Гайд новичка ZHCASH

План, что и говорить, был превосходный:
простой и ясный, лучше не придумать.
Недостаток у него был только один:
было совершенно неизвестно,
как привести его в исполнение.

Алиса в стране чудес

Если бы у меня был этот гайд,
то я бы закончил смарт на месяц раньше

Автор

Определения:

1) zh – блокчейн 5го поколения ZHCASH. Результат скрещивания (любви) биткоина и эфира, прошедший генные модификации https://zh.cash/ . Произносится как «зх». Является противоположностью «хз».

2) Шекель — основная монета ZHC блокчейна ZHCASH.

3) Закинь шекелей — перечисли на мой кошелек ZHC

4) Закинь лифта – перечисли на мой кошель токенов LIFT

5) ZRC20 – аналог стандартов ERC20 и QRC20 (QTUM)

7) QTUM – родитель zh https://qtum.org/en. Старше на 2 года. Был взят самый перспективный блокчейн в мире (у китайцев) на момент 2019 года и существенно улучшен до лучшего в мире zh. Есть хорошая поддержка в телеграмме, где вам быстро ответят на любой вопрос.

8) Консоль – командная строка в терминале кошелька zh или программа с интерфейсом в командной строке zerohour-cli. Это не сайт https://zhcash.org/. Используется для ввода команд и взаимодействия с блокчейном. API есть на сайте https://zh.cash/docs/en/ZHCash-RPC-API/

9) hex — реальный адрес кошелька в HEX формате. Именно по нему происходит начисление токенов и шекелей при взаимодействии через смарт контракт. Получается при введении команды
`gethexaddress ZEFnGiHuwdSthnBA3cvAgPPFhhAKKqXQna`
в консоль. Получится `184eb41e30b0d5974df3d1b2429fbdf728222a4c`

Это почти эфировский адрес, за исключением того, что перед ним не стоит 0x. Использовать в коде смарта нельзя (не компилируется), при добавлении 0x в начале не воспринимается в zh как кошелек. Пользоваться исключительно таким видом кошелька (без 0x в начале), но не применять непосредственно в самом коде смарта.

10) Ремикс – эфировская среда для разработки смарта https://remix.ethereum.org/

12) Битый — смарт (или тразакция), которому не хватило газа и он не встроился в блокчейн. Отображается черным цветом в эксплорере (зероскан). За одного битого двух небитых дают или же наоборот.

12) Эксплорер (зероскан) - https://zeroscan.io/

13) Смарт — смартконтракт на solidity

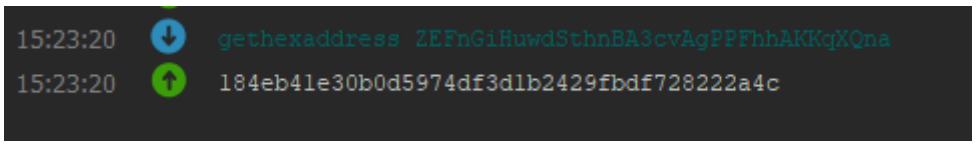
14) Начилить (начил)— получить нативные монеты из тестовой сети, проценты по токенам или получение вознаграждения за майнинг блока. Главное условие чтобы это происходило на расслабоне

Разработка смарта под zh

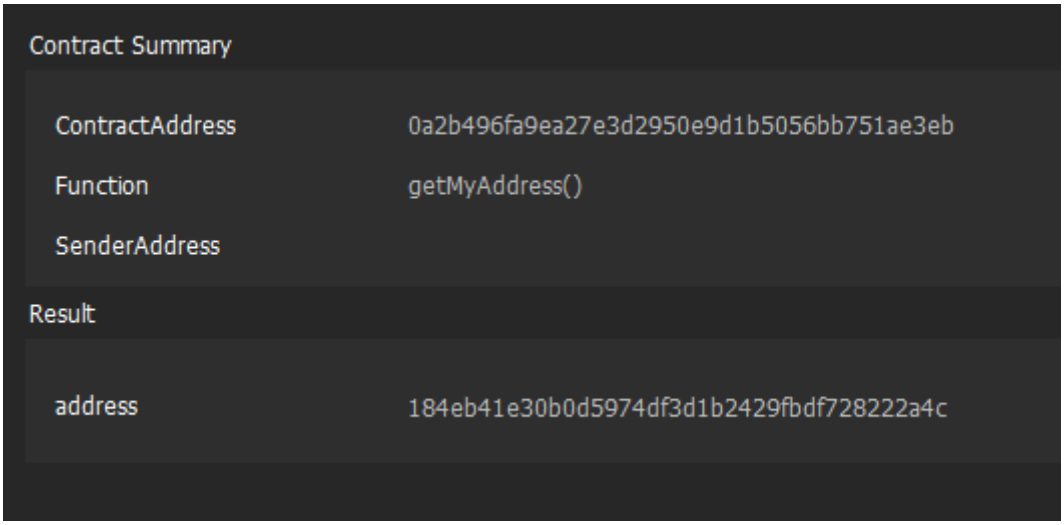
Предполагается, что читатель уже имеет начальный уровень знаний по solidity, который он может почерпнуть например здесь https://inaword.ru/smart-kontrakty/ или https://www.tutorialspoint.com/solidity/solidity_variables.htm

Система zh идентична Etherium, но есть некоторые ньюансы. Адрес в zh это тот же эфировский адрес, но без 0x. Но такой тип адреса нельзя указать непосредственно в коде ремикса, поэтому если мы хотим передать значение адреса (Например, первого пользователю, которому будет начислен миллион токенов), то делать мы это можем при создании контракта. При это прописывать адрес следует в формате hex

