Министерство науки и высшего образования
Российской Федерации
Федеральное государственное бюджетное образовательное
учреждение высшего образования
«Воронежский государственный технический университет»

Кафедра инноватики и строительной физики
им. профессора И.С. Суровцева

ТЕХНОЛОГИЯ БЛОКЧЕЙН

МЕТОДИЧЕСКИЕ УКАЗАНИЯ

к выполнению практических и самостоятельных работ
по дисциплине «Blockchain - технологии»
для студентов направления 27.03.05 «Инноватика»
(профиль «Инновационные технологии») всех форм обучения

Воронеж 2021

УДК 336.74:004
ББК 65.050.253

Составители:
канд. техн. наук Д.В. Сысоев,
докт. физ.-мат. наук П.А. Головинский

Технология блокчейн: методические указания к выполнению практических и самостоятельных работ по дисциплине «Blockchain - технологии» для студентов направления 27.03.05 «Инноватика» (профиль «Инновационные технологии») всех форм обучения / ФГБОУ ВО «Воронежский государственный технический университет»; сост.: Д.В. Сысоев, П.А. Головинский. Воронеж: Изд-во ВГТУ, 2021. 25 с.

Предназначены для проведения практических и самостоятельных работ по дисциплине «Blockchain - технологии» для студентов направления 27.03.05.

Методические указания подготовлены в электронном виде и содержатся в файле PrZad-BchT.pdf.

Ил. 40. Библиогр.: 5 назв.

УДК 336.74:004
ББК 65.050.253

Рецензент – Н.В. Акамсина, канд. техн. наук, доцент
кафедры систем управления и информационных
технологий в строительстве

Издается по решению редакционно-издательского совета Воронежского государственного технического университета

Практическая работа №1
Установка Ethereum Wallet

Цель работы: Получить представления и начальные навыки работы в сети Ethereum.

Результат: наличие установленного кошелька Ethereum Mist Wallet.

Теоретическая справка:

Индустрия криптовалют развивается стремительно быстро: каждый месяц появляются новые варианты кошельков, торговых площадок, обменных сервисов, да и ассортимент самих цифровых активов постоянно расширяется. Это новое веяние в финансовой сфере, позволяющее умелым инвесторам неплохо заработать на волатильности рынка. А чтобы проводить с криптовалютой различные спекулятивные операции и получать свою прибыль, ее нужно где-то хранить. Вариантов огромное количество и сегодня мы рассмотрим один из них: многофункциональный десктопный кошелек Ethereum Wallet.

Рис. 1.

Особенности хранилища

Прежде всего, Ethereum Wallet - это официальный кошелек для хранения Эфира и токенов, созданных на его блокчейне. Это продукт поддерживается разработчиками криптовалюты, а значит, обеспечивает максимальную надежность и защищенность ваших активов.

Бумажник имеет еще одно имя - Ethereum Mist Wallet - он является функцией браузера Mist, который в данный момент еще дорабатывается и в скором времени станет связующим приложением платформы для взаимодействия с сетью.

Ethereum Wallet поддерживает работу на следующих операционных системах:

- Windows;
- Mac OSX;
- Linux.

Само приложение полностью бесплатно, все комиссионные сборы, которые вы будете оплачивать за переводы, переходят майнерам сети Ethereum, поддерживающих ее работоспособность.

Ethereum Wallet - один из самых безопасных вариантов хранения, так как все приватные ключи и данные о ваших сбережениях хранятся локально на вашем устройстве и не попадают в сеть. Эфириум-кошелек является шлюзом для децентрализованных приложений на blockchain Ethereum.

Обратите внимание на то, что приложение кошелька является довольно тяжелым, так как будет хранить полноценный блокчейн на вашем ПК, поэтому

позаботьтесь о том, чтобы было достаточно свободного места (более 200 Гб).

Устанавливая Ethereum Wallet, вы имеете полную анонимность: предоставлять какие-либо личные данные или проходить процедуру верификации не придется.

Как установить кошелек

Так как кошелек имеет официальный статус, работу начинаем непосредственно с сайта разработчика - https://ethereum.org

Рис. 2.

Все установочные файлы хранятся на Github. Выбираем свою версию и скачиваем архив на компьютер.

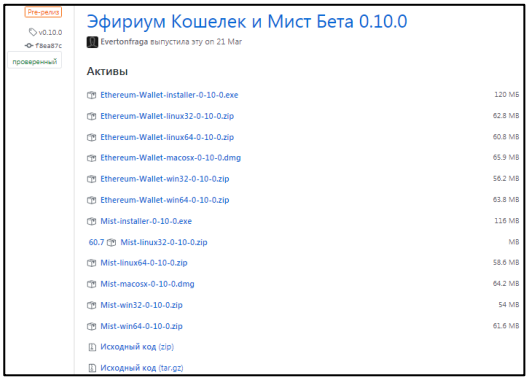

Рис. 3.

После этого папку нужно разархивировать и запустить установочный файл, как показано на рисунке 4.

Сама программа-клиент занимает не более 150 Мб, но синхронизация блокчейна потребует около 200 Гб. После того, как установка программы пройдет успешно, перед вами откроется окно (рис.5).

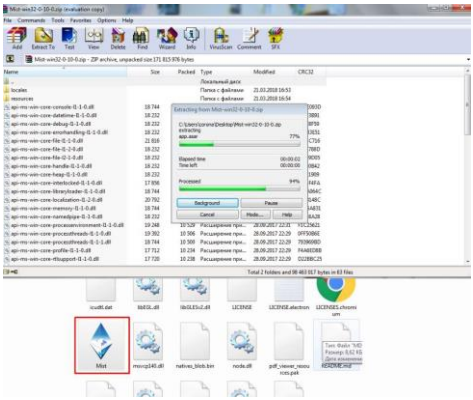

Рис. 4.

