

Ethereum & DeFi for JS/TS developer

ARTEM VOROBEV

1inch Network

План доклада:

- Посмотрим что такое Ethereum, EVM и децентрализованные финансы (DeFi) и как с ними работать
- Сделаем пару транзакций через BscScan и Metamask
- Разобремся как устроен фронтенд децентрализованных приложений (dApp)
- Напишем код

Обо мне

- Шел по наклонной C++ → C# → JS/TS → Ethereum & DeFi
- Попал в Ethereum через ETH Global хакатоны
- Разрабатываю под Ethereum 4ре года
- Core Contributor 1inch Network

1inch Network - начало

- Основан инженерами на хакатоне в 2019
- Single Page Application в качестве бэкенда Ethereum
- Изначально: агрегация 3 - 5 децентрализованных бирж (DEX) на сети Ethereum,

1inch Network - сейчас

- ~ 70% трафика в своей нише
- Google flight для обменов в сети Ethereum, и не только
- Экосистема протоколов / смарт контрактов

7d Volume 1inch Stats

 @k06a

\$2,717,689,823

7d Volume



24h Volume 1inch Stats

 @k06a

\$359,297,386

24h Volume



Total Swaps 1inch Stats

 @k06a

3,951,749

Total Swaps



Total Users 1inch Stats

 @k06a

856,856

Total Users



Это может быть интересно, потому что в DeFi:

- Работа с финансами идет в permissionless экосистеме
- Разработчики это ключевые лица
- Высокая зарплата и еще большая личная ответственность



**DeFi - смотреть могут не только лишь
все, мало кто может это делать**

Quick intro в Ethereum и DeFi

Концепт Ethereum — виртуальная машина поверх блокчейна

1. Транзакции запускают код в машине
2. Код меняет **state** машины
3. **state** хранит что угодно

Ethereum — за все надо платить эфиrom

1. MOV ADD XOR AND ... у всего есть цена в Gas Units
2. Лимит элементарных операций на тразакцию и блок
3. Концепт газа: Gas Units * Gas Price (п)

APPENDIX G. FEE SCHEDULE

The fee schedule G is a tuple of scalar values corresponding to the relative costs, in gas, of a number of abstract operations that a transaction may effect.

Name	Value	Description
G_{zero}	0	Nothing paid for operations of the set W_{zero} .
G_{jumpdest}	1	Amount of gas to pay for a JUMPDEST operation.
G_{base}	2	Amount of gas to pay for operations of the set W_{base} .
G_{verylow}	3	Amount of gas to pay for operations of the set W_{verylow} .
G_{low}	5	Amount of gas to pay for operations of the set W_{low} .
G_{mid}	8	Amount of gas to pay for operations of the set W_{mid} .
G_{high}	10	Amount of gas to pay for operations of the set W_{high} .
$G_{\text{warmaccess}}$	100	Cost of a warm account or storage access.
$G_{\text{coldaccountaccess}}$	2600	Cost of a cold account access.
$G_{\text{coldsload}}$	2100	Cost of a cold storage access.
G_{sset}	20000	Paid for an SSTORE operation when the storage value is set to non-zero from zero.

