 AND BLOCKID<1850000
```

2. Найдем ID всех блоков, которые они собрали

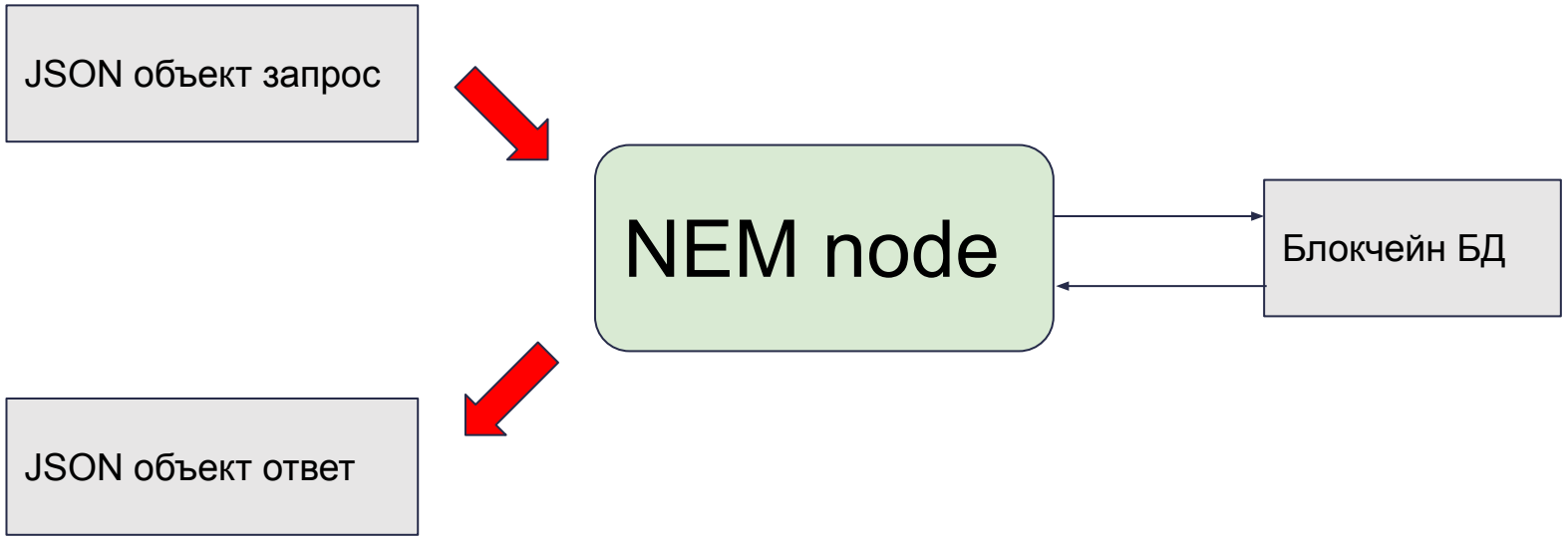
```
SELECT ID FROM BLOCKS WHERE ID>1840000 AND ID<1850000
AND BLOCKS.HARVESTEDINNAME IN (SELECT DISTINCT
RECIPIENTID FROM TRANSFERS WHERE SENDERID='5203' AND
BLOCKID>1840000 AND BLOCKID<1850000)
```

10000 блоков ≈ 1 неделя



NEM API

Быстрый старт



Announce (POST method)

/transaction/prepare-announce

- Private Key send to NIS
- NIS signs transaction

/transaction/announce (Highly Recommended)

- Signed with SDK/Wrapper
- Signed with Ed25519 library

Транзакции (структура JSON)

Time Stamp	Время в секундах от nemesis block.
Amount	Количество microXEM (1 microXEM = 1 XEM/1000000)
Signature	Подпись транзакции
Fee	Комиссия
Recipient	Адрес получателя
Type	Тип транзакции
Deadline	Время отмены транзакции если она не включена в блок.
Message	Объект сообщения
Version	Версия структуры
Signer	Public key инициатора транзакции

NEM API

Быстрый старт

7.2.1. Version 1 transfer transactions

Suppose you want to send 1000 NEM from sender account (referred hereafter as ***Alice***):

`TALICELCD3XPH4FFI5STGGNSNSWPOTG5E4DS2TOS`

to recipient account:

`TBOBBSXX7BESJXDWGLP5Z7FM5HSTKUH5WIMPW562`

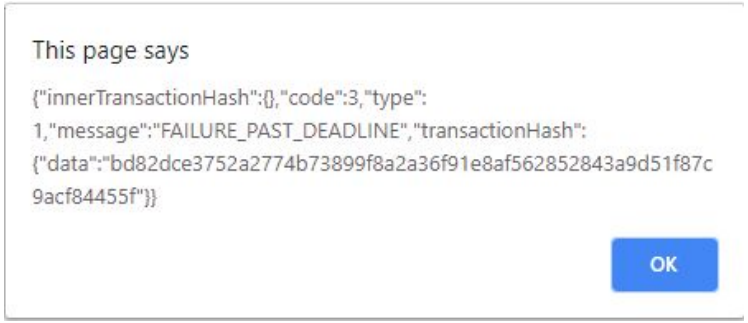
The RequestPrepareAnnounce JSON object you have to send to NIS via a POST request would look similar to this (test network):