Рис. 5.

Вы должны выбрать сеть, в которой будет работать ваш кошелек:

• тестовую (для знакомства с кошельком она вполне подойдет);

• или основную (в этой версии блокчейн будет полноценно загружен).

На следующем этапе можно вводить бэкап хранилища, но так как мы его не имеем, пропускаем этот шаг:

Рис. 6.

Затем открывается окно, в котором вам нужно будет создать пароль. Имейте в виду, что изменить пароль не получится, поэтому сразу отнеситесь очень серьезно к созданию сложной комбинации символов.

Используйте не только буквы и цифры, но и разные регистры. Не забывайте, вы создаете защиту своим денежным активам.

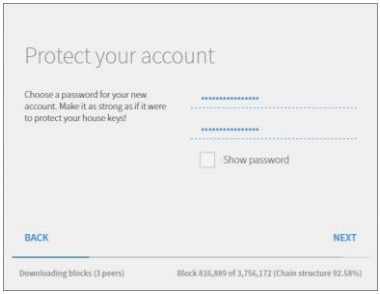

Рис. 7.

Следующее окошко напоминает нам, что использовать свой кошелек вы сможете, только имея на счету не менее 0,25 ETH. Перевести их можно прямо сейчас, используя встроенный инструмент ShapeShift. Если у вас пока нет нужных монет, вы сможете перевести их позже.

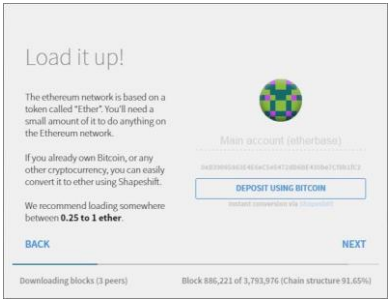

Рис.8.

Следующее действие - синхронизация хранилища с сетью Эфириум. Этот процесс может занять несколько суток, после чего Ethereum Wallet будет полностью установлен и готов к работе.

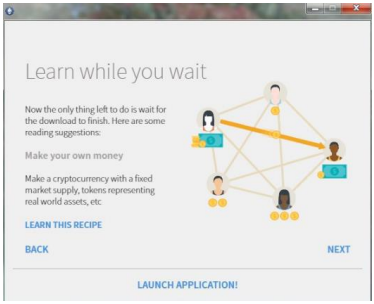

Рис. 9.

Что внутри?

Процесс использования кошелька Ethereum Wallet будет легким не только для опытных пользователей, но и для новичков. Его интерфейс вполне понятный и достаточно информативен.

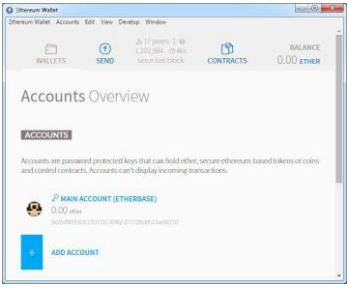

Рис. 10.

Перед тем, как пополнять баланс хранилища цифровой наличностью, лучше создать резервную копию. Для этого выбираем закладку «Аккаунты» и пункт «Резервное копирование». Сформированный файл сохраните в очень надежном месте, доступ к которому у вас будет всегда. Целесообразно иметь несколько копий.

Etereum Wallet имеет разделение на 2 основных раздела:

1. Accounts (в этом разделе вы сможете видеть баланс кошелька и суммы поступлений).

2. Contract Based Wallets (здесь вы сможете видеть полную информацию обо всех входящих и исходящих транзакциях и использовать DApps).

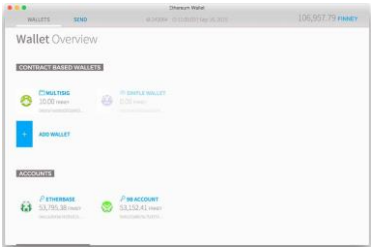

Рис.11.

Учетный раздел (Accounts) имеет закрытый ключ с паролем и адресом, контрактный раздел (Contract Based Wallets) не имеет закрытого ключа, но имеет свой адрес, код и хранилище, с помощью которых можно не только создавать кошельки, но и всевозможные интересные децентрализованные приложения (DApps).

Можно сказать, что учетная часть кошелька отвечает за простое хранение Эфира, а контрактная часть обеспечивает управление криптовалютой.

Как пользоваться кошельком

Первая задача - это пополнить баланс кошелька Эфиром. На главной

странице хранилища в разделе Main account вы можете видеть адрес хранилища, на который вам нужно перечислить как минимум 0,25 ETH.

Ethereum-адреса представлены в шестнадцатеричном формате: они состоят из 40 символов. Этот адрес вы будете давать всем своим знакомым, которые захотят перевести вам свои токены.

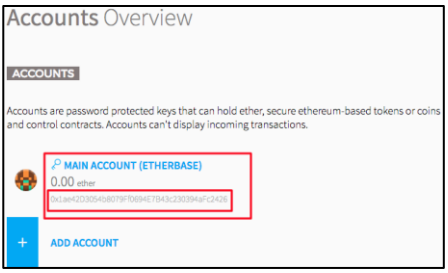

Рис.12.

Чтобы создать исходящую транзакцию и отправить эфир из Mist Wallet, выполните следующие действия:

1. Перейдите к разделу SEND.
2. Укажите адрес принимающей стороны.
3. Выберите количество эфира, который вы хотите отправить.

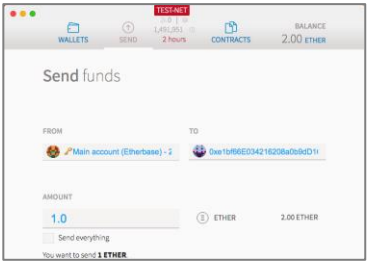

Рис.13.