- Газ прайс - наглядность (приложить дорогую транзакцию) и Yellow paper

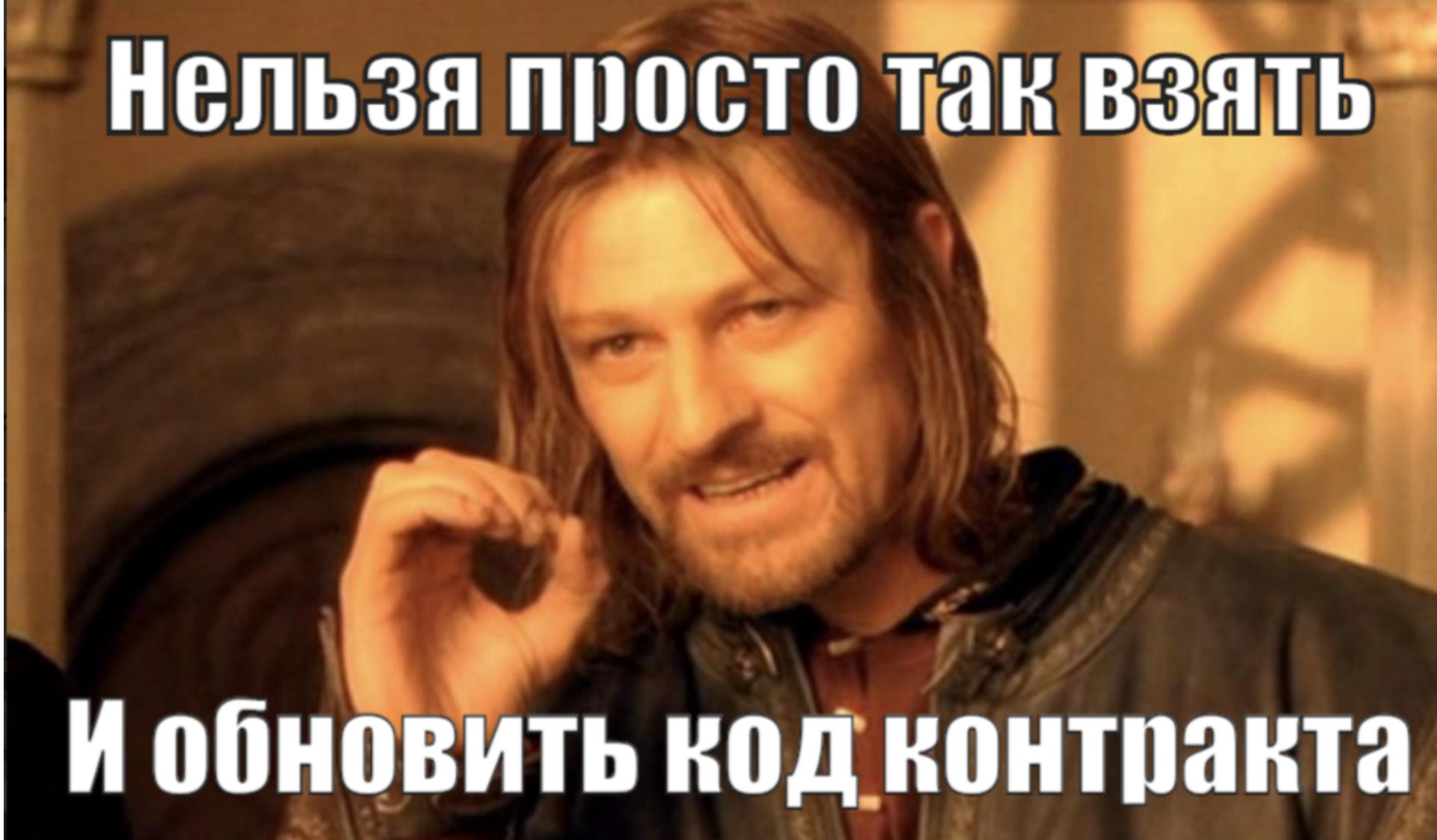
Смарт контракт

- В виртуальную машину можно деплоить программы
- Программа = исполняемый код + состояние
- Это и есть смарт контракт

Код это закон

- Программа делает только то, что в ней запрограммировано
- Программу можно верифицировать - скомпилировать исходный код и сравнить
- Пример, контракт токена [WBTC](#)

А как изменить код контракта?



