# Smart Contract

# 강사 소개

- 마경욱
  - 스마일 게이트 딥러닝 연구팀 재직 중
  - 모바일/백엔드/블록체인 프로젝트 진행
  - kyeongwook.ma@gmail.com
  - https://github.com/shwarzes89

# 목차

# 개요

# 개요

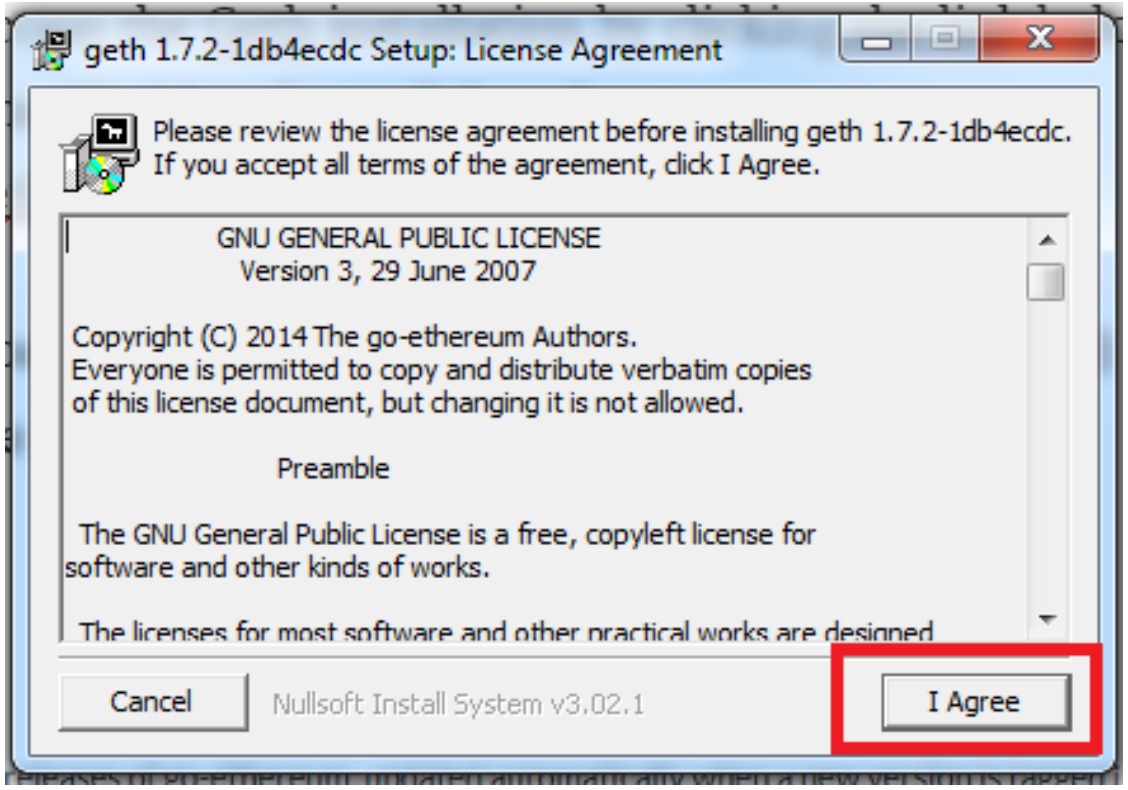
- 디지털 명령어로 계약을 작성하면 조건에 따라 계약 내용을 자동 실행
- 복잡한 프로세스를 간소화
- 조작 불가
- 즉각 이행 가능