Вы можете оставить комиссию такой, какую предложила программа по умолчанию, а можете изменять ее размер, перемещая бегунок:

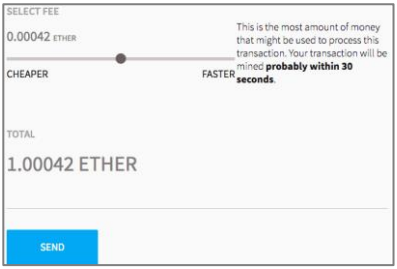

Рис.14.

Останется только ввести правильно свой пароль:

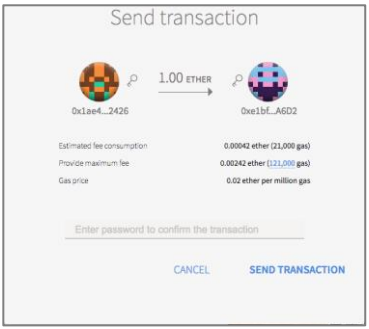

Рис.15.

Чтобы посмотреть информацию о какой-то конкретной транзакции, перейдите в раздел «Обзор кошелька» → «Последние транзакции» и кликните на нужную операцию в списке:

Рис.16.

Заключение

Ethereum Mist Wallet предназначен для надежного и безопасного хранения всех типов монет и токенов Эфириума, а также для смарт-контрактного взаимодействия пользователей через децентрализованные приложения браузера.

Положительные стороны Mist-кошелька - это его многофункциональность, понятный интерфейс и высокие показатели безопасности.

Из минусов можно отметить только очень долговременную синхронизацию: процесс занимает около 2 суток. Но, это плата за надежную сохранность ваших цифровых активов.

Задание

1. Установить кошелек Ethereum Mist Wallet. Подключаемся только к тестовой сети.
2. Ознакомиться с функционалом кошелька.

Контрольные вопросы

1. Почему при установке кошелька Ethereum Mist Wallet в рабочем режиме требуется большой объем свободного пространства на винчестере?
2. Какие операции доступны в кошельке Ethereum Mist Wallet?
3. Укажите недостатки кошелька Ethereum Mist Wallet?

Лабораторная работа №2
Знакомство с инструментами и средой разработки смарт- контрактов

Цель работы: изучить и закрепить на практике возможности основных инструментов разработчика смарт-контрактов.

Результат: наличие установленного кошелька MetaMask, настроенного на работу с тестовой сетью Rinkeby.

Теоретическая справка:

Смарт-контракты: что это?

Смарт-контракты, или "умные контракты", позволяют передавать некоторые ценности, например, собственность или акции, прозрачным и одновременно безопасным способом, что делает весь процесс сверхэффективным, одновременно устраняя промежуточные звенья, зачастую долгие и дорогие. Рассмотрим пример, который позволит понять, как блокчейн работает со смарт-контрактами.

Давайте представим, что есть два заинтересованных лица в сделке с недвижимостью. Один (продавец) желает продать жилье, а другой (покупатель) хочет купить это жилье. Сделка по продаже может быть реализована посредством блокчейна, и покупатель готов платить, например, биткоинами. Как только покупатель заплатит, то сразу получит подтверждение о транзакции, которое будет исполнено в виде виртуального смарт-контракта. Продавец, в свою очередь, передает покупателю цифровой ключ от входной двери, который будет доставлен в день, о котором заинтересованные стороны договорились. Если продавец вдруг передумает продавать дом, покупатель не получит ключ, блокчейн в этом случае автоматически вернет покупателю деньги в тот день, когда должен был быть получен ключ. А если покупатель получит ключ заранее, то блокчейн его удержит до дня, в который была договоренность осуществления передачи. Поэтому каждая из сторон получит то, что хочет, в оговоренный в контракте день: продавец - деньги, а покупатель - ключ. А поскольку блокчейн - это технология, основанная на пиринговой сети, договор по этой сделке будет храниться на множестве узлов, что обеспечит выполнение взятых по контракту обязательств, и ни одна из сторон не сможет изменить условия контракта после его заключения. Ну а если кто-то из сторон наберется смелости сделать это, все узлы в сети тут же об этом узнают, и проблема будет мгновенно решена.

Мы рассмотрели пример с куплей-продажей недвижимости. Но такие же соглашения могут заключаться при передаче акций, в страховании автомобилей или другого имущества и во многих других случаях.

Позвольте привести несколько ключевых преимуществ смарт-контрактов.

Первое качество, за которое смарт-контракты так ценятся, это автономность. Смарт-контракты не могут быть изменены третьими лицами, так как только их стороны заключают соглашение. Нет необходимости обращаться к услугам юристов при заключении соглашений.

Второе преимущество, за которое люди любят - или еще полюбят - смарт-контракты, это доверие к ним.

Смарт-контракт невозможно потерять. Они все зашифрованы и хранятся в общественном хранилище. Поэтому потеря любого из них исключена. Это подводит к следующему плюсу - резервированию. Можно положиться на надежность смарт-контрактов, потому что они все зарезервированы. Аннулирование договора по причине потери его копии просто невозможно.

Следующим в списке идет безопасность, которая опять же связана с предыдущими двумя. Ваши смарт-контракты будут защищены современными методами шифрования данных. Это отсылает нас к вопросу доверия - вы можете полностью доверять безопасности методов шифрования. Смарт-контракт практически невозможно взломать.

Пятая причина превосходства смарт-контрактов над обычными - это скорость их передачи. На заключение традиционных договоров уходит уйма времени, поскольку в их эту работу вовлечено множество третьих лиц. Если речь идет о распространении кода, смарт-контракты на высоте, поскольку позволяют решать задачи в разы быстрее.

