PRO Magento

PRO Magento
Meetup #6

# Применение блокчейн технологий в  eCommerce.

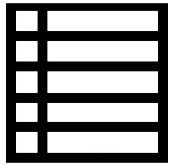
Алексеенко Игорь
Senior Backend Magento Developer
IT Delight

# О чем сегодня поговорим?

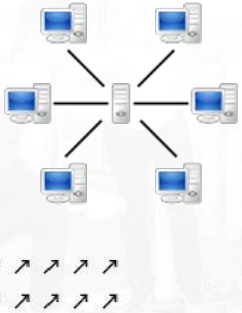
1. Что такое блокчейн и как он работает
2. Смарт-контракты
3. Токенизация
4. Возможности блокчейн технологий в eCommerce

# Blockchain.

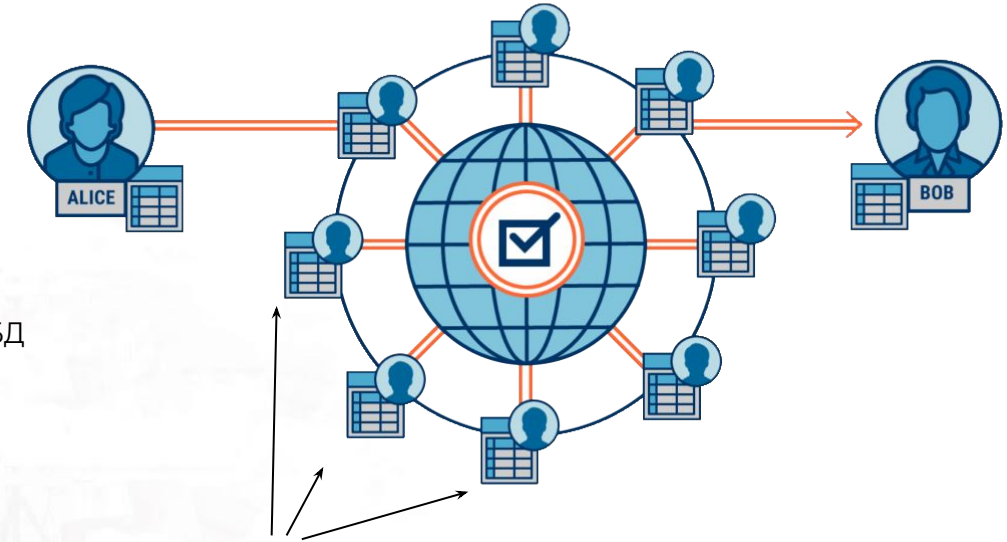
ledger (бух. книга)