```
{
        "transaction":
        {
            "timeStamp": 9111526,
            "amount": 1000000000,
            "fee": 3000000,
            "recipient": "TBOBBSXX7BESJXDWGLP5Z7FM5HSTKUH5WIMPW562",
            "type": 257,
            "deadline": 9154726,
            "message":
            {
                "payload": "",
                "type": 1
            },
            "version": -1744830463,
            "signer": "a1aaca6c17a24252e674d155713cdf55996ad00175be4af02a20c67b59f9fe8a"
        },
        "privateKey": "00983bb01d05edecfaef55df9486c111abb6299c754a002069b1d0ef4537441bda"
}
```

https://nemproject.github.io/

NEM API

```
<script>

// Объект транзакции из примера
var transaction=
{
        "transaction":
        {
            "timeStamp": 9111526,
            "amount": 1000000000,
            "fee": 3000000,
            "recipient": "TBOBBSXX7BESJXDWGLP5Z7FM5HSTKUH5WIMPW562",
            "type": 257,
            "deadline": 9154726,
            "message":
            {
                "payload": "",
                "type": 1
            },
            "version": -1744830463,
            "signer": "a1aaca6c17a24252e674d155713cdf55996ad00175be4af02a20c67b59f9fe8a"
        },
        "privateKey": "00983bb01d05edecfaef55df9486c111abb6299c754a002069b1d0ef4537441bda"
}

var apiurl="http://localhost:7890/transaction/prepare-announce"; // API путь для анонса транзакции
var xhr=new XMLHttpRequest();
xhr.open("POST", apiurl);
xhr.setRequestHeader("Content-Type", "application/json");
xhr.send(JSON.stringify(transaction)); // Отправка JSON объекта методом POST
xhr.onreadystatechange = function ()
{
    if (xhr.readyState === 4 && xhr.status === 200) {
        var json = JSON.parse(xhr.responseText);
        alert(JSON.stringify(json));
    }
    console.log(xhr.responseText);
};