Шестая причина - это экономия денег на заключении договоров. Нет необходимости прибегать к услугам адвокатов. Можно просто использовать технологию смарт-контрактов.

И, наконец, огромным преимуществом является точность. Если все подробности контракта указаны точно, то он будет выполнен значительно точнее, чем любой другой контракт.

Инструментарий и приложения экосистемы эфириума. Прежде чем погрузиться в написание кода, стоит изучить экосистему Ethereum. Давайте разберемся, какие инструменты и подходы существуют, как они называются и взаимодействуют.

В экосистеме Ethereum широко используются такие инструменты, как Geth, Parity, Solidity, Remix, Truffle, Webpack, Angular и так далее. Каждый из них используется для решения конкретных задач.

Узлы сети блокчейна: Go-Ethereum, Parity, CPP-Ethereum. Примерами узлов блокчейна выступают такие программы, как Geth, Parity или CPP-Ethereum.

Все они работают на клиентской стороне, то есть их можно загрузить и запустить на вашем компьютере, как и для всех других пользователей сети Ethereum. Они все выполняют одну и ту же задачу: реализуют протокол Ethereum. Несмотря на то, что разные инструменты выполняют одну и ту же роль, они написаны на разных языках программирования. Развитием инструментов занимаются различные команды, которые обязательно следят за тем, чтобы даже на разных языках программирования протокол Ethereum был реализован корректно. Если проводить аналогию, то эта схема похожа на использование среды MySQL в режиме "мульти-мастер", когда все узлы выполняют одну и ту же задачу по репликации базы данных. Это отлично описывает то, что делают все узлы в сети блокчейна - они копируют все блоки на своих компьютерах. Поэтому при загрузке Geth, Parity, или CPP-Ethereum и запуске клиента после установки подключения к другим узлам будет загружено все содержимое блокчейна. Исключение составляет

только режим "легкого клиента", когда загружаются только заголовки блоков.

Взаимодействие веб-сайтов и блокчейна. Рассмотрим популярные браузеры MetaMask и Mist. Оба они представляют собой связующее звено между обычным браузером для просмотра интернет-страниц и блокчейном С помощью корректно настроенного веб-сайта можно выполнять программы и отправлять команды в блокчейн. *Пользователь* сможет запустить любой *браузер*: например, *Chrome*, Firefox, *Internet* Explorer или другой *браузер*, зайти на такой *веб-сайт* и взаимодействовать с блокчейном. Для этого к блокчейну необходимо подключиться. MetaMask представляет собой надстройку для *Chrome* и Firefox, облегчающую подключение к блокчейну.

Рис. 17.

Mist, в свою *очередь* - это полновесный *браузер*, оснащенный собственным узлом сети блокчейна.

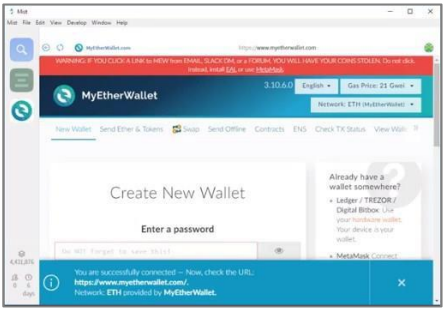

Рис. 18.

В случае с Mist, узлом блокчейна является Geth, или Go-Ethereum, непосредственно встроенный в *браузер*. MetaMask для работы использует сервис под названием Infura. В среде Infura используются узлы Geth и *Parity*, которые запущены на стороне сервера, а не на компьютере клиента, а Infura реализует подключение к ним. Чтобы познакомиться с надстройкой MetaMask, можно открыть *браузер*, например, *Chrome*, найти раздел с иконками надстроек, далее открыть MetaMask и можно начинать работать с блокчейном. Кошелек

Mist выглядит так: слева *доступ* к различным разделам, есть *отображение* статуса подключения и синхронизации данных, в центре располагается собственно *браузер* Это позволяет работать с блокчейном и просматривать веб-страницы можно одновременно.

Что такое Solidity. Solidity представляет собой *язык программирования* высокого уровня. Для ее работы требуется *компилятор* solc, который формирует байткод для виртуальных машин Ethereum.Встречаются мнения, что Solidity похож на JavaScript. В первых версиях так и было, однако сейчас эти два языка значительно расходятся. Тем не менее, Solidity похож на JavaScript больше, чем любой другой *язык программирования*.

Remix, веб-среда разработки для Solidity. Remix - это облачная *среда разработки*, поддерживающая много полезных функций. *Доступ* к Remix можно получить по адресу http://remix.ethereum.org. Среда Remix позволяет создавать и запускать код на языке Solidity прямо в окне браузера. Remix оснащена встроенным отладчиком и статическим анализатором кода, а также многими другими инструментами.

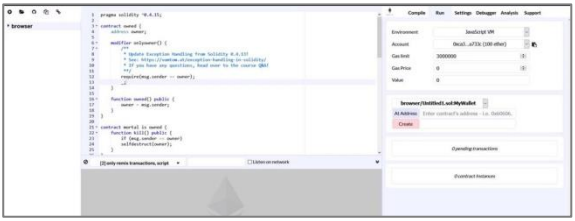

Рис.19.

На текущий момент Remix выглядит так. Слева расположен *браузер*, с помощью которого можно управлять файлами. В центре располагается окно для создания кода, а справа - *управляющие* элементы - вкладки для компиляции (*Compile*), запуска (Run), изменения настроек (Settings), отладки (*Debugger*), анализа (*Analysis*) и получения поддержки (*Support*).На вкладке Run можно выбрать среду запуска кода, например, виртуальную машину *Java*. Remix предоставляет *доступ* к нескольким счетам в эмулированной среде Ethereum для апробирования создаваемого кода. С их помощью можно размещать и обсчитывать контракты, а потом анализировать результаты благодаря наличию журнала исполнения кода.