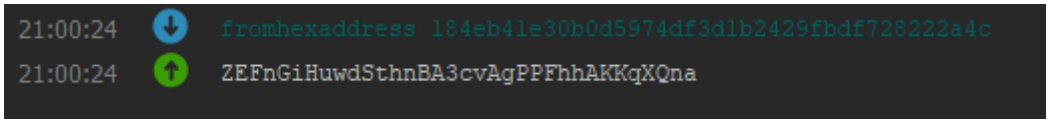
Получить hex можно через консоль gethexaddress (приглядывайтесь к первой строчке на скрине ниже)



Воспользовавшись функцией моего смарта получения своего адреса кошелька, заметим, что он будет также в hex формате

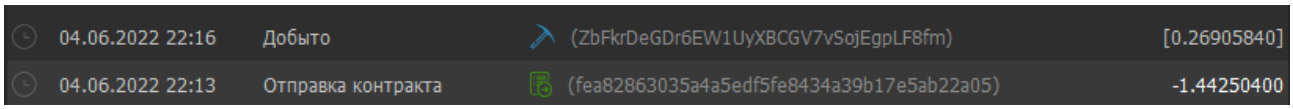


Можно перегнать hex в классический вид командой fromhexaddress (приглядывайтесь к первой строчке на скрине ниже)

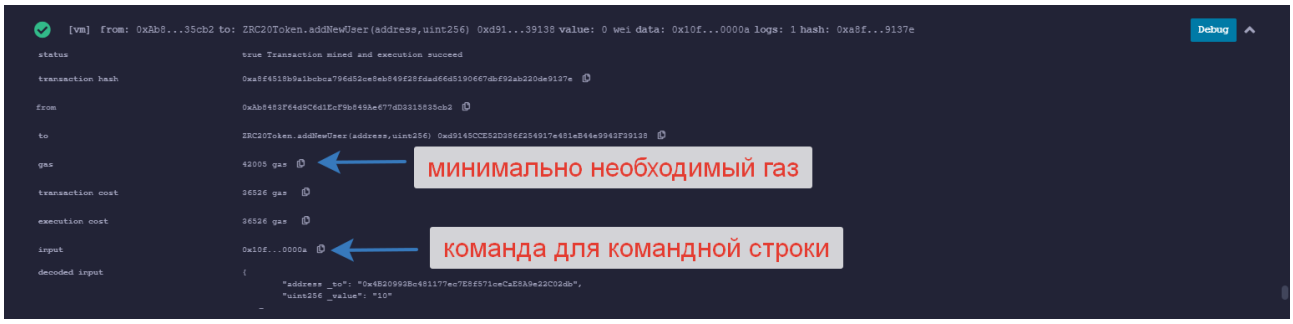


При aidrop не стоит засовывать в один блок больше 50 транзакций. Иначе тратится вся сумма на балансе. 20 транзакций безопасно.

Если контракт оказывается битым (после его выгрузки в транзакциях нет значка «добыто» и на зероскане в блоке смарт будет отображаться черным), то следует увеличить газ. Ниже показаны нормальные ситуации.



Узнать необходимый газ можно в ремиксе, развернув данные о транзакции. Но при попытке послать транзакцию, указанную на картинке ниже, с газом 50000 она не прошла, но с газом 100000 прошла. Рекомендуемый газ для любых транзакций в zh 250000, в qtum 100000.



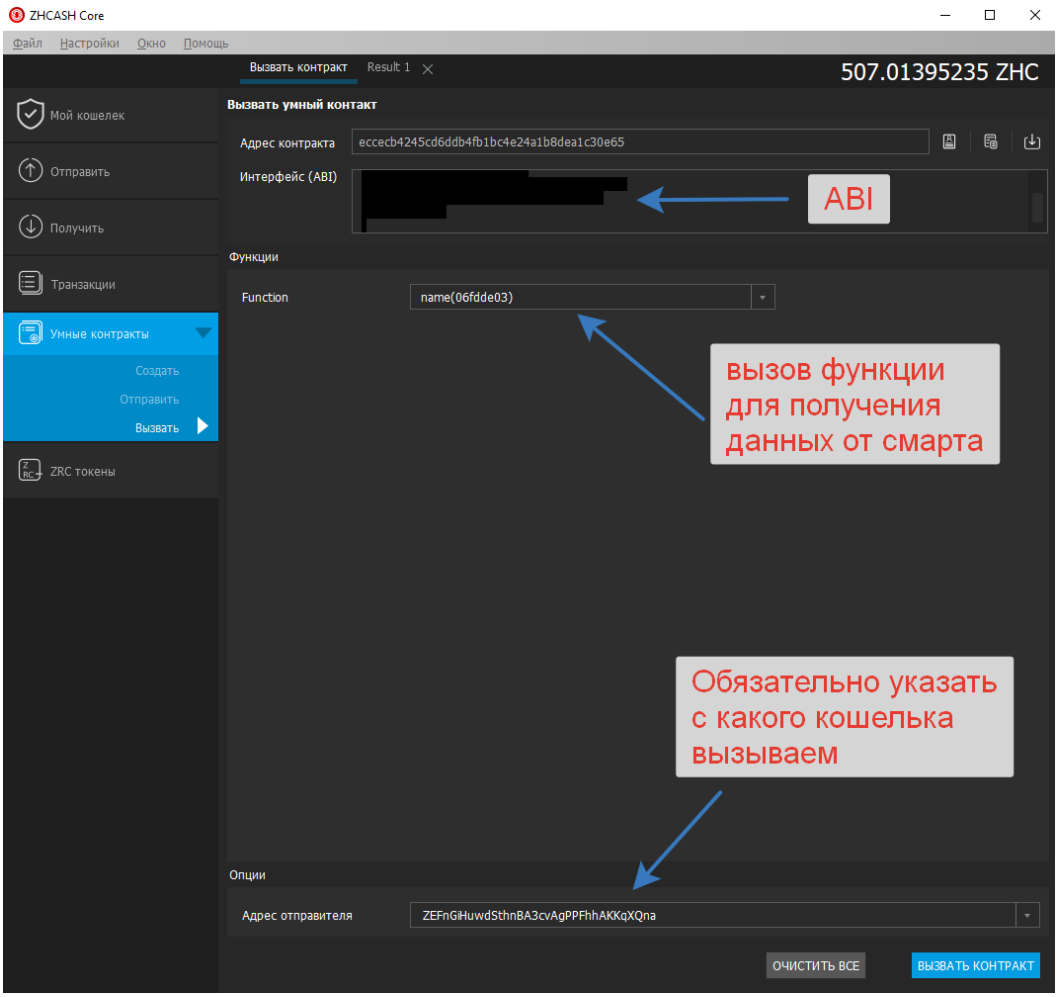
Также там можно узнать какая именно команда (код команды) при этом выполняется. Это необходимо для взаимодействия с блокчейном через консоль.