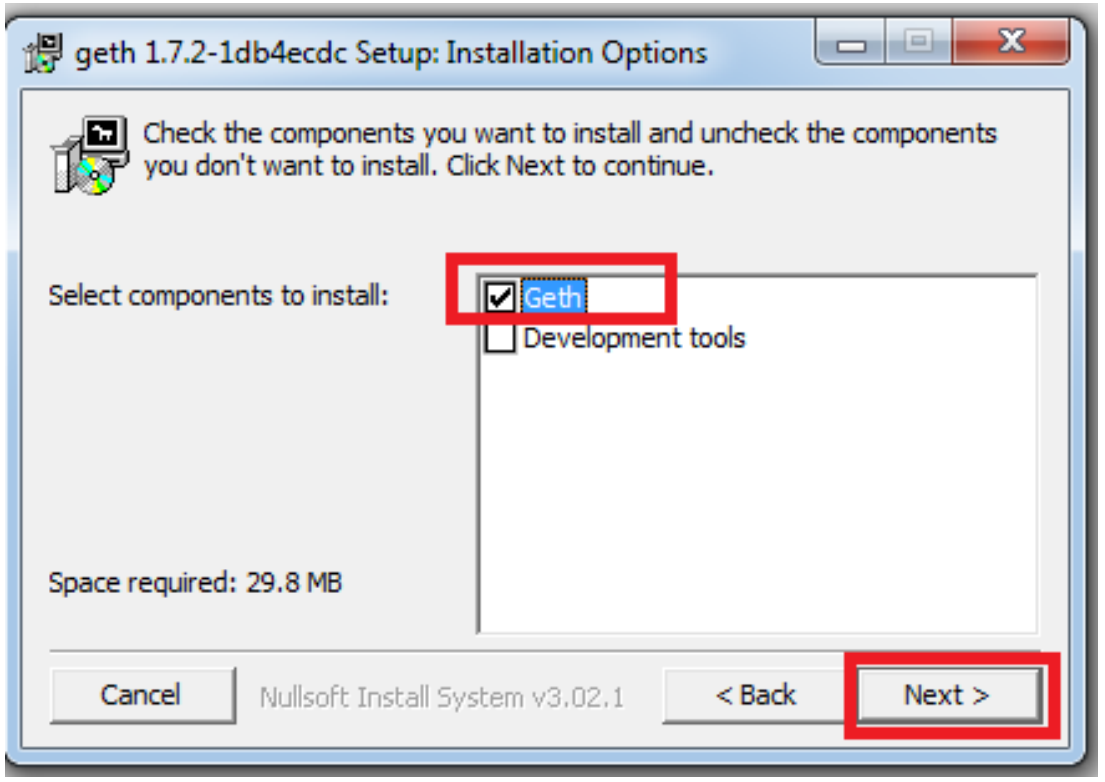
# 개발 환경 설정

# Geth 설치 및 실행

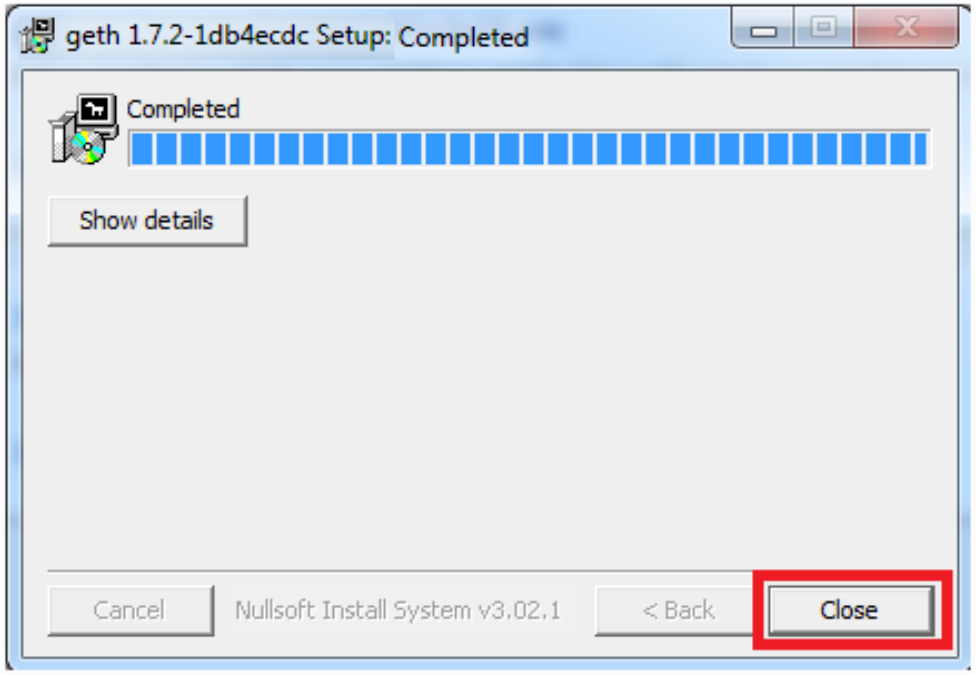
- https://geth.ethereum.org/downloads/
- Go언어로 작성된 Ethereum node
- 운영체제에 맞는 설치 파일 다운로드
- 환경 변수 설정
  1. ~/.bash_profile 수정
  2. export GETH=~/geth 추가
  3. export PATH=${PATH}:${GETH} 추가
  4. source ~/.bash_profile
- 실행

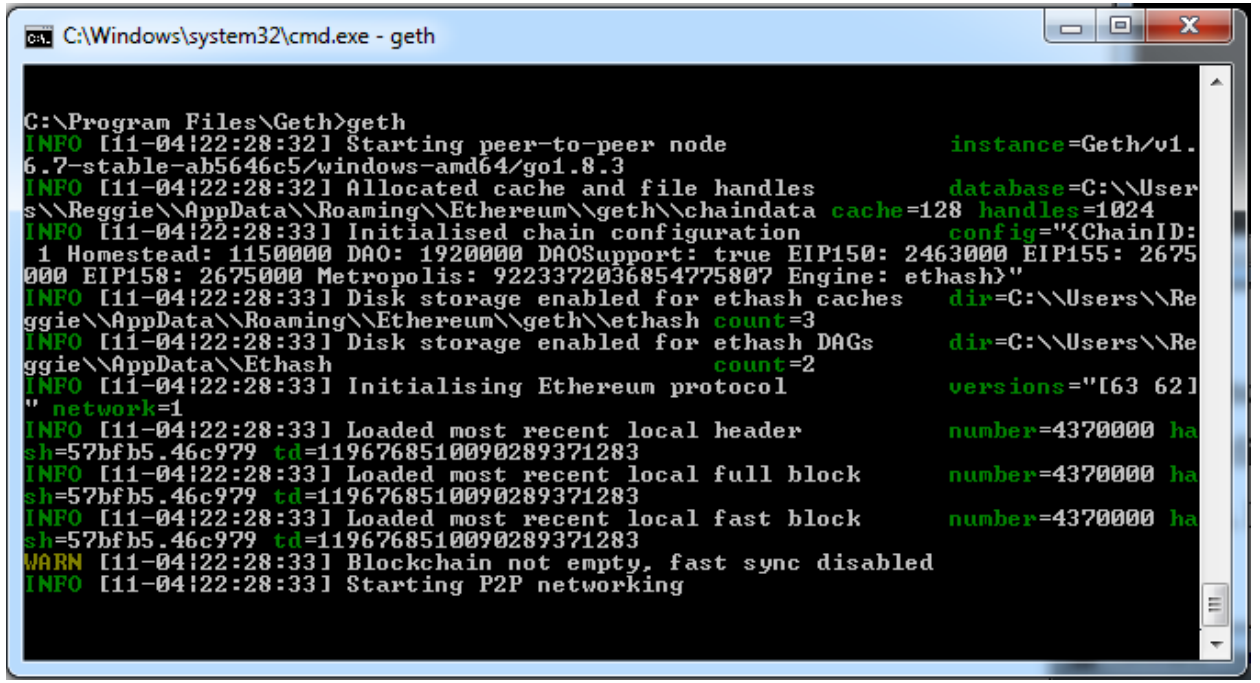
# Geth 설치 및 실행


geth 1.7.2-1db4ecdc Setup: License Agreement
Please review the license agreement before installing geth 1.7.2-1db4ecdc.
If you accept all terms of the agreement, click I Agree.
GNU GENERAL PUBLIC LICENSE
Version 3, 29 June 2007
Copyright (C) 2014 The go-ethereum Authors.
Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.
Preamble
The GNU General Public License is a free, copyleft license for
software and other kinds of works.
The licenses for most software and other practical works are designed
Cancel
Nullsoft Install System v3.02.1
I Agree


geth 1.7.2-1db4ecdc Setup: Installation Options
Check the components you want to install and uncheck the components
you don't want to install. Click Next to continue.
Select components to install:
Geth
Development tools
Space required: 29.8 MB
Cancel
Nullsoft Install System v3.02.1
< Back
Next >

# Geth 설치 및 실행


geth 1.7.2-1db4ecdc Setup: Completed
Completed
Show details
Cancel
Nullsoft Install System v3.02.1
< Back
Close