централизованная БД



децентрализованная сеть/БД

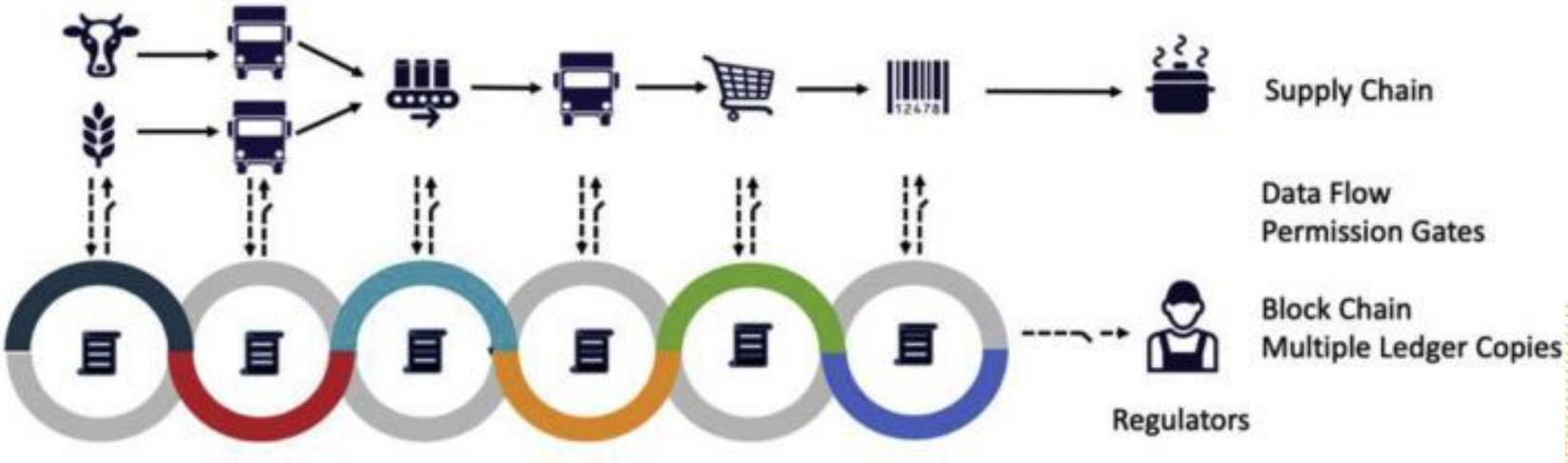


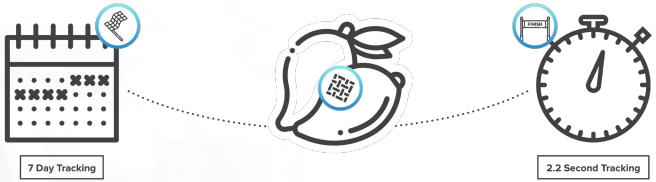

НОДЫ

**Принципы:**

- децентрализация
- распределенность
- прозрачность
- неизменность
- защищенность