Zh поддерживает последнюю версию солидити, но рекомендуется использовать 0.7 версии. В первых контрактах я использовал версию 0.4.18, как в стандартном примере qtum QRC20 Token https://docs.qtum.site/en/QRC20-Token-Introduce.html , потому что не изменял стандартное значение газа. А при компиляции на версиях 0.7 даже стандартный пример смарта бьётся если оставить газ 250000.

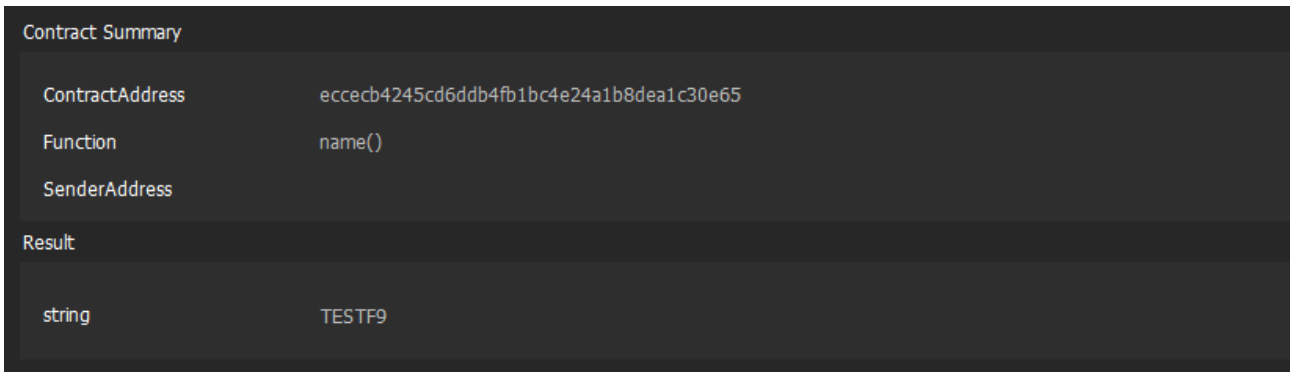
Загружать смарты и отсылать данные (sendtocontract) на смарт рекомендуется с кошелька, на котором баланс не больше 100 шекелей, т.к. велика вероятность проёба всего баланса. Автор неоднократно с этим сталкивался. Данное правило не касается вызова функций контракта (callcontract) и единичных отправок

токенов кому либо. Было замечено, что при попытке впихнуть более 30 транзакций в один блок с одного кошелька начинается сильное списание шекелей с баланса (от 1 тыс до 400 тыс). Так на airdrop можно слить весь баланс ноды (1 млн и больше влёгкую), разослав токены 60 пользователям в одном блоке. В итоге автор сделал airdrop рассылку 20 пользователям в одном блоке (с задержкой в 10 минут). В час рассылается 200 пользователям, что приемлемо. За 4 часа airdrop начилился всем. Всё таки есть способы обойти это ограничение и начилить токены на тысячи адресов мгновенно, но оставим это упражнение читателю.

Рассмотрим как осуществлять обмен информацией со смартом. Разберем сначала как взаимодействовать со смартом через графический интерфейс кошелька. Затем рассмотрим как это сделать через командную строку.