C:\Windows\system32\cmd.exe - geth
C:\Program Files\Geth>geth
INFO [11-04|22:28:32] Starting peer-to-peer node instance=Geth/v1.6.7-stable-ab5646c5/windows-amd64/go1.8.3
INFO [11-04|22:28:32] Allocated cache and file handles database=C:\\Users\\Reggie\\AppData\\Roaming\\Ethereum\\geth\\chaindata cache=128 handles=1024
INFO [11-04|22:28:33] Initialised chain configuration config="{ChainID: 1 Homestead: 1150000 DAO: 1920000 DAOSupport: true EIP150: 2463000 EIP155: 2675000 EIP158: 2675000 Metropolis: 9223372036854775807 Engine: ethash}"
INFO [11-04|22:28:33] Disk storage enabled for ethash caches dir=C:\\Users\\Reggie\\AppData\\Roaming\\Ethereum\\geth\\ethash count=3
INFO [11-04|22:28:33] Disk storage enabled for ethash DAGs dir=C:\\Users\\Reggie\\AppData\\Ethash count=2
INFO [11-04|22:28:33] Initialising Ethereum protocol versions="[63 62]" network=1
INFO [11-04|22:28:33] Loaded most recent local header number=4370000 hash=57bfb5..46c979 td=11967685100902893712 83
INFO [11-04|22:28:33] Loaded most recent local full block number=4370000 hash=57bfb5..46c979 td=11967685100902893712 83
INFO [11-04|22:28:33] Loaded most recent local fast block number=4370000 hash=57bfb5..46c979 td=11967685100902893712 83
WARN [11-04|22:28:33] Blockchain not empty, fast sync disabled
INFO [11-04|22:28:33] Starting P2P networking

# Private blockchain network 개설

1. mkdir private_chain

2. mkdir private_chain/chaindata && cd private_chain/chaindata

3. vim **genesis.json**
    1. Copy https://gist.github.com/shwarzes89/fffbc91cbce71f7161550202e21185d3
    2. Paste

# Private blockchain network 개설

- Network 초기화
  - geth --datadir=./chaindata/ init ./genesis.json

```
INFO [03-02|09:55:49] Maximum peer count                       ETH=25 LES=0 total=25
INFO [03-02|09:55:49] Allocated cache and file handles         database=/Users/user/private_chain/chaindata/geth/chaindata cache=16 handles=16
INFO [03-02|09:55:49] Writing custom genesis block
INFO [03-02|09:55:49] Persisted trie from memory database      nodes=0 size=0.00B time=9.415µs gcnodes=0 gcsize=0.00B gctime=0s livenodes=1 livesize=0.00B
INFO [03-02|09:55:49] Successfully wrote genesis state         database=chaindata                                                hash=2fb1a7…f0181a
INFO [03-02|09:55:49] Allocated cache and file handles         database=/Users/user/private_chain/chaindata/geth/lightchaindata cache=16 handles=16
INFO [03-02|09:55:49] Writing custom genesis block
INFO [03-02|09:55:49] Persisted trie from memory database      nodes=0 size=0.00B time=2.466µs gcnodes=0 gcsize=0.00B gctime=0s livenodes=1 livesize=0.00B
INFO [03-02|09:55:49] Successfully wrote genesis state         database=lightchaindata                                           hash=2fb1a7…f0181a
```

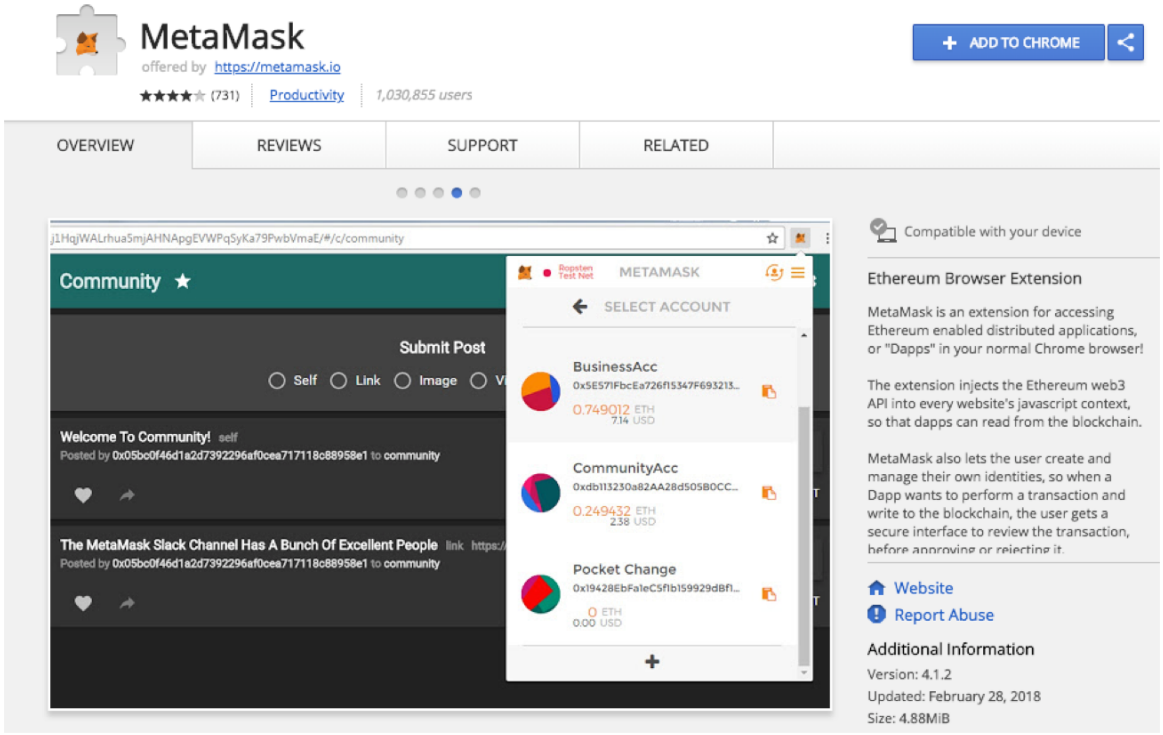
# Private blockchain network 개설

- 실행
  - geth --datadir=./chaindata/

```
INFO [03-02|10:04:38] Starting P2P networking
INFO [03-02|10:04:40] UDP listener up                          self=enode://2636ea60b4c3eb9db6241b9708b88e13cc000fab6cee67001392196d9dbdabf21038aae395c4b2735f3a4cdf501ac84b8
a0c10e4644ad757302f03bd50c74992@[::]:30303
INFO [03-02|10:04:40] RLPx listener up                         self=enode://2636ea60b4c3eb9db6241b9708b88e13cc000fab6cee67001392196d9dbdabf21038aae395c4b2735f3a4cdf501ac84b8
a0c10e4644ad757302f03bd50c74992@[::]:30303
INFO [03-02|10:04:40] IPC endpoint opened                      url=/Users/user/private_chain/chaindata/geth.ipc
```