</script>
```

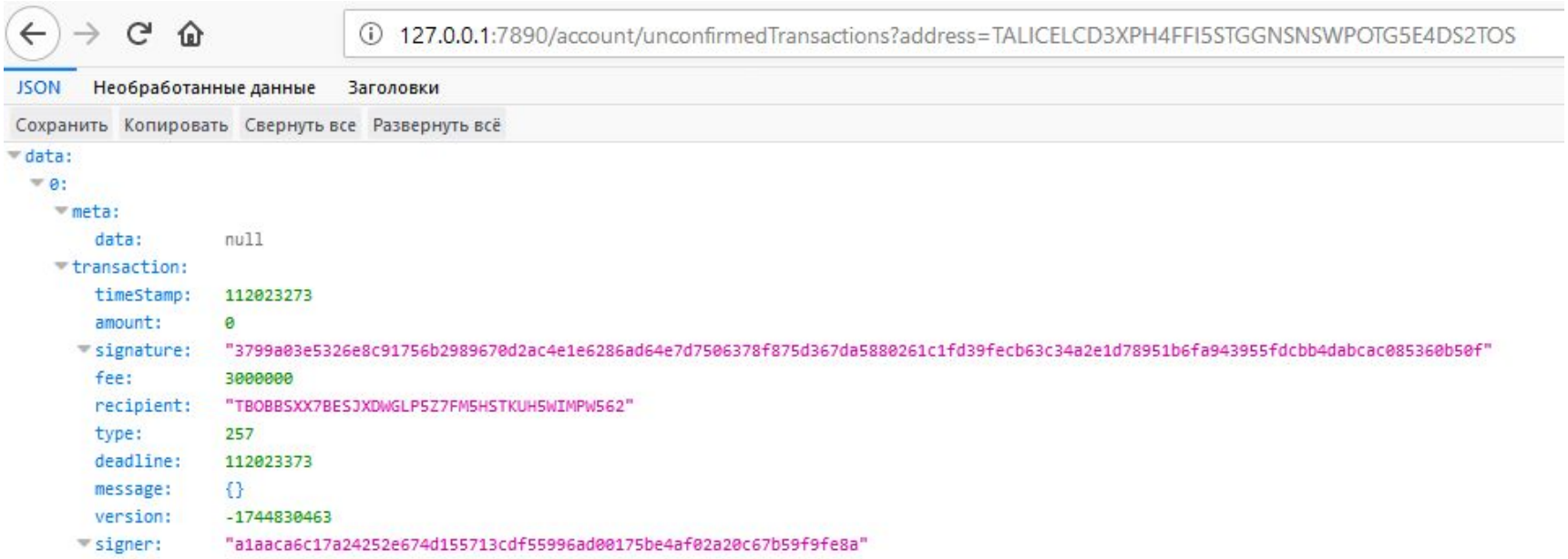


NEM API

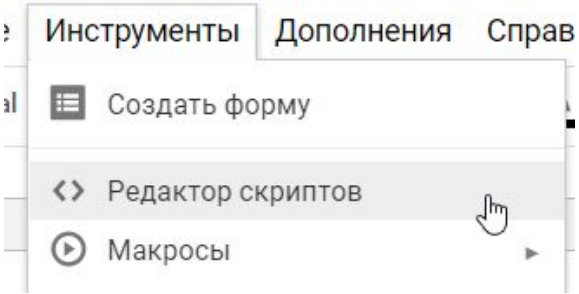
```
<script>

function nemtimestamp() //Время в секундах от начала эпохи NEM
{
var now = new Date();
var nemepoch = new Date('March 29, 2015 00:06:25 -0000');
return Math.round((now-nemepoch)/1000);
}

// Объект транзакции из примера
var transaction=
{
        "transaction":
        {
            "timeStamp": nemtimestamp(),
            "amount": 0,
            "fee": nemtimestamp(),
            "recipient": "TBOBBSXX7BESJXDWGLP5Z7FM5HSTKUH5WIMPW562",
            "type": 257,
            "deadline": nemtimestamp()+100,
            "message":
            {
                "payload": "",
                "type": 1
            },
            "version": -1744830463,
            "signer": "a1aaca6c17a24252e674d155713cdf55996ad00175be4af02a20c67b59f9fe8a"
        },
        "privateKey": "00983bb01d05edecfaef55df9486c111abb6299c754a002069b1d0ef4537441bda"
}

var apiurl="http://localhost:7890/transaction/prepare-announce"; // API путь для анонса транзакции
var xhr=new XMLHttpRequest();
xhr.open("POST", apiurl);
xhr.setRequestHeader("Content-Type", "application/json");
xhr.send(JSON.stringify(transaction)); // Отправка JSON объекта методом POST
xhr.onreadystatechange = function ()
{
    if (xhr.readyState === 4 && xhr.status === 200) {
        var json = JSON.parse(xhr.responseText);
        alert(JSON.stringify(json));
    }
    console.log(xhr.responseText);
};