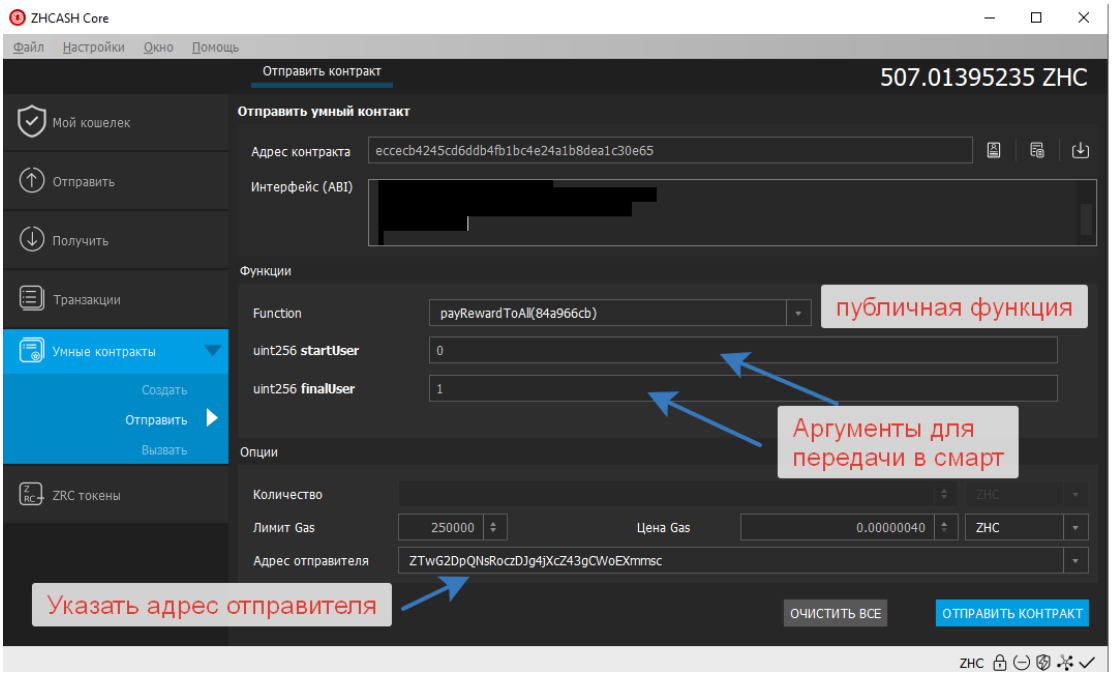


Если мы хотим получить данные, то следует использовать вкладку «Вызвать». Получим следующий результат.

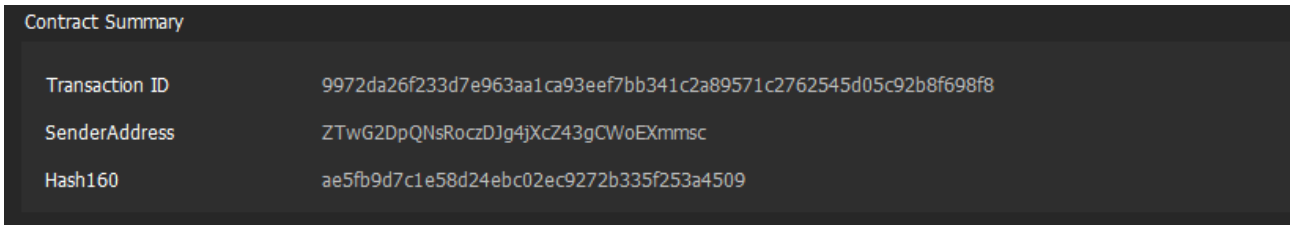


Так мы можем получим любое значение публичной переменной или выполнение внешней (external view) функции.

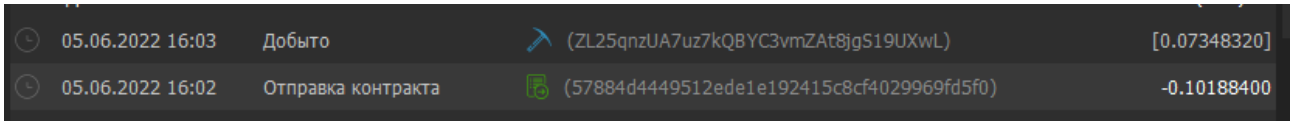
Если мы хотим отправить данные, то следует использовать вкладку «Отправить»



Если мы хотим выплатить вознаграждение всем пользователям, то следует также выплачивать порциями по 20 транзакций в блоке. И поставить большее значение газа. После такого обращения получим следующее.



Подождав новый блок и зайдя в «Транзакции» заметим, что появится значок «Добыто»



Это значит, что отправка данных успешна.

Теперь рассмотрим взаимодействие с блокчейном через консоль.

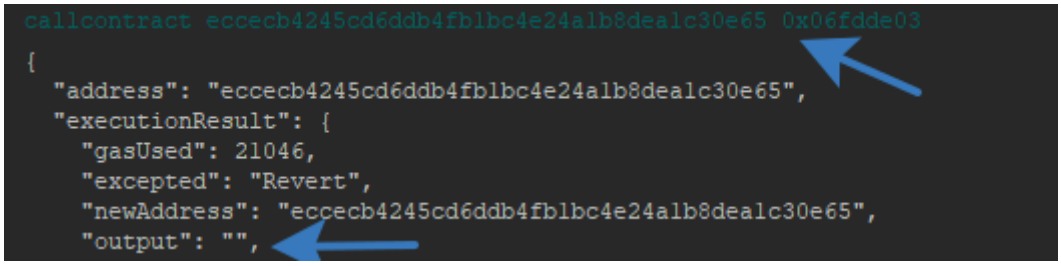
Для этого есть две команды: callcontract для получения данных и sendtocontract для отправки данных в смарт. Ниже приведены пример использвания.

```
sendtocontract efa851007505513c91c0f80e3e9f571a544d5245
a9059cbb000000000000000000000001c96419179e5d6e2cb98b0b2c
a658daff6d12cb50000000000000000000000000000000000000000
0000000000003b9aca00 0 100000 0.0000004
ZTwG2DpQNsRoczDJg4jXcZ43gCWoEXmmsc

callcontract eccecb4245cd6ddb4fb1bc4e24a1b8dea1c30e65
06fdde03
```