# Hyperledger, Wallmart, IBM



# Blockchain. Транзакции

software wallet

инициация транзакции
и подпись

создание блока с
транзакцией

отправка блока во все
ноды

валидация блока
нодами

**вознаграждение нодам**

добавление блока к
существующим блокам

обновление сети

транзакция завершена

# Blockchain. Кошелек

software wallet

Blockchain Wallet

Email client

public key → Blockchain wallet ID

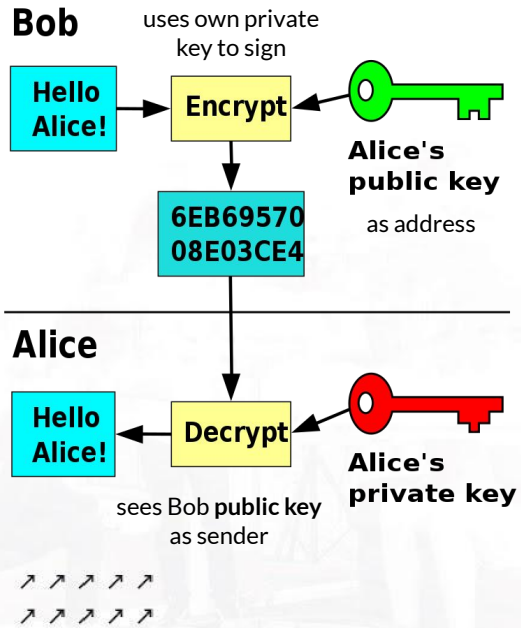
Email address