Использование библиотек Web3.js и Eth.js. Библиотеки Web3.js и Eth.js облегчают взаимодействие между браузером и блокчейном и позволяют работать узлами сети Ethereum по протоколу *RPC* посредством *HTTP* и кода JavaScript. Если запустить локальный узел блокчейна, он откроет *интерфейс HTTP-RPC*, что позволит браузеру отправлять узлу команды, чтобы узел, в свою *очередь*, переправлял данные в блокчейн.

Библиотека Truffle и ее отличие от Web3.js. Truffle и Embark являются инструментариями для среды Solidity и разработки распределенных приложений для работы с блокчейном. Оба они поддерживают управление контрактами, их *размещение* в блокчейне, или миграцию, оснащены встроенной

системой тестирования приложений, а Truffle еще и предлагает решение Truffle Boxes - предварительно настроенные среды разработки распределенных приложений, значительно облегчающие работу, такие как Truffle-React, Truffle-Webpack и так далее. При серьезном подходе к разработке приложений для блокчейна стоит уделить внимание Truffle и Embark и постепенно отходить от использования только библиотеки Web3.js.

Использование Angular, Vue.js, React и Redux в разработке приложений для блокчейна. Такие наборы инструментов, как Angular, Vue.js, React, Redux предназначены для разработки веб-страниц и непосредственно не работают с блокчейном, Truffle, Solidity и другими подобными средами. Для работы с Angular, Vue.js, React, Redux или другими инструментариями для создания веб-страниц обычно достаточно загрузить библиотеку Web3, подключиться к узлу блокчейна и настроить взаимодействие с блокчейном с помощью Web3.

Применение инструментов Browserify и Webpack. Webpack - это упаковщик файлов JavaScript, необходимо использовать тогда, когда *программа* использует большое количество файлов: Webpack собирает их воедино, разрешает все взаимозависимости между файлами, позволяя коду обращаться только к паре мастер-файлов, что значительно ускоряет загрузку веб-приложения, тк веб-серверу больше не приходится отправлять несколько сотен файлов.

Browserify делает примерно то же самое, но на базовом уровне - ведь Webpack сразу решает спектр задач по упаковке файлов для веб-разработки. Browserify представляет собой только упаковщик, разрешающий файловые взаимозависимости и объединяющий много файлов в один. *Node Package Manager*, или NPM, загружает и управляет пакетами для узлов сети Ethereum, что облегчает разработку веб-проектов.

Обзор и возможности MetaMask. Поговорим о надстройке для браузера *Chrome* - MetaMask. В этом разделе установим ее, разберем функционал, а также узнаем, как получить немного эфира для тестирования MetaMask и размещения контрактор в тестовой сети Rinkeby.

Установка MetaMask.

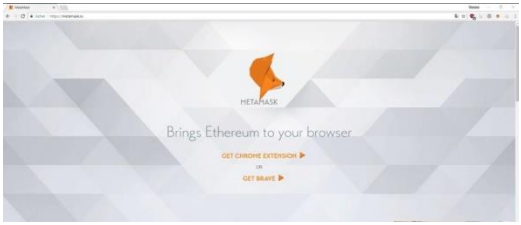

Рис. 20.

Откроем веб-*сайт* MetaMask, маскотом которой является лисичка.

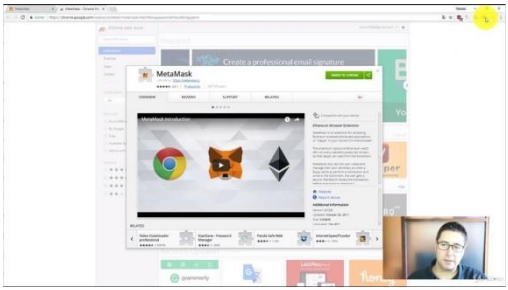

Рис. 21.

При щелчке на *Get the Chrome Extension* (*браузер Chrome* существует для всех платформ) вы перейдете на страницу установки надстройки MetaMask и увидите кнопку *Install*, если MetaMask еще не установлена.

Рис.22.

После установки MetaMask можно будет запустить с помощью ее иконки в верхнем правом углу браузера *Chrome*, а несколько простых шагов установки сопровождаются подробными инструкциями. Потребуется задать *пароль* для защиты ваших счетов, после этого можно работать с MetaMask.

Элементы MetaMask. Продолжим разговор о надстройке MetaMask. Для начала выбраем *сеть*, с которой вы будете взаимодействовать - ее можно выбрать в левом верхнем углу окошка MetaMask.

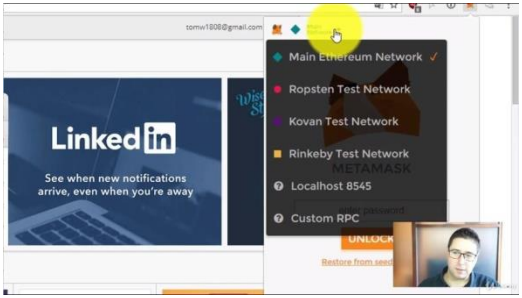

Рис.23.

После этого необходимо опубликовать свое *приложение* в главной сети Ethereum, которое будем использовать для взаимодействия с другими смарт-контрактами, опубликованными там.

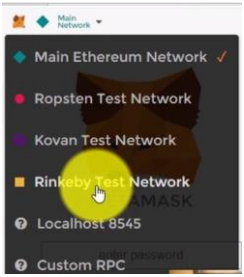

Рис. 24.