Команду для транзакции через консоль можно получить из ремикса

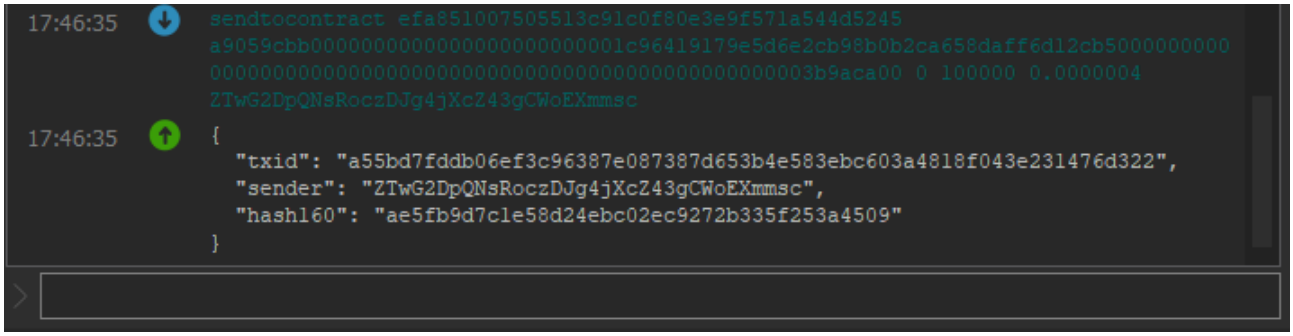
В названии команды не следует писать 0x. В zh это неправильно. Так вы ничего не получите. Следует писать то, что идет после 0x.



Вот так будет правильно:

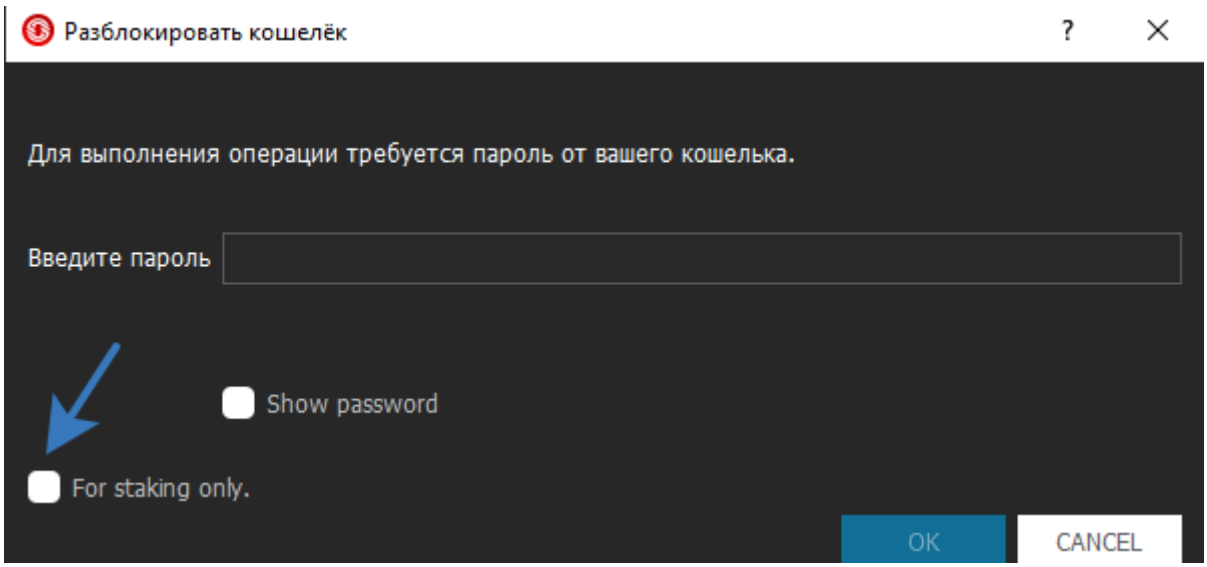
```
callcontract eccecb4245cd6ddb4fblbc4e24alb8dealc30e65 06fdde03
{
  "address": "eccecb4245cd6ddb4fblbc4e24alb8dealc30e65",
  "executionResult": {
    "gasUsed": 21891,
    "excepted": "None",
    "newAddress": "eccecb4245cd6ddb4fblbc4e24alb8dealc30e65",
    "output":
"0000000000000000000000000000000000000000000000000000000000000020000000000000000000
0000000000000000000000000000000000000000000000000654455354463900000000000000000000000000
0000000000000000000000000000",
```

Мы получим какие то данные, которые затем можно перегнать в строку. Ниже показан скриншот для sendtocontract