wallet access → Private Key

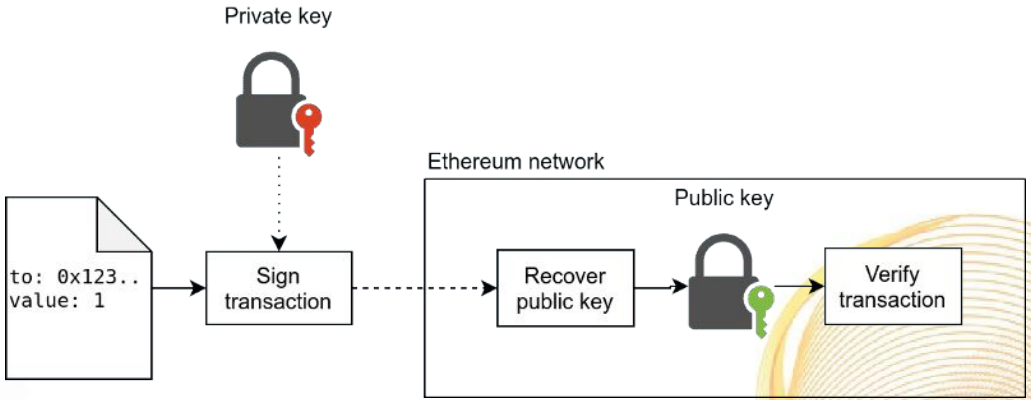
Email password

# Blockchain. Подпись транзакции

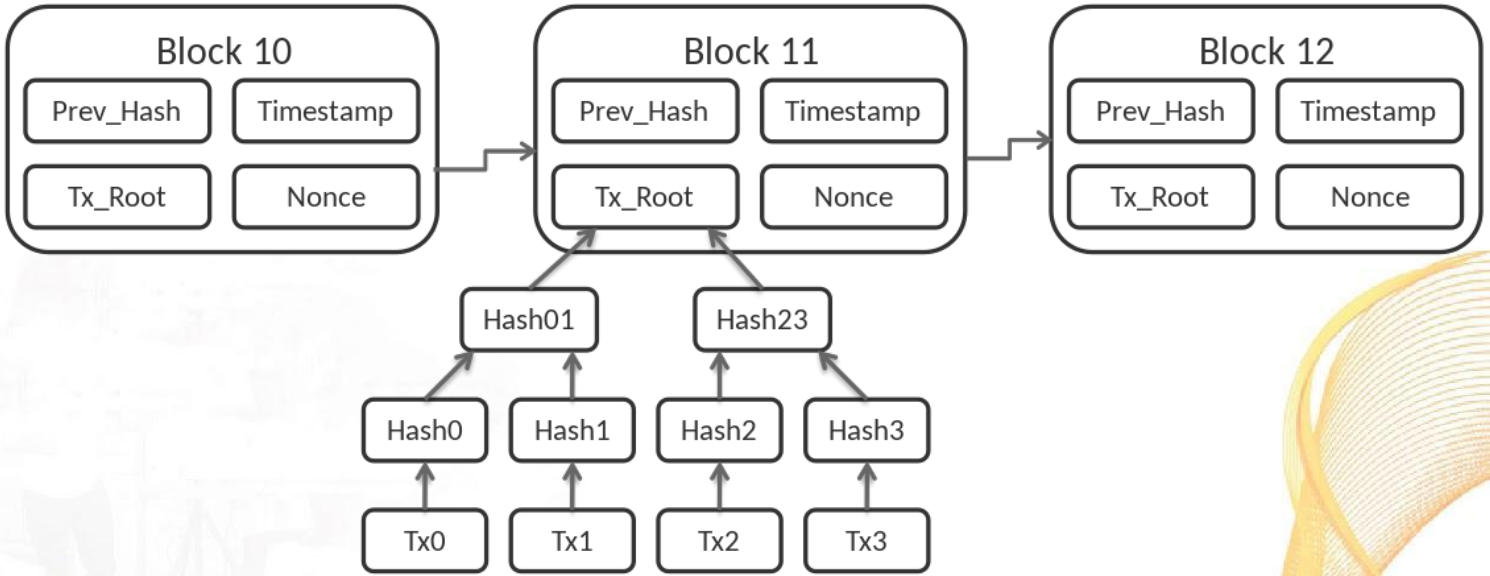
шифрование



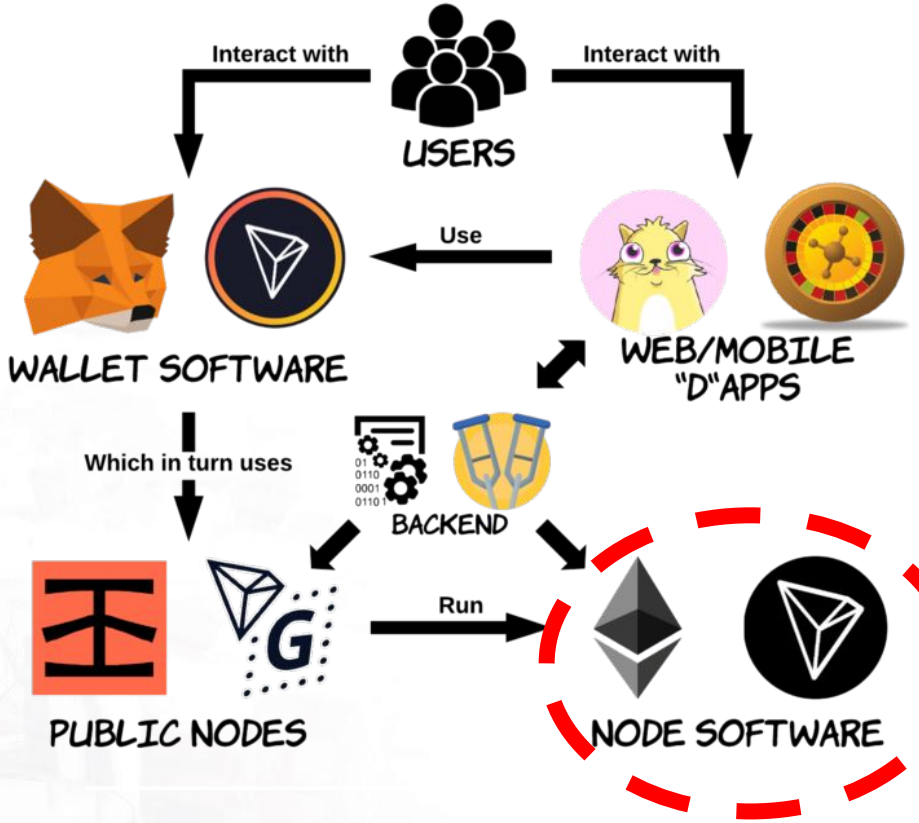

транзакция



# Blockchain. Создание блоков

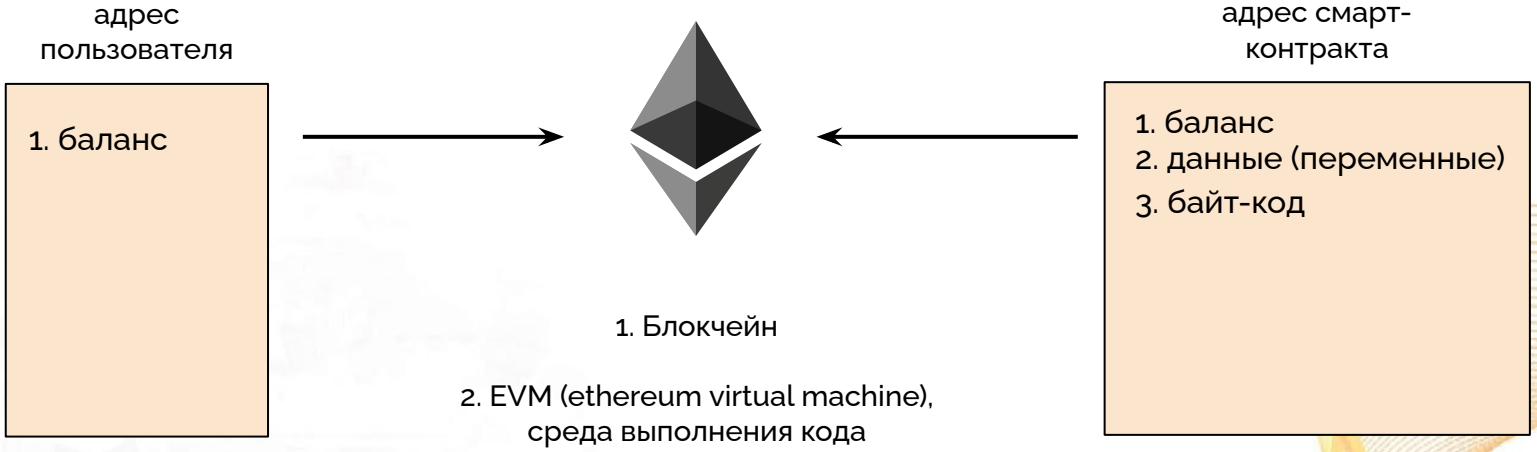


# Blockchain. Взаимодействие с пользователем




закрытая от внешнего мира система

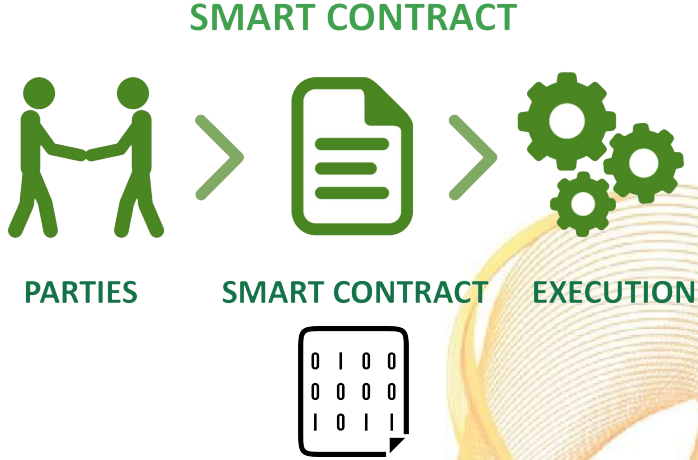
# Смарт-контракты. Что это в блокчейне?