</script>
```

```
function nemtimestamp()
{
var now = new Date();
var nemepoch = new Date(“March 29, 2015
00:06:25 -0000”);
return Math.round((now-nemepoch)/1000);
}
```

NEM API


127.0.0.1:7890/account/unconfirmedTransactions?address=TALICELCD3XPH4FFI5STGGNSNSWPOTG5E4DS2TOS
JSON
Необработанные данные
Заголовки
Сохранить
Копировать
Свернуть все
Развернуть всё
data:
0:
meta:
data: null
transaction:
timeStamp: 112023273
amount: 0
signature: "3799a03e5326e8c91756b2989670d2ac4e1e6286ad64e7d7506378f875d367da5880261c1fd39fecb63c34a2e1d78951b6fa943955fdcbb4dabcac085360b50f"
fee: 3000000
recipient: "TBOBBSXX7BESJXDWGLP5Z7FM5HSTKUH5WIMPW562"
type: 257
deadline: 112023373
message: {}
version: -1744830463
signer: "a1aaca6c17a24252e674d155713cdf55996ad00175be4af02a20c67b59f9fe8a"

NEM API in Google Sheets




```
function getNEMbalance(nemadr) {

 var address=nemadr.split("-").join("");
 var url = 'http://hugealice.nem.ninja:7890/account/get?address='+address;
 var response = UrlFetchApp.fetch(url, {'muteHttpExceptions': true});
 var json = response.getContentText();
 var data = JSON.parse(json);
  if (data.error)
  {
   return 0;
  }
  else return data.account.balance/1000000;
}
```

Код.gs

```
1  function getNEMbalance(nemadr) {
2
3   var address=nemadr.split("-").join("");
4   var url = 'http://hugealice.nem.ninja:7890/account/get?address='+address;
5   var response = UrlFetchApp.fetch(url, {'muteHttpExceptions': true});
6   //Logger.log(response)
7   var json = response.getContentText();
8   var data = JSON.parse(json);
9    if (data.error)
10   {
11    return 0;
12   }
13   else return data.account.balance/1000000;
14 }
```

NEM API in Google Sheets

fx =getNEMbalance(A1)

	A	B	C
1	NDJ74QHCGE4LV3Q7SQE2CN3SVLQ5X2QQHHRQNXPX	=getNEMbalance(A1)	
2	NBQXN7EJO47HBOY5LZHSW3XZZLO4C5BEKWN4MTS6	1058,8	
3	NDXMM5BH64U3PH4MYEMNKAC2YCTYVRZ7KZPXVOUF	836,477	
4	NBQXN7EJO47HBOY5LZHSW3XZZLO4C5BEKWN4MTS6	1058,8	
5	NAEV6DH5V6MJB5MKEYO7EILINYTB3FHPPUFV4AQH	0	
6	NBGYBNHDQYFBLP2QQCZHFMU3AOYIK5NBSFOQ636J	0	
7	NCOQQRIKJCMKCOEISPDR37ZDA35JSHFEELEH7EVY	160,4	
8	NAQ55SORFP3MTXLRQDOVSE32OQALCDNNGETE2I6J	4316,012767	
9	NCMKQSD3EUU5OZVUD45RR6GKEMYUCAZ4CIDGUR6T	2534,872713	
10	NCNR4TRV4ATJ2ESUI7LZZQ5CCWTMQZT3LJ25B4BG	0	
11	NBHTRKUUXMDFYZB5WYZR37PN42LDX4YBPCI3GYIO	0,95	
12	NBAJMGQRDYI27O63RYB7S5LHGA5X7MVQ34QLHVC5	32,029205	
13	NBXWMBP6TIJZYMHKNS3NM27KBU3CTIUOAQDMS43V	3994,487321	
14	NA7RGKL5PK3KGQKPVXSGL4T2EFEGJMXED4NKXHGA	266,794228	
15	NDXNTF3WXVEV3OL4HX3SGCBS3UL7WTKIALX6I74H	10945,16987	
16	NACUB6MB2P5SLFPMPC4ARAHY3GKZXOEC6CMMWWHJ	0	
17	NA3NIFJO5VA65QCORIYU6YX7V2W7CEIB4ZKPZAXI	10	
18	NCQLSQKG4TUQS62C7P2LHHHFQOQ5DSOMI5YZZJPP	2359,583691	
19	NDCBKVR4ZNJNXFH3SLSH7UH2PV4EC5V2ECBZFTDW	1,556189	

Спецификация всех объектов

https://nemproject.github.io/#appendix-A:-description-of-the-JSON-structures

9. Appendix A: Description of the JSON Structures

9.1. AccountHistoricalDataViewModel

Description:

Nodes can support a feature for retrieving historical data of accounts. If a node supports this feature, it will return an array of AccountHistoricalDataViewModel objects.

JSON structure by example:

```
{
	"height": 8976,
	"address": "NALICELGU3IVY4DPJKHYLSSVYFFWYS5QPLYEZDJJ",
	"balance": 80670000000,
	"vestedBalance": 13949175142,
	"unvestedBalance": 66720824858,
	"importance": 0.000081667608466172 21,
	"pageRank": 0.0006944567083595363
}
```

Description of the fields:

`height`	The height for which the data is valid.
`address`	The address of the account.
`balance`	The balance of the account.
`vestedBalance`	The vested part of the balance.
`unvestedBalance`	The unvested part of the balance.
`importance`	The importance of the account.
`pageRank`	The page rank part of the importance.

Ошибки

https://nemproject.github.io/#appendix-B:-NIS-errors

10. Appendix B: NIS Errors

In case NIS encounters an error while processing a request it returns a JSON error object whose structure is described in *Appendix A:* Error object . This chapter describes the errors messages that can be returned from NIS.

FAILURE_NEMESIS_ACCOUNT_TRANSACTION_AFTER_NEMESIS_BLOCK
The supplied transaction has the nemesis account as sender and cannot be included in a normal block.

Address	NANEMOABLAGR72AZ2RV3V4ZHDCXW25XQ73O7OBT5
Public key	8d07f90fb4bbe7715fa327c926770166a11be2e494a970605f2e12557f66c9b9
Balance	2,000,263.01

Nemesis аккаунт может быть использован для Proof of burn

Websocket API

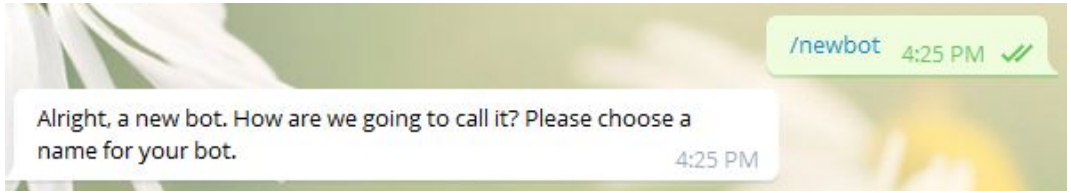
https://github.com/QuantumMechanics/nem-lightwallet/tree/master/lightwallet

localhost:7778/w/messages/websocket
localhost:7778/w/api/account/subscribe

Channels:

- `/errors` - should always subscribe to errors channel, any problems will be sent there
- `/account/<address>` - account information, this should be send whenever account state has changed
- `/recenttransactions/<address>` - (multiple) most recent transactions for the account
- `/transactions/<address>` - (single) confirmed tx, related to given address
- `/unconfirmed/<address>` - (single) unconfirmed tx, related to given address
- `/account/namespace/owned/<address>` - (single) namespace owned by given account
- `/account/mosaic/owned/<address>` - (single) mosaics owned by given account
- `/account/mosaic/owned/definition/<address>` - (single) definition of mosaic owned by given account
- `/unconfirmed` - every unconfirmed tx, that nis receives
- `/blocks/new` - height of a newly obtained chain fragment (this can be lower than current, height if there has been rollback)
- `/blocks` - full blocks

Telegram bot



@NEM_Ukraine_bot

Telegram bot

https://www.npmjs.com/package/stompjs
https://www.npmjs.com/package/telegram-node-bot