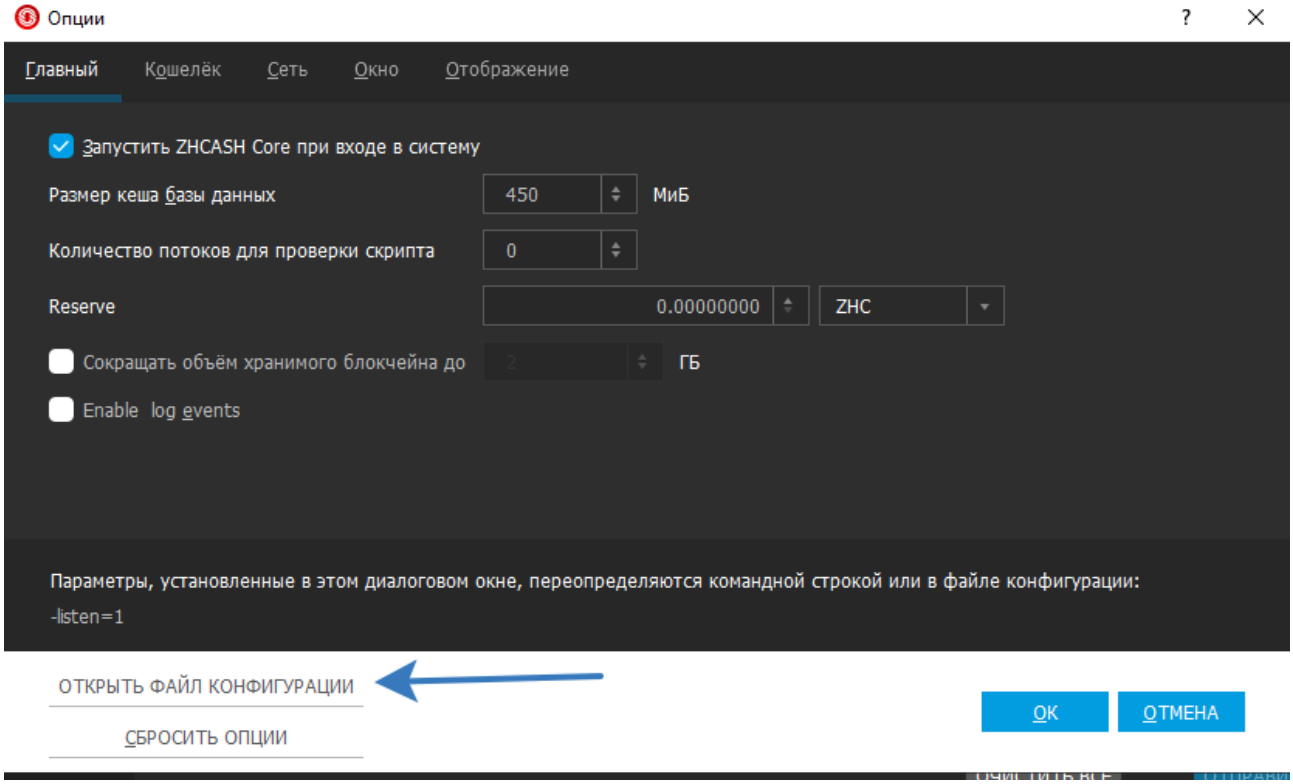


Для того, чтобы взаимодействовать через командную строку cmd нужно скачать консольную (серверную) версию кошелька

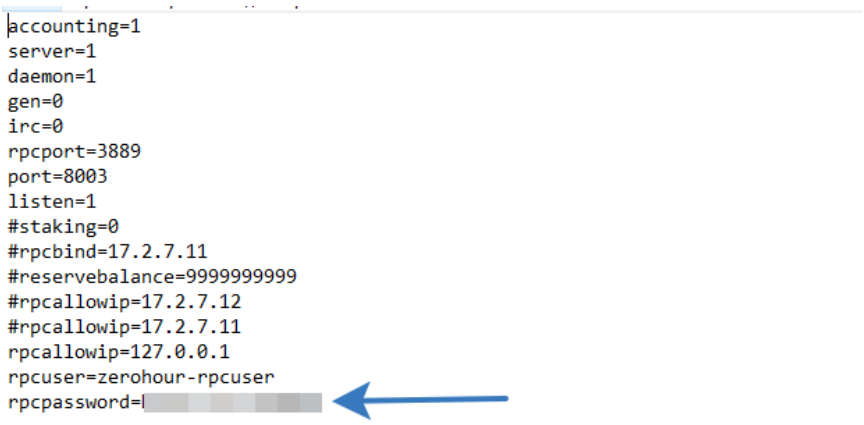
Затем разблокировать кошелек и снять галочку “For staking only”



Добавить файл конфигурации в параметрах

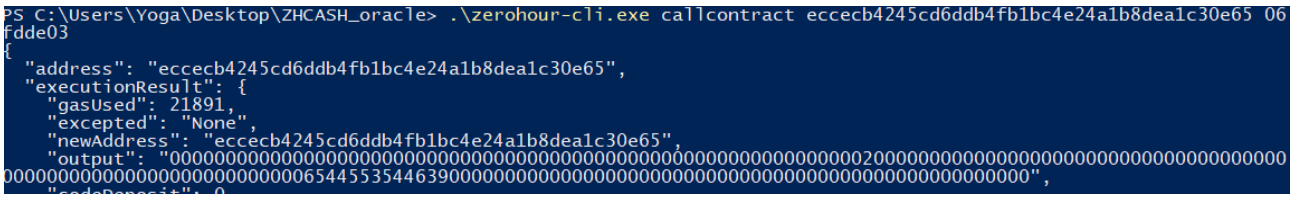


И сохранить следующий текст, где в последнем аргументе rpcpassword установить свой пароль от кошелька



```
accounting=1
server=1
daemon=1
gen=0
irc=0
rpcport=3889
port=8003
listen=1
#staking=0
#rpcbind=17.2.7.11
#reservebalance=9999999999
#rpcallowip=17.2.7.12
#rpcallowip=17.2.7.11
rpcallowip=127.0.0.1
rpcuser=zerohour-rpcuser
rpcpassword=1123581321
```