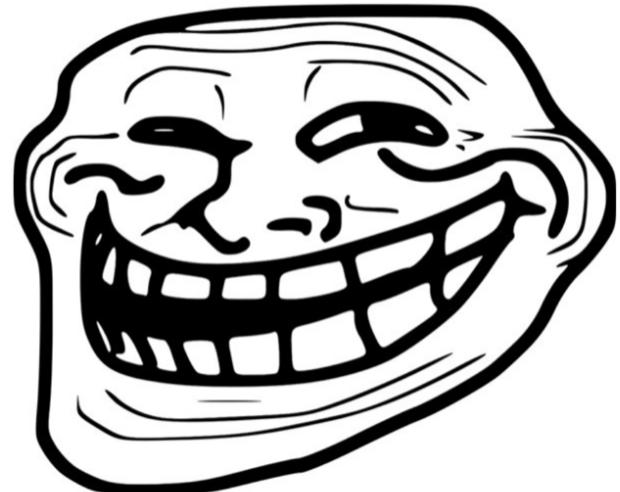
Нельзя просто так взять

И обновить код контракта

Можно:

- Использовать **Proxy** паттерн
- Написать **Upgradable** контракт
- Или даже вызвать деструктор (**selfdestruct**)
- Примеры **OpenZeppelin**

Но ты не написал



ЭТОТ КОД В КОНТРАКТЕ

И это даже хорошо

- Код это закон
- Вопрос доверия

А как понять что контракт обновили ?

А как понять что контракт обновили ?

- Очень просто, у нас же блокчейн
- Можно найти на [etherscan](#)

Если провести аналогию блокчейн и Git:

- Транзакции — коммиты
- Блоки — merge pull request'а в мастер
- У вас нет прав на мастер 😊
- Контракт и транзакции останутся в истории навсегда

Погружаемся в DeFi



Смарт контракты могут работать сообща

1. Пользователь отправляет транзакцию
2. Транзакция запускает метод на контракте
3. Метод контракта может вызвать метод другого контракта

**Что бы контракты были совместимы нужны
интерфейсы**

ERC-20 токены USDT, DAI

**Пример, разрешим контракту 1inch потратить потратить
10,000 BUSD моего кошелька**

Контракты - обменники (AMM, Liquidity Pool)

Пример [uniswap](#)

Обменивает токены по формуле $x * y = \text{const}$

Пример транзакции

Контракты ломбарды

Кредиты в одном токене под залог другого

Контракт может распродать обеспечение

Пример [Compound](#)

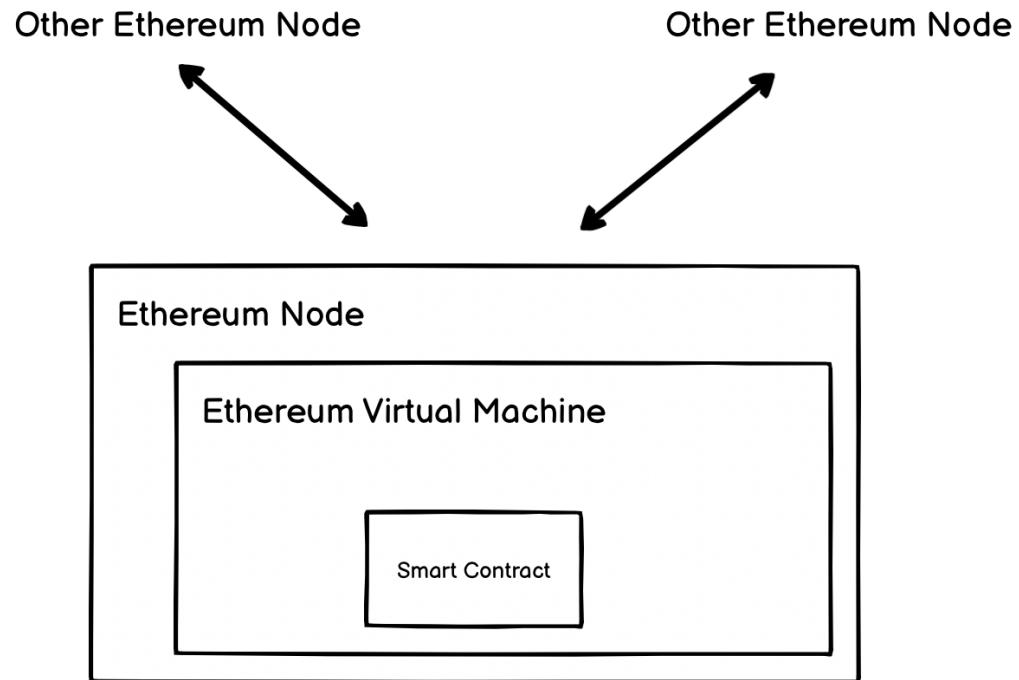
**4. Контракты Оракулы - поставщики цены/погоды/
результатов выборов, [Chainlink](#)**