В рамках данного курса будет достаточно тестовой сети Rinkeby. Мы узнаем, как получить немного эфира для использования в этой тестовой сети, разобраться с тем, как *сеть* реагирует на запросы, посмотрим на майнинг, задержки и проблемы одновременных вычислений, характерные для блокчейна.

Теперь авторизуемся в MetaMask. Используем *пароль* и после входа в систему будет создан счет.

Рис.25.

Доступен обзор имеющихся счетов, просмотр совершенных транзакций, а также жетонов среды Ethereum.

Рис. 26.

Если на счету есть эфир, его можно отправить на другой *адрес* и добавить к транзакции данные. Об этом мы поговорим позднее. На данном этапе

ограничимся просто пересылкой эфира.

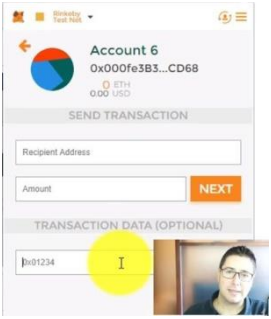

Рис. 27.

В *меню* обзора счета можно скопировать *адрес* в *буфер* обмена и экспортировать частный *ключ* вашего счета для использования его в другом приложении.

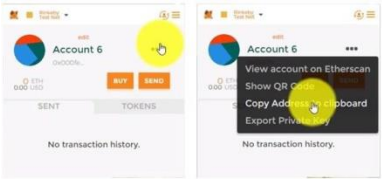

Рис. 28.

Обратите внимание, что *доступ* к счетам обеспечивается с помощью вашего частного ключа.

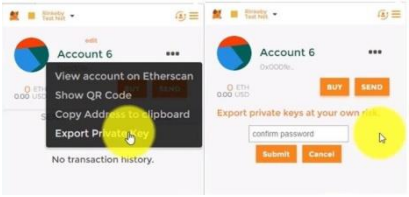

Рис.29.

Для переключения между счетами можно использовать опцию в верхнем правом углу.

Рис. 30.

Можно создавать новые счета с помощью кнопки *Create Account* или импортировать счета с помощью *Import Account*.

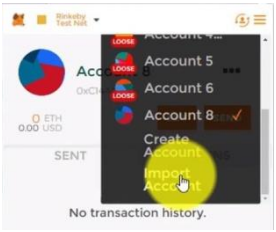

Рис. 31.

Для этого потребуется частный *ключ* к счетам, предварительно созданным в Geth или другой системе, создающей ключи *JSON*.

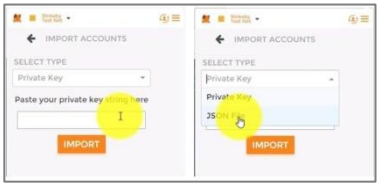

Рис.32.

Импорт осуществляется посредством этих файлов-ключей *JSON*.

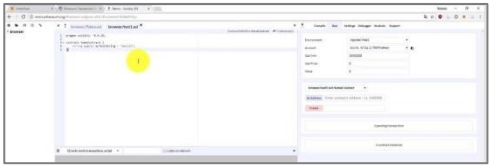

Рис. 33.

Надстройка MetaMask подключается непосредственно к окну браузера и таким образом обеспечивает *связь* с блокчейном. В раскрывающемся списке выбрана Injected Web3, а в окне разработки, доступном для любой веб-страницы, видно, что с помощью объекта *web3.currentProvider* можно работать с надстройкой MetaMask посредством обычного кода JavaScript.

Рис. 34.

Выберем Injected Web3, затем счет - тот же самый, который открыт в надстройке MetaMask. Теперь можно перейти на вкладку Run и создать кон-

тракт.

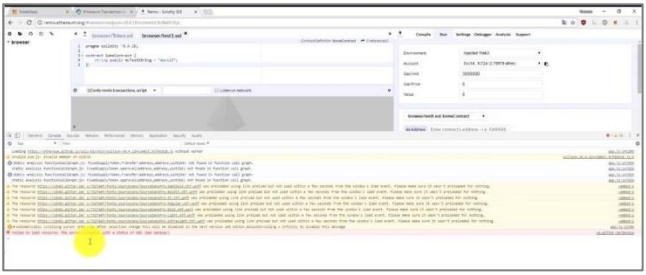

Рис. 35

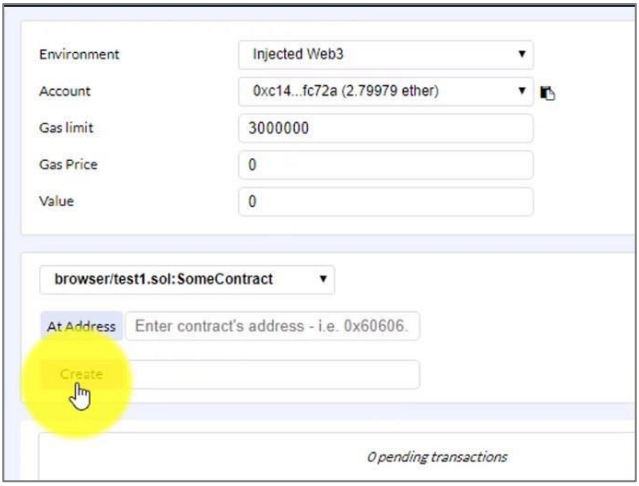

Рис. 36.

Отследим всю цепочку: контракт, написанный в среде Remix на языке JavaScript, исполняется, надстройка MetaMask отслеживает эту транзакцию и открывает всплывающее окно для подтверждения создания контракта.