# MetaMask

- Ethereum 브라우저 & 지갑
- Blockchian을 다운받을 필요 없이 Dapp, smart contract와 연동
- https://chrome.google.com/webstore/detail/metamask/nkbihfbeogaeaoehlefnkodbefgpgknn?hl=en



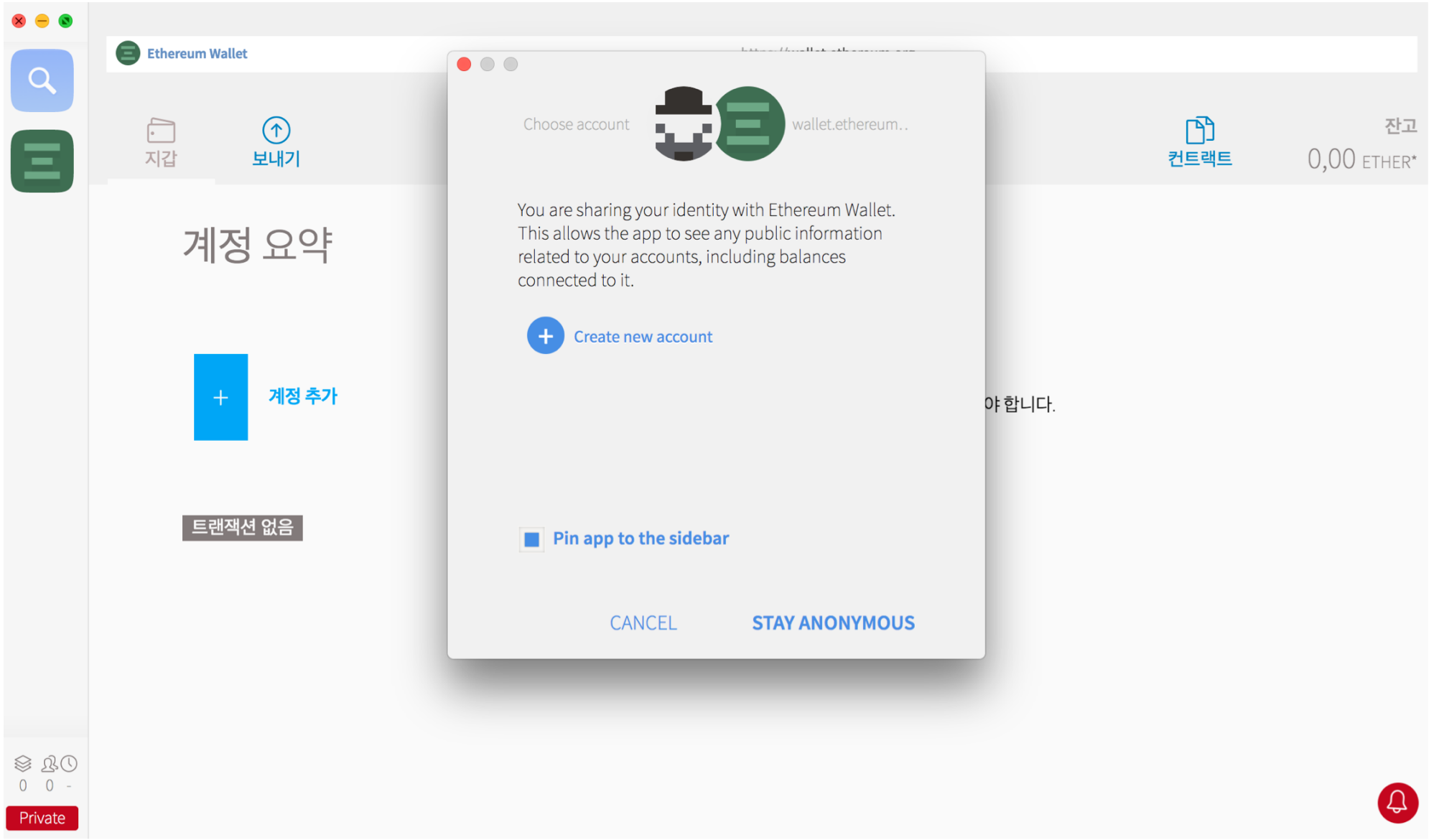
# Mist

- Ethereum GUI 지갑

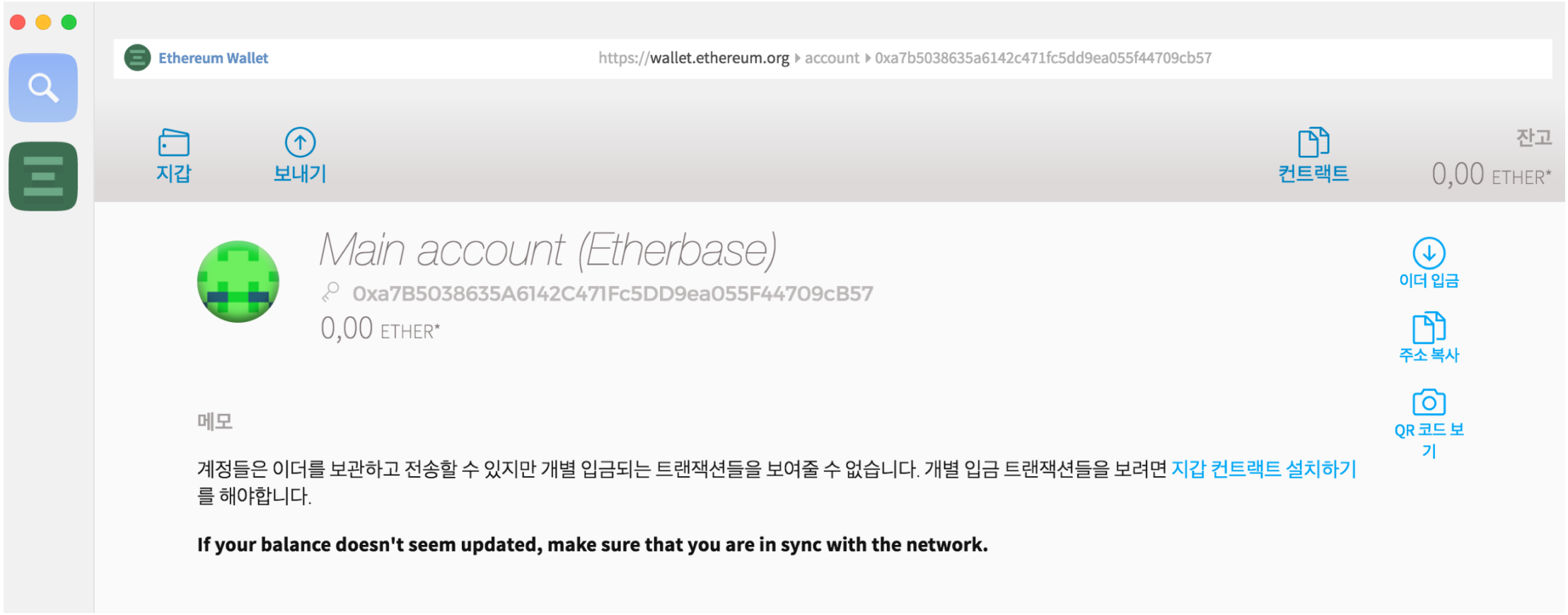
- 설치
  - https://github.com/ethereum/mist/releases

- 실행
  - Geth 실행
    - geth --datadir=./chaindata/
  - Mist 실행
    - ❖ Mac의 경우 /Applications/Mist.app/Contents/MacOS/Mist --rpc ./private_chain/chaindata/geth.ipc

# Mist

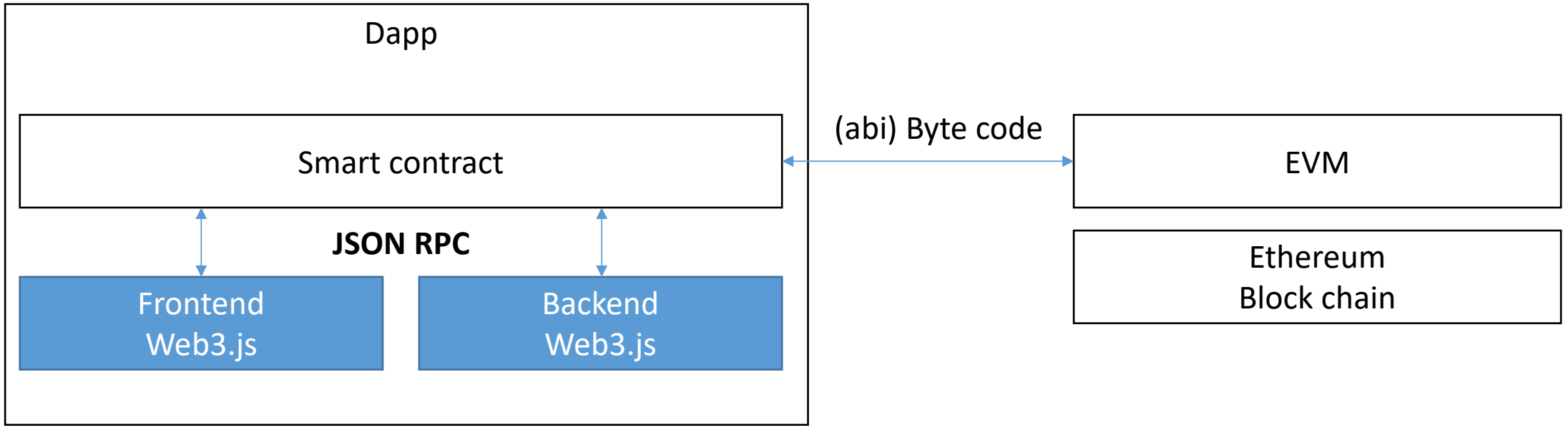

Ethereum Wallet
지갑
보내기
컨트랙트
잔고
0,00 ETHER*
계정 요약
계정 추가
트랜잭션 없음
야 합니다.
Choose account
wallet.ethereum..
You are sharing your identity with Ethereum Wallet. This allows the app to see any public information related to your accounts, including balances connected to it.
Create new account
Pin app to the sidebar
CANCEL
STAY ANONYMOUS
0 0 -
Private

# Mist


Ethereum Wallet
https://wallet.ethereum.org ▸ account ▸ 0xa7b5038635a6142c471fc5dd9ea055f44709cb57
지갑
보내기
컨트랙트
잔고
0,00 ETHER*
Main account (Etherbase)
0xa7B5038635A6142C471Fc5DD9ea055F44709cB57
0,00 ETHER*
이더 입금
주소 복사
QR 코드 보기
메모
계정들은 이더를 보관하고 전송할 수 있지만 개별 입금되는 트랜잭션들을 보여줄 수 없습니다. 개별 입금 트랜잭션들을 보려면 지갑 컨트랙트 설치하기를 해야합니다.
If your balance doesn't seem updated, make sure that you are in sync with the network.