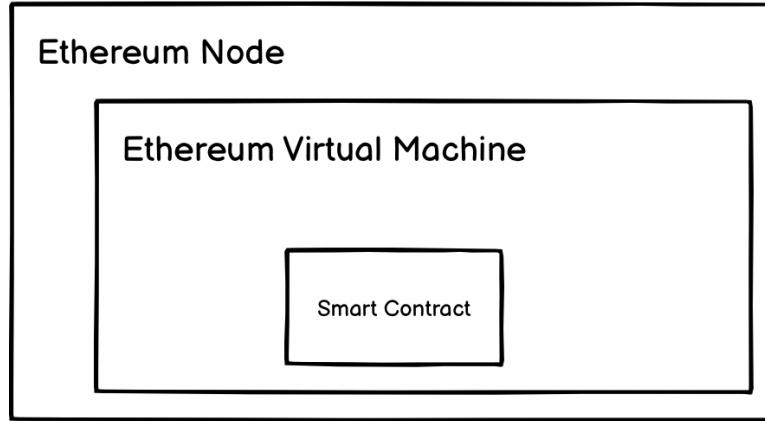
dApp - Decentralized Application

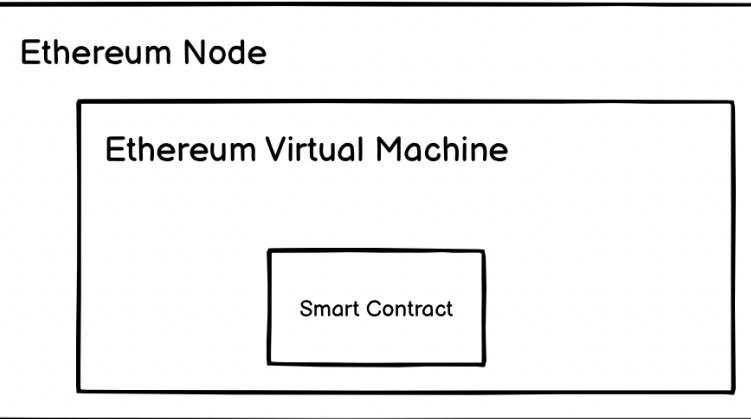
- **Uniswap,**
- **Compound**
- **1inch**, все это dApp