```
npm install --save telegram-node-bot
npm install stompjs
```

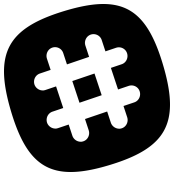
Telegram bot code 1

```
'use strict'

const Telegram = require('telegram-node-bot')
const TelegramBaseController = Telegram.TelegramBaseController
const TextCommand = Telegram.TextCommand
const tg = new Telegram.Telegram('TOKEN',{workers: 1}) // Токен получаем у @BotFather
var stomp=require('stompjs');

tg.router
    .when(
        new TextCommand('start', 'startCommand'),
        new StartController()
    )
```

Telegram bot code 2

```
class StartController extends TelegramBaseController {

    startHandler($) {

            $.sendMessage('Hi! I am bot connected to websocket and i Just tell you NEM blockchain height.'); \\ Бот представляется и даем задание
            $.sendMessage('But also I have a task for you: Find the block in NEM blockchain which has maximal commission ever, then reverse hash of this block and you get a private key for
address containing 111 xem for register your own namespace and token. Good luck!');

            var stompClient = stomp.overWS('ws://localhost:7778/w/messages/websocket'); \\Подключение к сокету

            var stcallback=function(frame){
                stompClient.subscribe('/blocks/new', function(data) { \\Подписываемся на событие
                var blockHeight = JSON.parse(data.body);
                $.sendMessage("new block arrived and height now is " + blockHeight.height); \\Отправляем сообщение
                console.log(blockHeight);
                                                                    })
            }

                var stompFailureCallback = function (error) {
                console.log('STOMP: ' + error);
                stompClient = stomp.overWS('ws://localhost:7778/w/messages/websocket');
                setTimeout(stompClient.connect({}, stcallback , stompFailureCallback), 10000);
                console.log('STOMP: Reconecting in 10 seconds');
                }

        if (!stompClient.connected) stompClient.connect({}, stcallback , stompFailureCallback);

    }

    get routes() {
        return {
            'startCommand': 'startHandler'
        }
    }
}
```

Connect with the NEM Development Community



SLACK

Join #general discussion

STACKOVERFLOW

Ask development questions

TWITTER

Follow the dev team **@nemtechdev**

TELEGRAM

Discover projects **@nemprojects**

Like, Subscribe, and Follow!

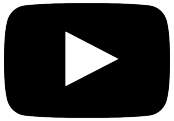
FACEBOOK
@ourNEM

TWITTER
@NEMofficial

REDDIT
/r/nem

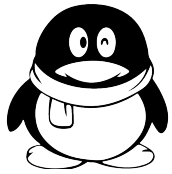
INSTAGRAM
@nem.io

MEDIUM
/nemofficial



YOUTUBE
/c/NEMofficial

WECHAT
nemxem



QQ
121037425

Join the Conversation

@nemred

For topics related to NEM

@nemberia

For general and price talks

@nemhelpdesk

For tech support and queries

See more: **http://bit.ly/nem-telegram-channels**