# Web3.js

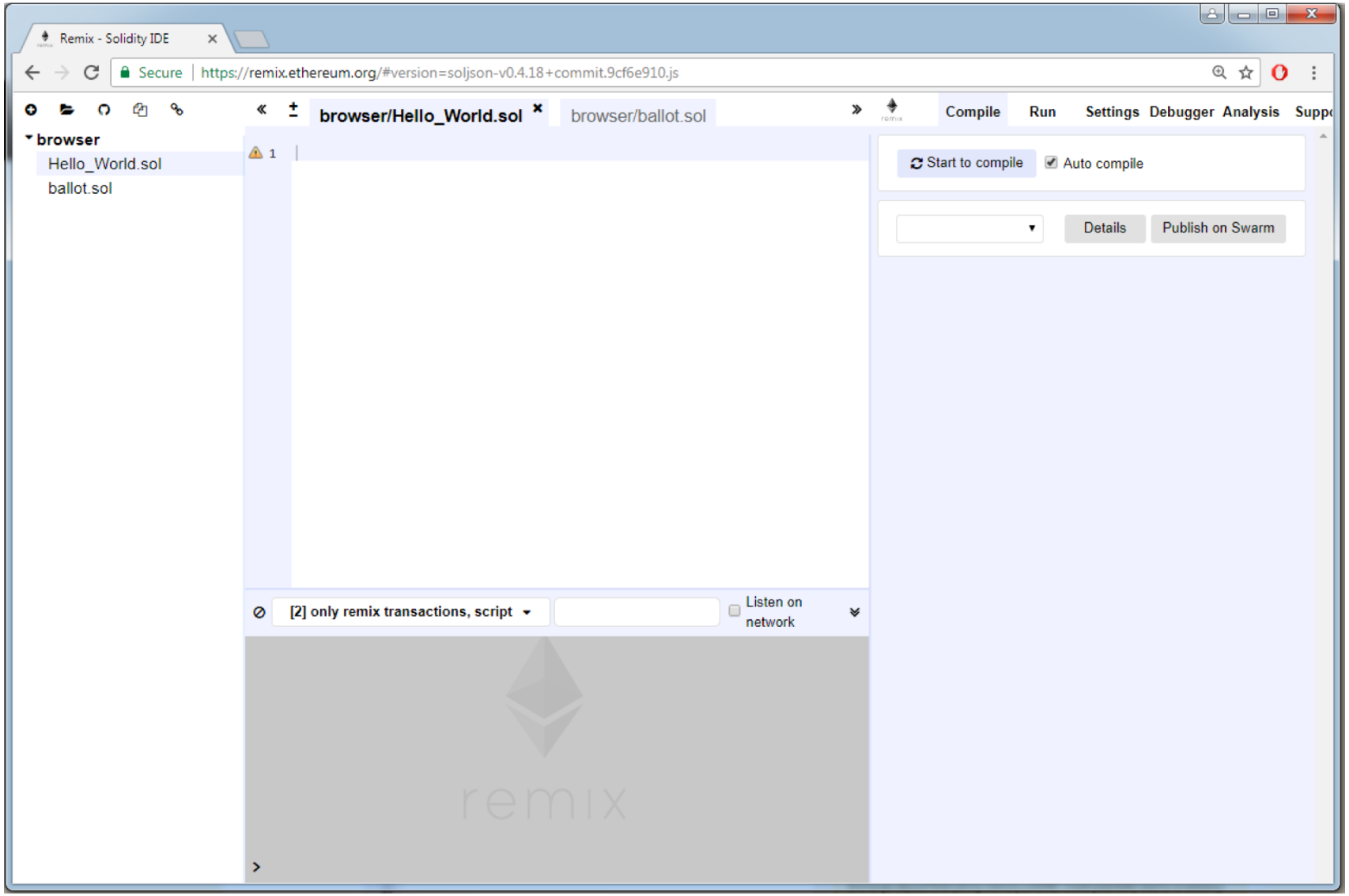
- Ethereum 노드와 브라우저간의 통신
- HTTP 나 IPC 연결을 통해 JSON RPC를 통하여 ethereum 과 통신
- https://github.com/ethereum/wiki/wiki/JavaScript-API



# Remix

- IDE (정적 분석, 구문 검사, …)
- Web3 와 연동 가능
- MetaMask 나 Mist 에 직접 배포 가능
- https://remix.ethereum.org/

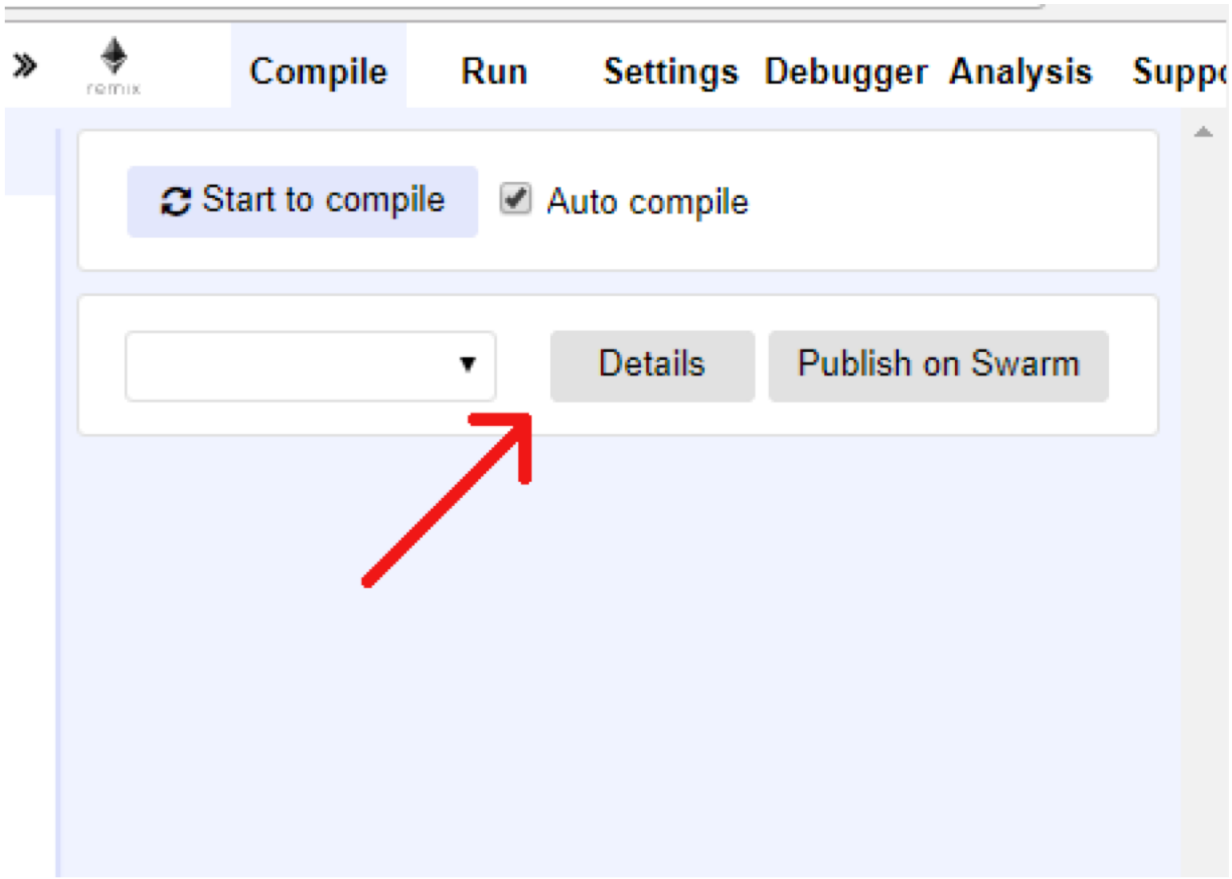
# Remix


Remix - Solidity IDE
Secure
https://remix.ethereum.org/#version=soljson-v0.4.18+commit.9cf6e910.js
browser
Hello_World.sol
ballot.sol
browser/Hello_World.sol
browser/ballot.sol
Compile
Run
Settings
Debugger
Analysis
Start to compile
Auto compile
Details
Publish on Swarm
[2] only remix transactions, script
Listen on network
remix