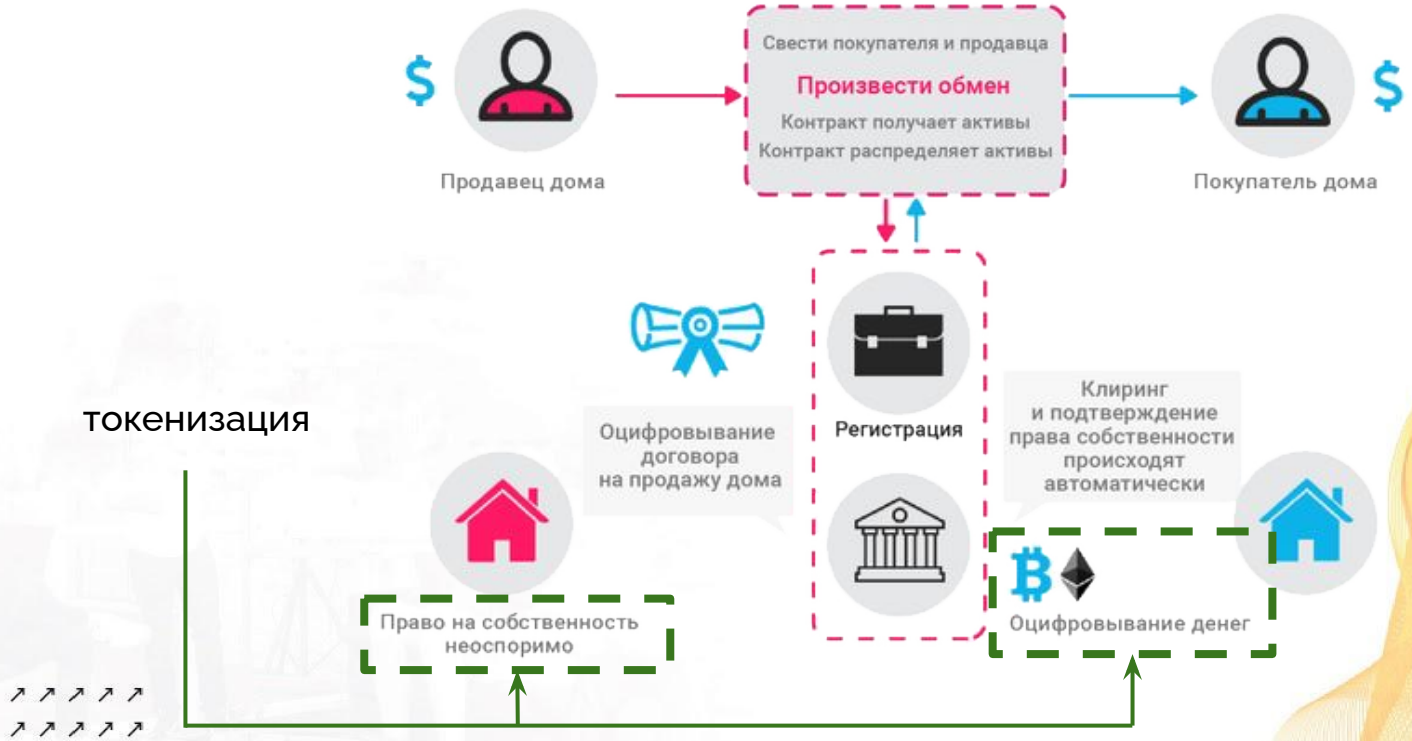

0xeb6ae6fa22d307eae06be0862087fdffdd25bab4

0x883af9f99e61cef58a12278f8fa6d39b779456de

# Смарт-контракты. Общее понятие

# Смарт-контракты. Как работают смарт-контракты?



# Смарт-контракты. Создание, деплой и исполнение

Magento Pro

# Как выглядит смарт-контракт

## Язык

Solidity (EVM)

SmartPy (Tezos)

ink! - Rust eDSL (Polkadot)

Rust,C,C++ (Solana)

и др.

## IDE и деплой

Solidity IntelliJ IDEA plugin

Remix IDE

Hardhat/truffle - фреймворки

https://ropsten.etherscan.io/address/0x883af9f99e61cef58a12278f8fa6d39b779456de#code

```sol
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.0;

contract deal_example {

    address payable public immutable OWNER;
    mapping(uint => address) public customerOrder;
    struct Order {
        address customerAddress;
        uint orderNumber;
        uint amount;
        bool isRefunded;
    }
    mapping(uint256 => Order) public orders;

    constructor() payable {
        OWNER = payable(msg.sender);
    }

    //Modifiers
    modifier onlyOwner() {
        require(payable(msg.sender) == OWNER, "Not owner");
        _;
    }

    modifier onlyCustomer(address customerAddress, uint orderNumber) {
        require(customerOrder[orderNumber] == customerAddress, "Customer does not have this order");
        _;
    }

    modifier withdraw(uint orderNumber) {
        _;
        uint amount = orders[orderNumber].amount;
        (bool success, ) = OWNER.call{value: amount}("");
        orders[orderNumber].amount = 0;
        require(success, "Failed to send Ether");
    }

    //client pay function
    function pay(uint256 orderNumber) public payable {
        orders[orderNumber] = Order(msg.sender, orderNumber, msg.value, false, false ,false);
        customerOrder[orderNumber] = msg.sender;
    }

    function withdrawNoApprove(uint256 orderNumber) public onlyOwner withdraw(orderNumber)
    {}

    function refundApprove(uint256 orderNumber) public onlyOwner {
        orders[orderNumber].isRefunded = true;
    }

    function claimMoney(uint256 orderNumber) public onlyCustomer(msg.sender, orderNumber) {
        require(orders[orderNumber].isRefunded, "Refund is not approved");
        address payable claimCustomerAddress = payable(msg.sender);
        uint256 amount = orders[orderNumber].amount;
        orders[orderNumber].amount = 0;
        (bool success, ) = claimCustomerAddress.call{value: amount}("");
        require(success, "Failed to send Ether");
    }
}