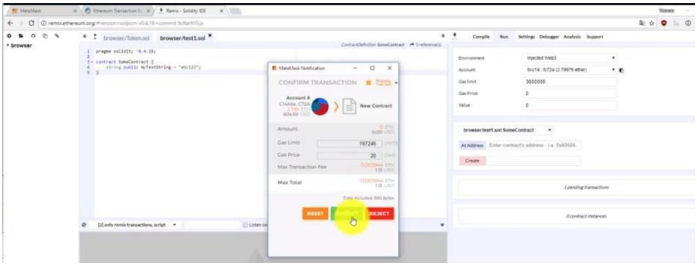

Рис. 37.

Для этой процедуры потребуется израсходовать немного газа. После подтверждения MetaMask отправляет транзакцию в блокчейн.

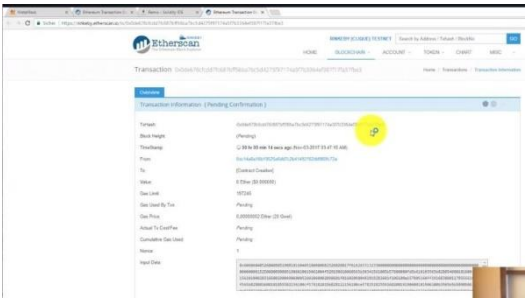

Рис.38.

MetaMask располагает некоторым числом узлов сети, расположенных на серверах разработчиков, которые играют роль посредника между вашим браузером и блокчейном. Если вы хотите создать и использовать свой собственный блокчейн, потребуется один из клиентов: Geth, *Parity*, Mist. Mist обеспечивает интегрированный в *браузер достуn* к блокчейну, а Geth - это работающий на вашем компьютере клиент, загружающий все блоки и предоставляющий к ним непосредственный *достуn*. Успешность завершения транзакции подтверждается обсчитанными блоками.

Основные понятия среды Ethereum. Блокчейн в среде Ethereum очень напоминает блокчейн для Bitcoin. Есть транзакции, есть эфир, подобный биткоину, есть майнеры и так далее. Однако есть и различия. Во-первых, майнеры все еще работают по механизму Proof-of-Work, то есть решают математические задачи по шифрованию данных. Это требует большого количества энергии и вычислительной мощности, а после успешного завершения к блокчейну добавляется новый блок. В скором будущем будет внедрен механизм Proof-of-Stake, который потребует приобретения определенного количества эфира, который будет использоваться для обсчета новых блоков.

Самым большим отличием от блокчейна Bitcoin является возможность размещать приложения непосредственно в блокчейне. Именно этой теме будет посвящен данный раздел.

Приложения с максимальной доступностью. Технология блокчейн подразумевает, что все приложения обладают невероятной устойчивостью и надежностью, поскольку исполняются непосредственно в блокчейне - единый *сервер* отсутствует, код выполняется на всех узлах блокчейна одновременно. Это открывает большие возможности для решения сложных вычислительных задач, а также построения высокораспределенных сред.

Основы работы среды Solidity. Среда Solidity представляет из себя высокоуровневый *язык программирования*. Существуют и другие способы разработки приложений для блокчейна, но общим правилом является необходимость компиляции текста на языке программирования в байткод, который затем размещается в блокчейн посредством транзакции. Такие транзакции очень похожи на используемые в блокчейне Bitcoin, когда вы отправляете биткойны с одного адреса на другой; в среде Ethereum для тех же целей используются единицы эфира. Если есть потребность отправить в блокчейн

байткод, то он прикрепляется к транзакции как данные, а в самой транзакции при этом должно быть пустым *поле* получателя *To* - в этом случае блокчейн создаст новый *адрес* для размещения этого байткода.

Практический пример

Рис. 39.

Например, предположим, что у нас есть *функция ABC*, требующая *параметр* a. Если a меньше 50, *функция* возвращает 10, в противном случае a. *Компилятор* обрабатывает функцию, и в блокчейн отправляется новая транзация со следующими значениями полей: *from* содержит *адрес* отправителя - ваш *адрес*, *поле value* пусто, как и *поле to* (это самый важный момент), а *поле data* содержит байткод из функции, созданной в Solidity. В процессе обсчета транзакции этот код будет добавлен в очередной блок, получит собственный *адрес*, например, *0xabcdef001*, или какой-нибудь другой, и у каждого пользователя сети появится возможность взаимодействовать с этим кодом по заданному адресу.

Важность обсчета кода и учета валюты в одном блокчейне. Блокчейн представляет собой значительно распределенную базу данных. Это означает, что при сохранении в блокчейне какой-либо величины или участка кода их больше нельзя удалить, их доступность крайне высока. Ни какие-либо величины, ни участки кода не доступны из централизованного источника, поэтому никакому правительству не под силу ограничить *доступ* к ним или удалить данные из блокчейна, если только они не выключат все узлы сети по всему миру. Эта концепция напоминает сохраненные процедуры в MySQL, только применимо по отношению к коду. В частности, в среде MySQL вы можете запускать некоторые программы, изменяющие запросы *SELECT* или *RETURN* - аналогично в блокчейне, особенно в Ethereum, с помощью смарт-контрактов можно изменять значения переменных или данные, отправлять валюту и другим образом взаимодействовать с другими смарт-контрактами.

В среде Ethereum и обработка кода, и валютные *операции* проводятся в едином *поле*. Это открывает широкие возможности для различных приложений из областей условного депонирования, краудфандинга, страхового дела, операций с недвижимостью, сферы услуг, юриспруденции и прочих. В настоящее время наблюдается множество проектов, использующих возможности запуска кода в блокчейне и работы с криптовалютой с последующим проведением краудфандинговых кампаний.