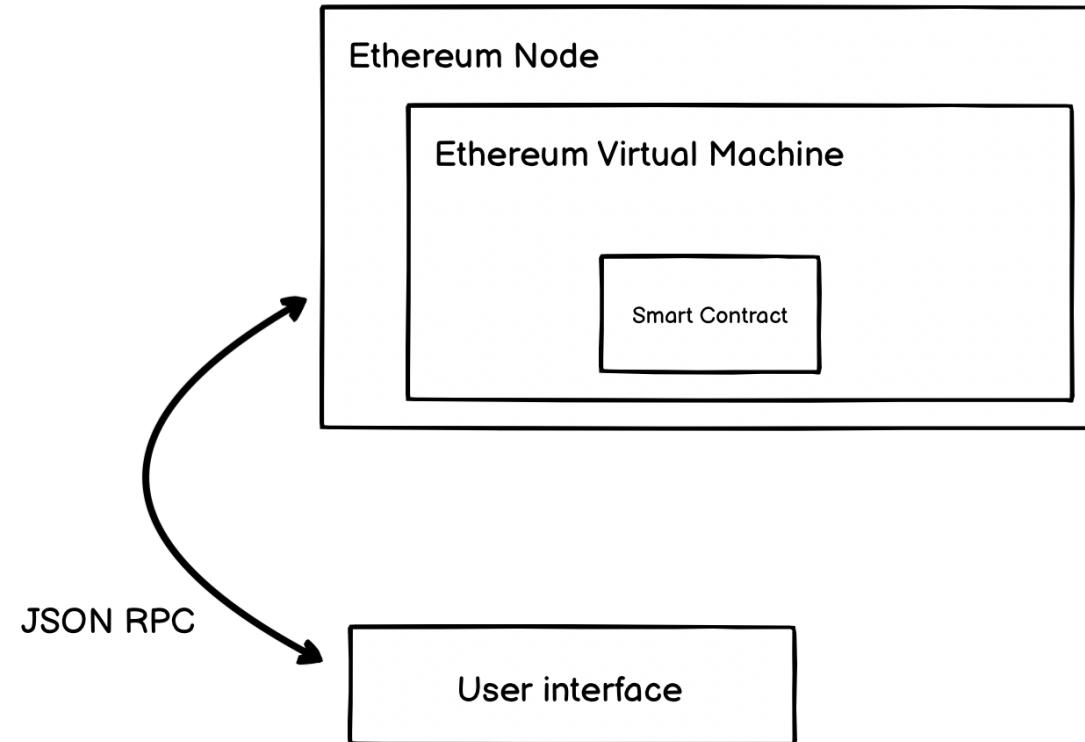
Посмотрим как устроены dApp





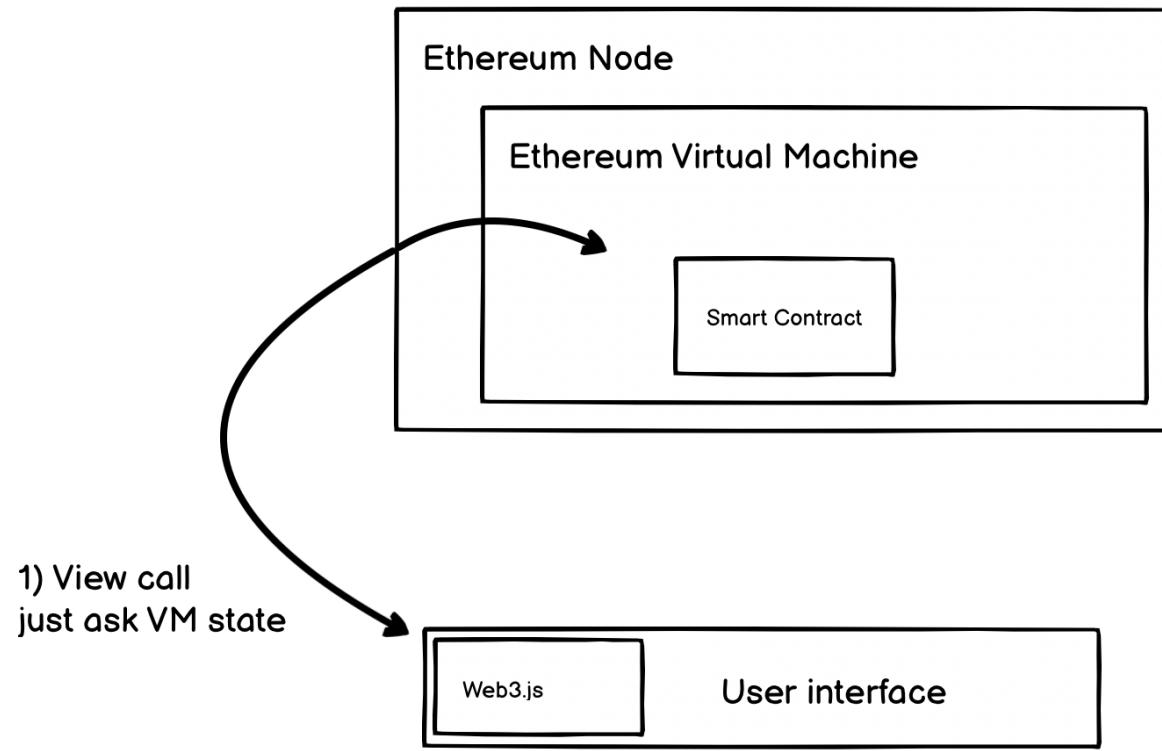


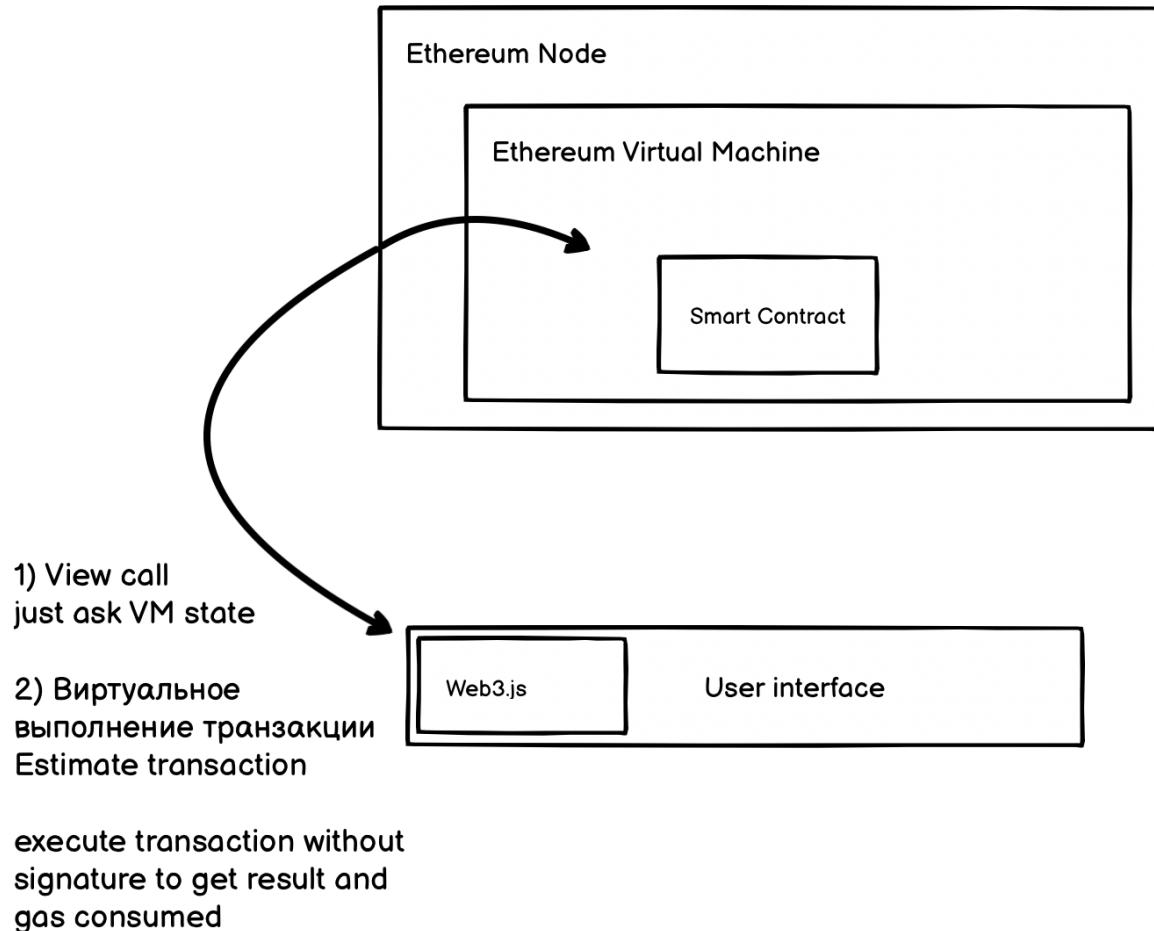
- Usually SPA hosted on CDN with fallback to IPFS
- Usually allow open source to allow everyone to verify
- The matter of trust