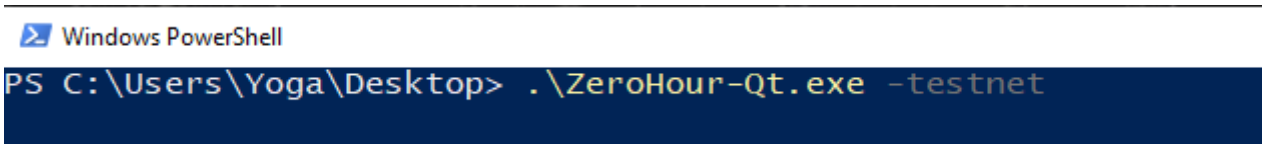
После этих манипуляций можно взаимодействовать с блокчейном через командную строку, что позволяет писать скрипты на питоне для автоматизации каких то действий с блокчейном. Например, для организации airdrop токенов, отправки транзакций, создание миллиона тестовых токенов, спама, хакинга и тому подобного. Ниже показано как можно вызвать функцию callcontract через cmd



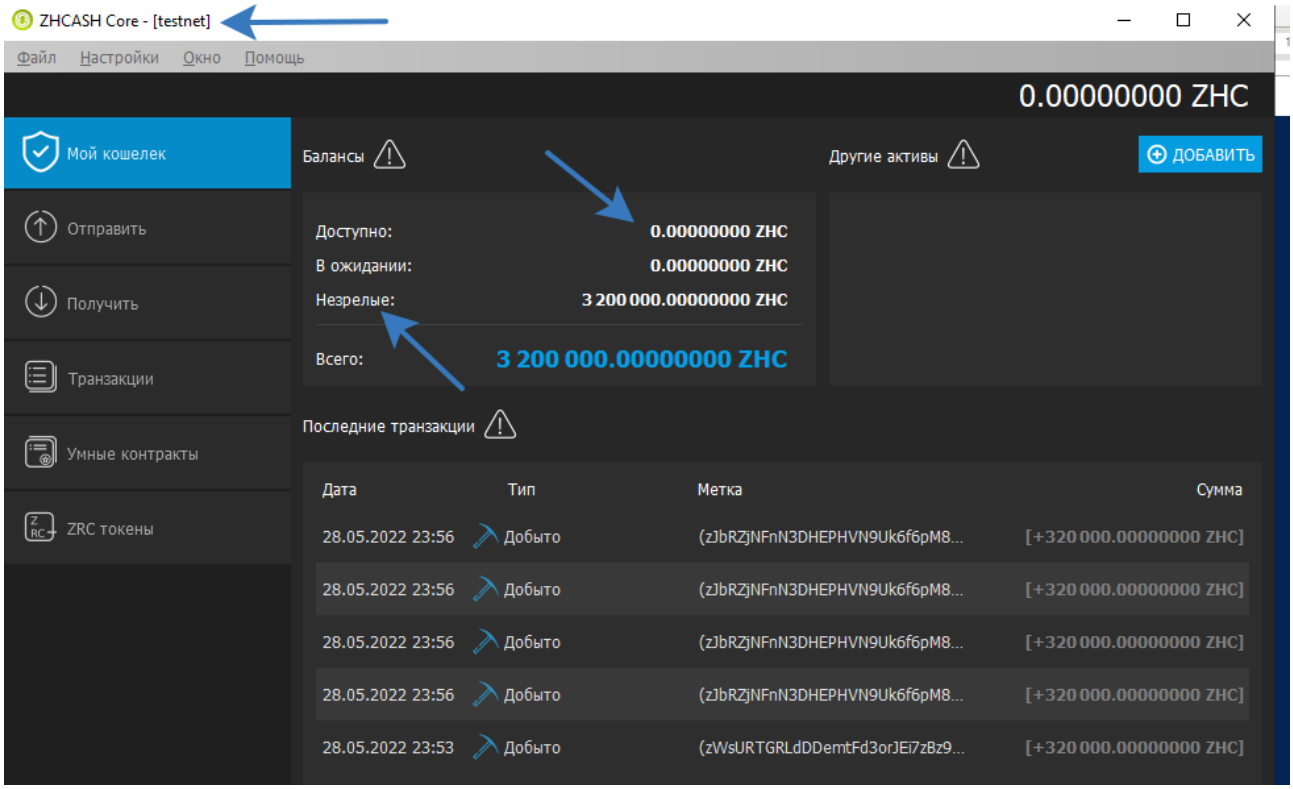
Тестирование смарта

Для тестирования смарта можно использовать три подхода.

1) Запустить свой кошелек zh в режиме тестнет (с ключом -testnet через командную строку).



В данном методе у вас организуется локальный блокчейн с нулевого блока, где вам ещё надо нагенерить 500 блоков для того, чтобы получить тестировочные шекели. Пока вы это не сделаете все шекели будут незрелыми и оплатить создание смарта не получится(