Классические примеры распределенных приложений. Приведем несколько примеров распределенных приложений. Начнем с ДАО - демократических автономных организаций. Эта система представляет собой платформу для краудфандинга. Одно время широко обсуждалась в прессе, поскольку разработчики смогли привлечь шестьдесят миллионов долларов США в виде инвестиций. К большому сожалению, она впоследствии была взломана, но оставила значительный след в сознании людей, благодаря ясной логике и новому подходу к краудфандингу, при котором не представлялось возможным собрать средства и сбежать (в отличие, например, от Kickstarter, который тоже принимает средства для разработки новых продуктов, но нет гарантии, что он не обанкротится). В случае с ДАО отсутствует центральное передаточное звено, способное скрыться с деньгами, намеренно, поскольку все договоренности обеспечиваются смарт-контрактами.

Вторым распространенным типом приложений являются решения по обмену валюты. В настоящее время наблюдается всплеск числа ICO, - первичных размещений криптовалюты - реализуемых с помощью токенов стандарта *ERC,* которые можно напрямую обменивать в среде блокчейна, то есть менять токены на эфир или токены на токены, и вся эта логика хранится непосредственно в блокчейне.

Токены - это участки кода, связанные с пользовательскими учетными записями и базами данных и используемые как валюта, баллы в программах лояльности, индикаторы доли в компании, жетоны в играх виртуальной реальности и так далее. Все эти платежные средства учитываются и хранятся в среде блокчейна.

И, наконец, *базы данных*. Снабдив их некоторой логикой, можно вести в блокчейне учет владельцев земельных участков, дипломов об образовании (например, Массачусетский технологический институт выпускал свои сертификаты в блокчейне), даже законов. В случае использования блокчейна нет необходимости прописывать положения закона в каком- то документе, вместо этого можно задать условия работы смарт-контрактов, сразу разрешая или запрещая какие-то действия в блокчейне. В перспективе можно даже прийти к автоматическому списанию средств со счета, например, в качестве штрафа за неправильную парковку.

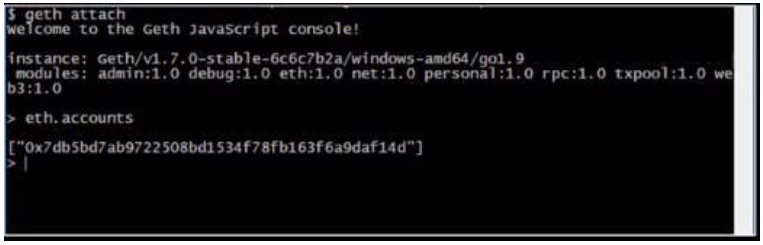

Рис. 40.

Как работает доступ к блокчейну. Для обеспечения доступа к блокчейну используются узлы сети Ethereum, взаимодействующие друг с другом

посредством протокола Ethereum. Каждый *узел сети* может обращаться к любому другому. Одним из узлов, доступных для свободной загрузки, является Go-Ethereum. Он, как и остальные реализации, подключается и взаимодействует с сетью посредством протокола Ethereum.

С другой стороны, для выполнения операций в блокчейне можно применять *удаленный вызов процедур* (*Remote Procedure Call*, *RPC*), запуская файлы *JSON*, созданные на JavaScript.

Удаленный вызов процедур можно реализовать через протокол *HTTP*, что позволяет взаимодействовать пользователям с узлами сети, а самим узлам - друг с другом посредством протокола Ethereum. Важно понять схему: *пользователь* работает с файлами *JSON* для удаленного вызова процедур, а узлы передают эту информацию между собой по протоколу Ethereum. Это напоминает работу в консоли MySQL, когда *пользователь* задает запросы MySQL (в случае с Ethereum отправляются команды для удаленного вызова процедур в формате *JSON*), а узлы сети MySQL обмениваются информацией по протоколу MySQL (в блокчейне узлы взаимодействуют по протоколу Ethereum).

Задание

1. Установить кошелек MetaMask. Подключаемся только к тестовой сети Rinkeby.
2. Откройте несколько счетов (минимум 3).
3. Получите эфир для дальнейшей работы.
4. Распределите полученный эфир между тремя счетами.
5. Ознакомьтесь с возможностями кошелька MetaMask.

Контрольные вопросы

1. Что такое Go-Ethereum?
2. Что такое Web3?
3. Что такое Remix?
4. Что такое Solidity?
5. Что такое MetaMask?
6. Как MetaMask взаимодействует с браузером?
7. Укажите несколько различий сети Ethereum и блокчейна Bitcoin?

Библиографический список

1. Могайар У. Блокчейн для бизнеса. – М: Издательство «Эксмо», 2018. – 224 с.
2. Фергюсон Н., Шнайер Б. Практическая криптография: Пер. с англ. – М.: Издательский дом “Вильямс”, 2004. — 432 с.
3. Головинский, П.А. Блокчейн и криптовалюты [Текст] : учебно-справочное пособие для преподавателей и студентов высших учебных заведений / ФГБОУ ВО "Воронеж. гос. техн. ун-т". - Воронеж : Цифровая полиграфия, 2019. - 88 с.
4. Свэн М. Блокчейн. Схема новой экономики; перевод, оформление, издание – М,: Издательство «Олимп – Бизнес», 2017. – 240 с.
5. Максуров, А.А. Блокчейн, криптовалюта, майнинг: понятие и правовое регулирование : монография / А. А. Максуров. - Блокчейн, криптовалюта, майнинг: понятие и правовое регулирование ; 2024-05-18. - Москва : Дашков и К, 2021. - 212 с.

ТЕХНОЛОГИЯ БЛОКЧЕЙН

МЕТОДИЧЕСКИЕ УКАЗАНИЯ

к выполнению практических и самостоятельных работ
по дисциплине «Blockchain - технологии»
для студентов направления 27.03.05 «Инноватика»
(профиль «Инновационные технологии») всех форм обучения

Составители:
канд. техн. наук Д.В. Сысоев,
докт. физ.-мат. наук П.А. Головинский

Подписано к изданию _____.
Уч.-изд. л. _____.