# Remix

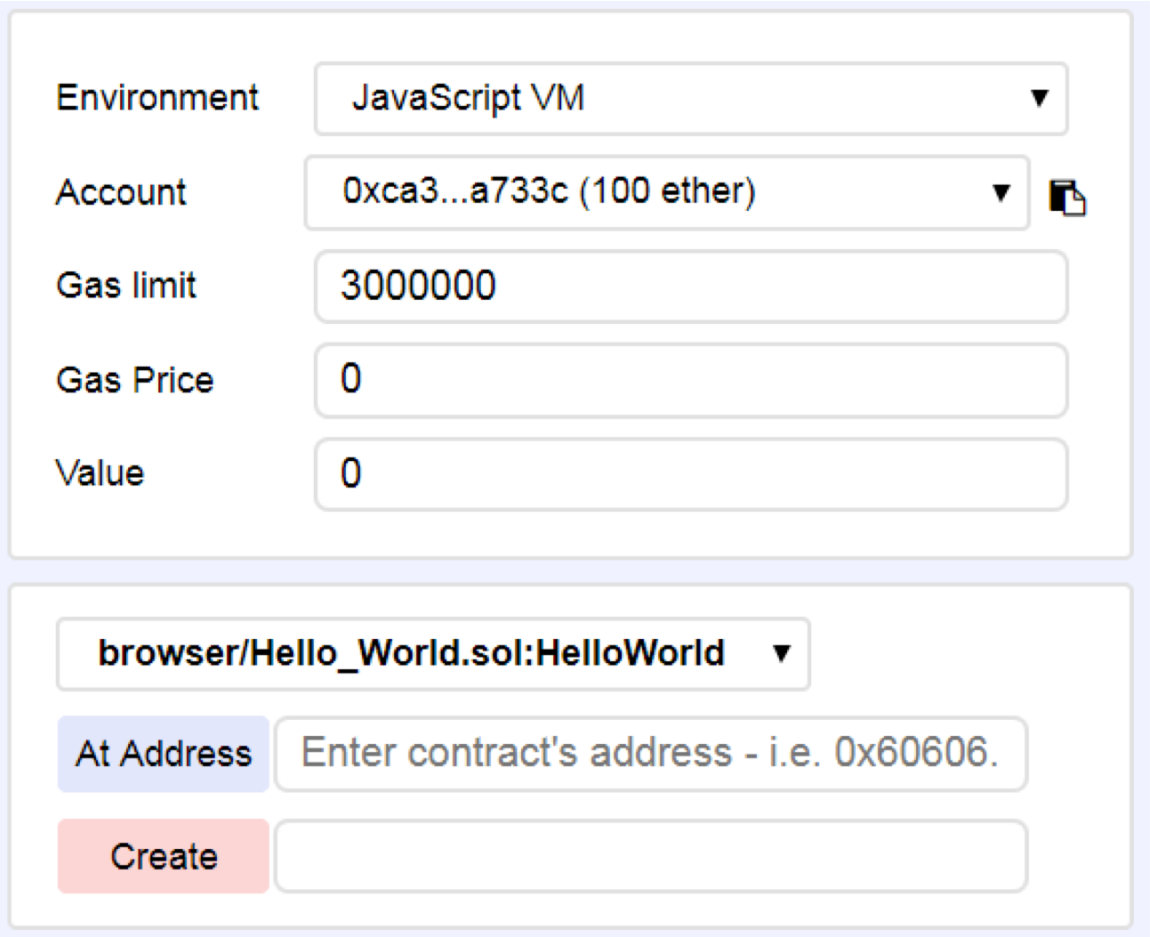
- 소스 코드 브라우져
- 소스 코드 에디터
- 스마트 컨트랙트의 컴파일 및 배포, 디버깅, 분석
  - 환경
    - Javascript VM : geth 연결 없이 모든 개발이 Remix상에서 이뤄짐
    - Injected Web3 : Mist나 Metamask와 같이 공급자에 의해 제공되는 환경
    - Web3 provider : 로컬에서 구동되는 geth 노드에 연결