```

## Contract bytecode


```
60a06040523373ffffffffffffffffffffffffffffffffffffffff1660808173ffffffffffffffffffffffffffffffffffffffff1660601b81525050608051
60601c6116f361009f60003960008181610330a0152
818161035e0152818161041d015281816104dc0152
818161064e01.....
```


## Contract ABI

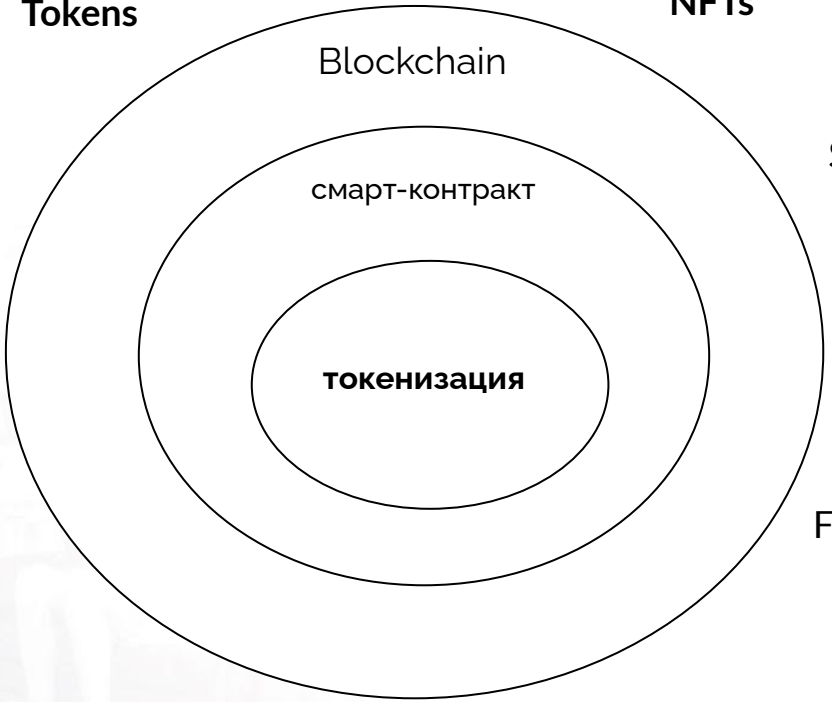

```
[{
   "inputs":[

   ],
   "stateMutability":"payable",
   "type":"constructor"
},
{
   "inputs":[

   ],
   "name":"OWNER",
   "outputs":[
     {
       "internalType":"address payable",
       "name":"",
       "type":"address"
     }
   ],
   "stateMutability":"view",
   "type":"function"
},
{
   "inputs":[
     {
       "internalType":"uint256",
       "name":"orderNumber",
       "type":"uint256"
     }
   ],
```

# Смарт-контракты. Преимущества и недостатки

- Независимость. Без посредников
- Данные о сделке
- Неизменяемость
- Прозрачность

- Комиссия за деплой и использование
- Ошибки в коде
- Сложность описания условий

# Токенизация




DAO

DeFi

GameFi

и др.

Swapping

dApp

Metaverse

Farming

# Токенизация. Примеры возможностей в eCommerce

## ERC-20/ERC-777*

взаимозаменяемый

- балы и бонусы
- валюта виртуального счета

## ERC-721 - NFTs

не взаимозаменяемый

- дисконт карты
- цифровой товар
- отражение права владения на товар

*токен (erc-20) - число в смарт-контракте, НЕ криптовалюта

# Применение блокчейн в Magento


Оплаты

бронирование товара (смарт-контракты)