1) View режим

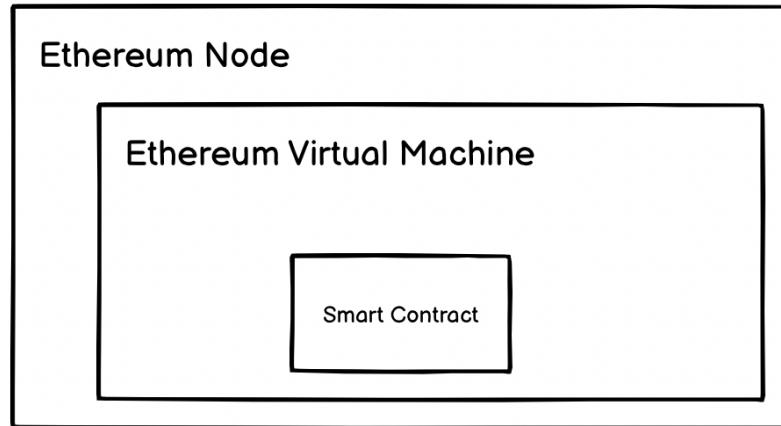
До того как кошелек подключен

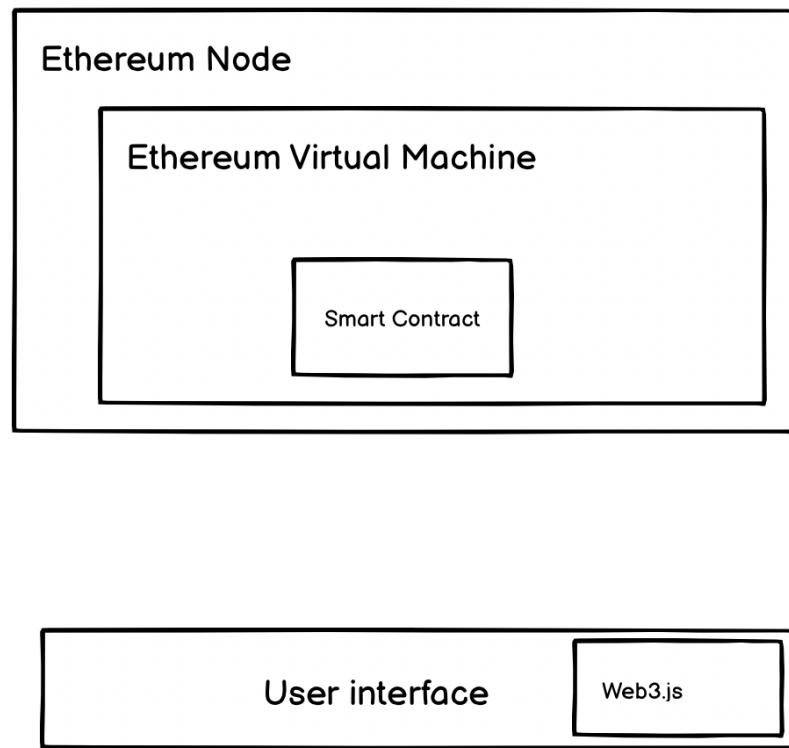




2) Write Режим

Когда кошелек подключен





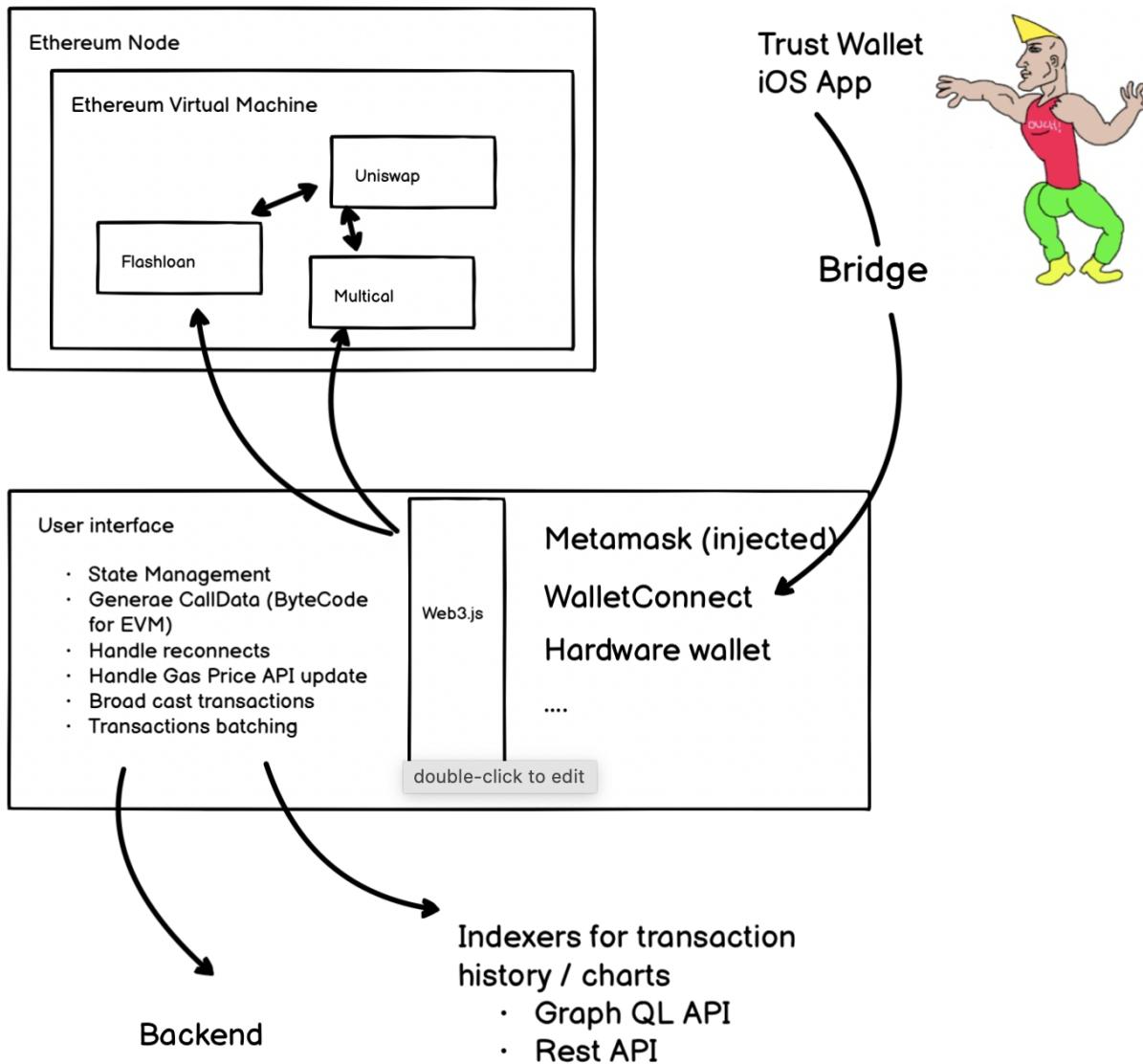
С кошельком появляются

- `SignTransaction`
- `SignAndSendTransaction`

3) Режим Reallife

- Много контрактов
- Нужно составлять сложный байт код вызовов контрактов
- Иногда нужно создавать новые контракты через фабрики
- Много провайдеров кошельков
- Появляется backend
- Некоторые данные можно достать только через индексаторы

Режим реальной жизни



4) Режим Hardcore

Режим Хардкор - Support Several networks

