ДЗ. Инструментарий Ethereum

Форма: github репозиторий, транзакция в тестовой сети Ethereum
Область: Ethereum
Цель работы: получить начальные навыки работы в сети Ethereum
Результат: подготовленное окружение для работы с Ethereum

Описание

Перед тем, как начинать разрабатывать смарт-контракты, нужно познакомиться с основными инструментами экосистемы.

В основе экосистемы Ethereum лежат “узлы” блокчейна Ethereum, который реализует протокол Ethereum, принимает запросы от пользователей и участвует в синхронизации состояния блокчейна, действуя в унисон с другими узлами. Самые часто используемые реализации узлов Ethereum: Geth (https://geth.ethereum.org), Parity, CPP-Ethereum.

Для взаимодействия с сетью используется вид приложений, известные как кошельки. Самые часто используемые: Metamask (https://metamask.io), MyEtherWallet.

Для написания кода смарт-контрактов используется широкий набор инструментов, такие как: Remix IDE, инструменты разработчика Truffle (https://www.trufflesuite.com) и HardHat (Hardhat), библиотеки полезных смарт-контрактов OpenZeppelin (https://www.openzeppelin.com) и библиотеки для тестирования Waffle (Waffle).

Отдельно стоит упомянуть про специальные библиотеки под общим названием web3 (web3js, web3py, web3j, и так далее) которые позволяют создавать Децентрализованные Приложения, являясь связующим звеном между блокчейном и внешними информационными системами.

Для целей разработки и тестирования новых версий протокола и смарт-контрактов в экосистеме Ethereum работают несколько тестовых сетей, полный список найдете здесь: https://ethereum.org/ru/developers/docs/networks/.

Каждая сеть имеет свои особенности. Но все они (и основная сеть) требует оплаты выполнения операций с данными в блокчейне. Таким образом любую транзакцию в сети Ethereum следует оплачивать. Если в основной сети в качестве оплаты используется ether, которая имеет конкретную рыночную стоимость, то в тестовых сетях используется технические расчётные единицы, которые не имеют рыночной стоимости, но всё равно необходимы, для проведения операций в блокчейне Ethereum. Для получения технических расчётных единиц используют “краны” (faucet), которые начисляют тестовые р.е. на указанный адрес.

Отличительная особенность экосистемы Ethereum - разнообразие инструментов для разработчиков смарт-контрактов. Первым инструментом из которых является online IDE Remix (http://remix.ethereum.org/). Remix это браузерная IDE, которая позволяет разрабатывать, отлаживать, размещать и запускать смарт-контрактов в сетях Ethereum.

Состав Remix:

- редактор кода,
- статический анализатор кода,
- компилятор, средства настройки компиляции и размещения кода,
- адаптер подключения к блокчейн сети на основе Ethereum,
- разнообразные плагины для специфических случаев разработки и использования смарт-контрактов.

Например в настройках подключения можно выбрать провайдера среды исполнения (локальный узел, удалённый узел, временная среда). Remix - используется в случаях, когда нет возможности установить локальную IDE или требуется оперативно проверить исследовательскую гипотезу. При этом все доступные возможности этой IDE позволяют создавать смарт-контракты произвольной сложности.

Задание

Основное (7 баллов)

1. Установить https://metamask.io
2. Переключиться MetaMask на сеть Sepolia ~5 минут
3. Пополнить свой счёт в тестовой сети Sepolia.

4. Отправить SepoliaETH (в любом количестве) на адрес: 0xaB854be0A4d499B6FD8D0bB5F796Ab5b33cE825b
5. Создать github репозиторий
6. Подготовить структуру папок, в помощь https://hardhat.org.
7. Разместить подготовленную структуру папок в репозитории
8. Сообщить адрес репозитория в личку didexBot

Дополнительное (9 баллов)

1. Используя один из стандартов токенов (ERC-20, ERC-721, ERC-1155) создать простой смарт-контракт в подготовленной структуре папок.
2. Разместить смарт-контракт в сети Sepolia
    a. Скинуть адрес смарт-контракта в README.md
3. Добавить функцию чеканки новых токенов
4. Обновить репозиторий в github

Контрольные вопросы (10 баллов)

Ответы сформировать в виде текстового документа в репозитории README.md, указать автора и группу.

1. Почему для работы с приложением MetaMask'у нужен синхронизированный узел? Можно ли обойтись без этого и какие риски могут быть с этим связаны?
2. Как вы оцениваете производительность работы Ethereum, с чем это может быть связано?
3. Имеет ли зависимость адреса кошелька от сети с которой он работает?
4. Что такое Remix VM и чем она отличается от EVM?
5. Какие основные артефакты создаются после компиляции смарт-контрактов и после размещения в сети?

Ссылки

1. The Complete Hands-On Hardhat Tutorial - DEV Community
2. ERC20
3. Ethereum (ETH) Blockchain Explorer
4. json-rpc
5. Infura: Ethereum API | IPFS API & Gateway | ETH Nodes as a Service
6. List of Ethereum's Major Network and Chain IDs

7. Adding a Custom RPC to Metamask
8. Account Management — Ethereum Homestead 0.1 documentation