регистрация и авторизация

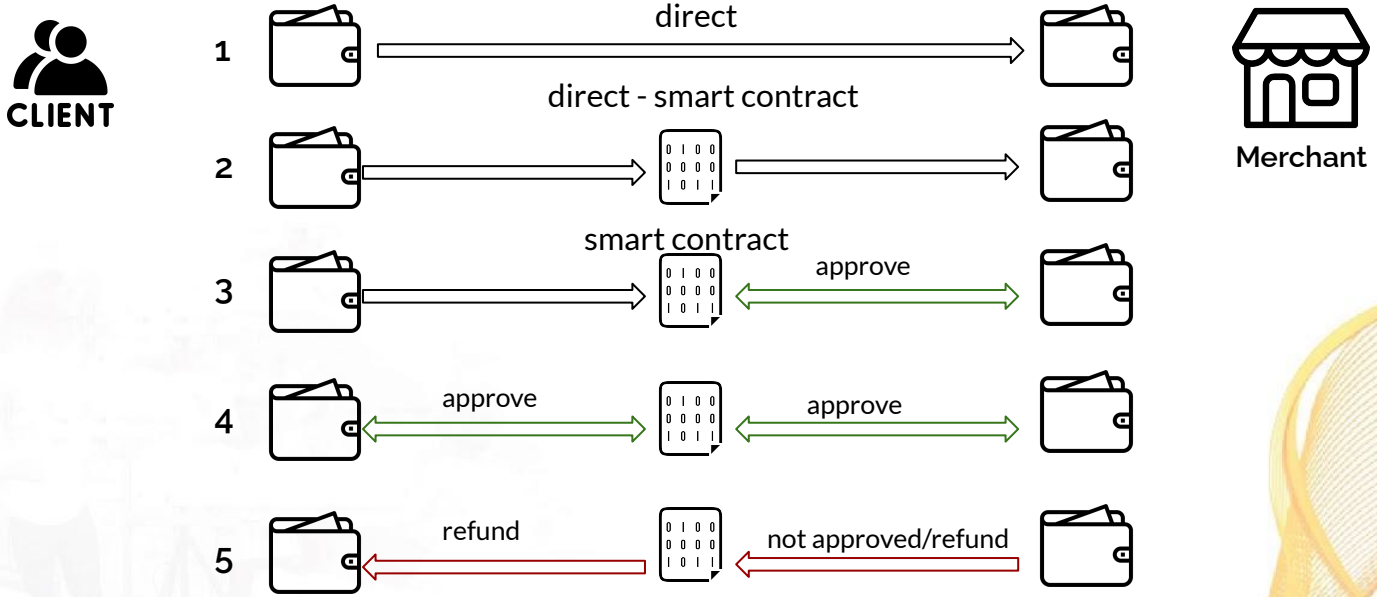

Дисконт карты (NFTs)


аукционы и розыгрыши (смарт-контракты)