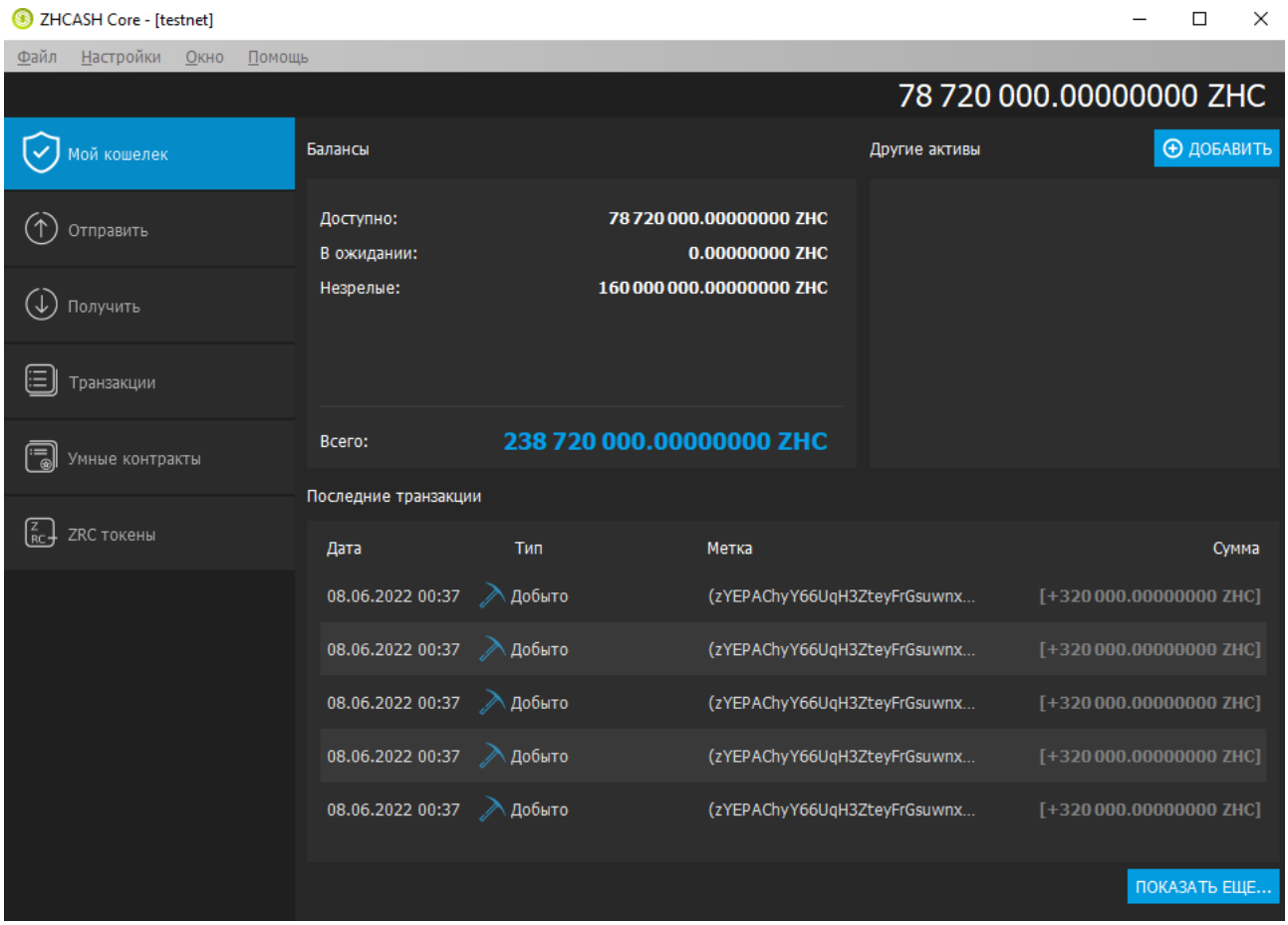


Заходим в консоль, пишем generate 10 и получаем ошибку. Для того, чтобы нагенерить блоки надо запускать с ключами -testnet -deprecatedrpc=generate. Повторяем ещё раз.

```
PS C:\Users\Yoga\Desktop> .\ZeroHour-Qt.exe -testnet
PS C:\Users\Yoga\Desktop> .\ZeroHour-Qt.exe -testnet -deprecatedrpc=generate
PS C:\Users\Yoga\Desktop>
```

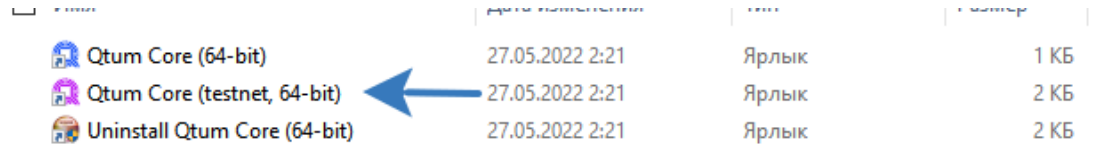
Тыкаем в консоле эту команду много-много раз, пока что-то не произойдет с балансом в разделе «Доступно». От незрелых шекелей у блокчейна несварение.

После многократного изнасилования клавиши вверх (для повторения прошлой командны) и enter блокчейн раздупляется и готов к работе



Можно грузить смарты, все проверять, потом генерить блок и все тестить.

2) Использования тестовой сети QTUM. У них есть кран для тестовой сети, где можно указать свой адрес и вам начилят 50 ± 20 тестировочных монет. При скачивании кошелька сразу доступен отдельный кошелек testnet.



На синхронизацию с тестировочным блокчейном уходит порядка 2 часа. Гайд по тестнету qtum https://docs.qtum.site/en/Testnet-User-Guide.html

3) Создание десятков тестовых смартов в основной сети (как сделал по началу автор. Поэтому он решил написать гайд), но это осуждается.

Автором был написан смарт для токена LIFT
https://github.com/dimaystinov/Token-LIFT-ZHCASH

Автор выражает благодарность инициаторам создания токена LIFT
https://t.me/lift_club с адресом `f180d0a911d09853685764a9ad6d366398c50656`
Николаю, Арджуну и Денису.

Главному инженеру блокчейна zh Роману, программисту Alex за ответы на тупые вопросы, которые легли в основу данного гайда.

@QtumLeandro (Из чата https://t.me/qtumofficial) за ответ, что отправлять данные в смарт надо всё-таки командой sendtocontract.