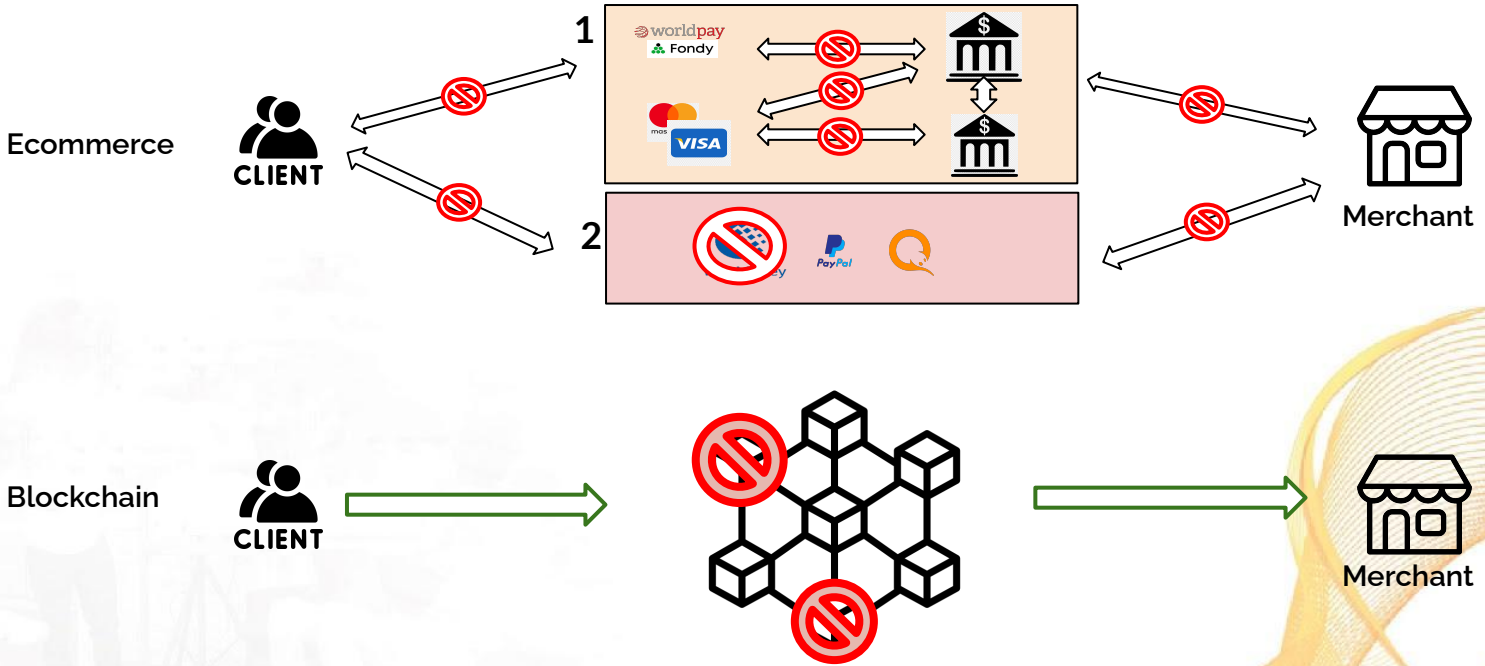

Баллы и бонусы (tokens, erc-20)

блокчейн кошельки

# Способы реализации блокчейн оплаты

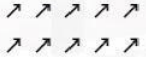


# Чем блокчейн оплата лучше



ProMagento

# Спасибо за внимание



Magento Pro

# Материалы

https://etherscan.io/ - сервис проверки транзакций

https://ropsten.etherscan.io/ - сервис проверки транзакций для тестовой сети Ropsten

https://ethervm.io/decompile - декомпилятор смарт-контрактов

https://ipfs.io/ - децентрализованное хранилище файлов

https://infura.io/ - infura - публичная нода

https://hardhat.org/ - фреймворк, среда разработки ethereum

https://remix.ethereum.org/ - Remix IDE - тестирование и деплой смарт-контрактов

https://www.openzeppelin.com/ - библиотека хелперов, интерфейсов для создания смарт-контрактов

https://github.com/OpenZeppelin/openzeppelin-contracts

https://trufflesuite.com/ - фреймворк, среда разработки ethereum

# Материалы

https://etherscan.io/ - сервис проверки транзакций

https://ropsten.etherscan.io/ - сервис проверки транзакций для тестовой сети Ropsten

https://ethervm.io/decompile - декомпилятор смарт-контрактов

https://ipfs.io/ - децентрализованное хранилище файлов

https://infura.io/ - infura - публичная
нода

https://hardhat.org/ - фреймворк, среда разработки ethereum

https://remix.ethereum.org/ - Remix IDE - тестирование и деплой смарт-контрактов

https://www.openzeppelin.com/ - библиотека хелперов, интерфейсов для создания смарт-контрактов
https://github.com/OpenZeppelin/openzeppelin-contracts

https://trufflesuite.com/ - фреймворк, среда разработки ethereum

# Материалы

https://www.hyperledger.org/learn/publications/walmart-case-study - hyperledger case

https://www.rbc.ru/crypto/news/61a4b3ed9a794781ff8b184f - токенизация

https://www.youtube.com/watch?v=l7cDRqsU61U - BlockchainUA. Как использовать Hyperledger Fabric

https://coinguides.org/evm-blockchains-add-evm-network/ - EVM блокчейны

# Материалы. Смарт-контракты

https://docs.soliditylang.org/ - Solidity документация

https://solidity-by-example.org/ - Solidity на примерах

https://cryptozombies.io/ru/ - изучение Solidity, на примере создания простой игры

https://www.youtube.com/watch?v=8A8-7Ks26yY&list=PLWlFXymvoaJ_0ok740kLXTn5qn-i1UnYr - плейлист обучения Solidity

https://www.youtube.com/c/YuliyaBedrosova - разбор некоторых смарт-контрактов, разбор скам смарт-контрактов

https://www.youtube.com/c/DappUniversity - канал о веб3, уроки, теория, практика

https://ethereum.github.io/yellowpaper/paper.pdf - стр.27 - стоимость операция в EVM

# Материалы. Токены

https://ethereum.org/en/developers/docs/standards/tokens/ - стандарты токенов

https://docs.openzeppelin.com/contracts/4.x/wizard - сервис создания смарт-контрактов токенов

https://github.com/OpenZeppelin/openzeppelin-contracts/blob/v4.6.0/contracts/token/ERC20/IERC20.sol - интерфейс ERC-20

https://github.com/OpenZeppelin/openzeppelin-contracts/blob/v4.6.0/contracts/token/ERC20/ERC20.sol - базовый контракт ERC-20

https://github.com/OpenZeppelin/openzeppelin-contracts/blob/v4.6.0/contracts/token/ERC721/IERC721.sol - интерфейс ERC-721

https://github.com/OpenZeppelin/openzeppelin-contracts/blob/v4.6.0/contracts/token/ERC721/ERC721.sol - базовый контракт ERC-721 NFTs