# Remix




- 스마트 컨트랙트 정보 확인
  - ABI
  - Web3 배포 코드
  - 메타데이터
  - 바이트 코드

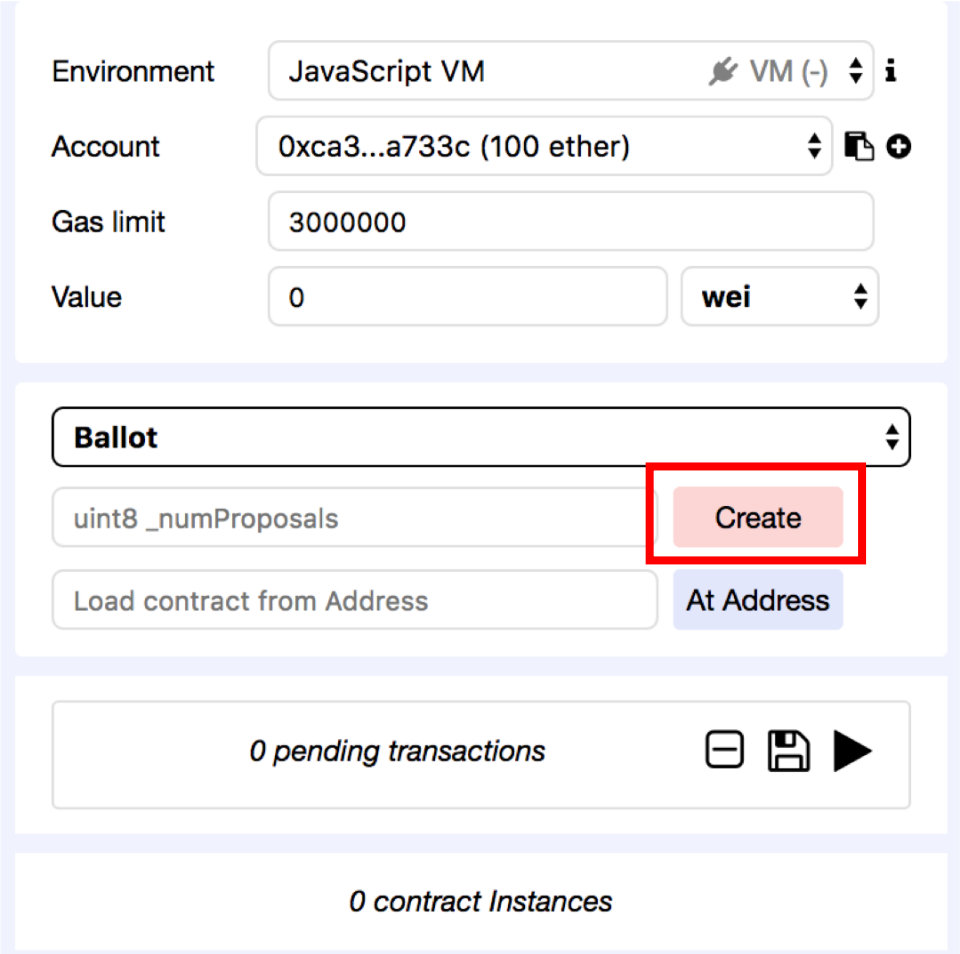
# Remix



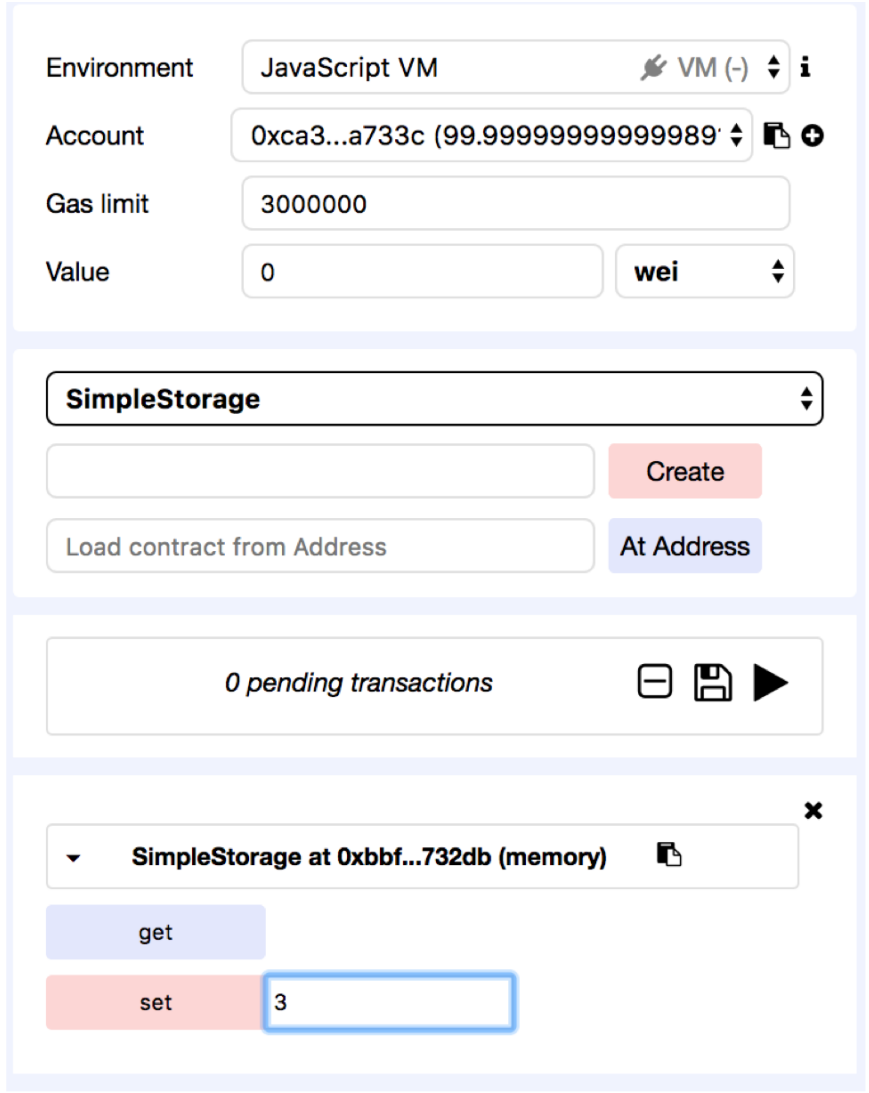

- Environment: JavaScript VM 을 이용해서 브라우저 안에서 실행
- Account: 100 ether가 있는 테스트 계정
- Gas limit: gas (사용료) 제한 설정
- Gas Price: 필요 gas 설정
- Gas : Ethereum 트랜잭션을 실행시키기 위한 수수료
  - Ether는 가상화폐이기 때문에 변동성이 있으나 Gas는 변동성이 거의 없음

# Hello World.sol

- https://gist.github.com/shwarzes89/495d65e31cdf655aff660e4ddb0a4e6d
- Copy & Paste
- "Create"



# Hello World.sol




- 임의의 값 설정

# Hello World.sol

```
creation of SimpleStorage pending...

[vm] from:0xca3...a733c, to:SimpleStorage.(constructor), value:0 wei, data:
0x606...d0029, 0 logs, hash:0x85c...39126
```

Details Debug

```
transact to SimpleStorage.set pending ...

[vm] from:0xca3...a733c, to:SimpleStorage.set(uint256) 0xbbf...732db, valu
e:0 wei, data:0x60f...00003, 0 logs, hash:0x6d2...d0532

call to SimpleStorage.get

[call] from:0xca35b7d915458ef540ade6068dfe2f44e8fa733c, to:SimpleStorage.ge
t(), data:6d4ce...ce63c, return:
{
        "0": "uint256: 3"
}
```

Details Debug

Details Debug

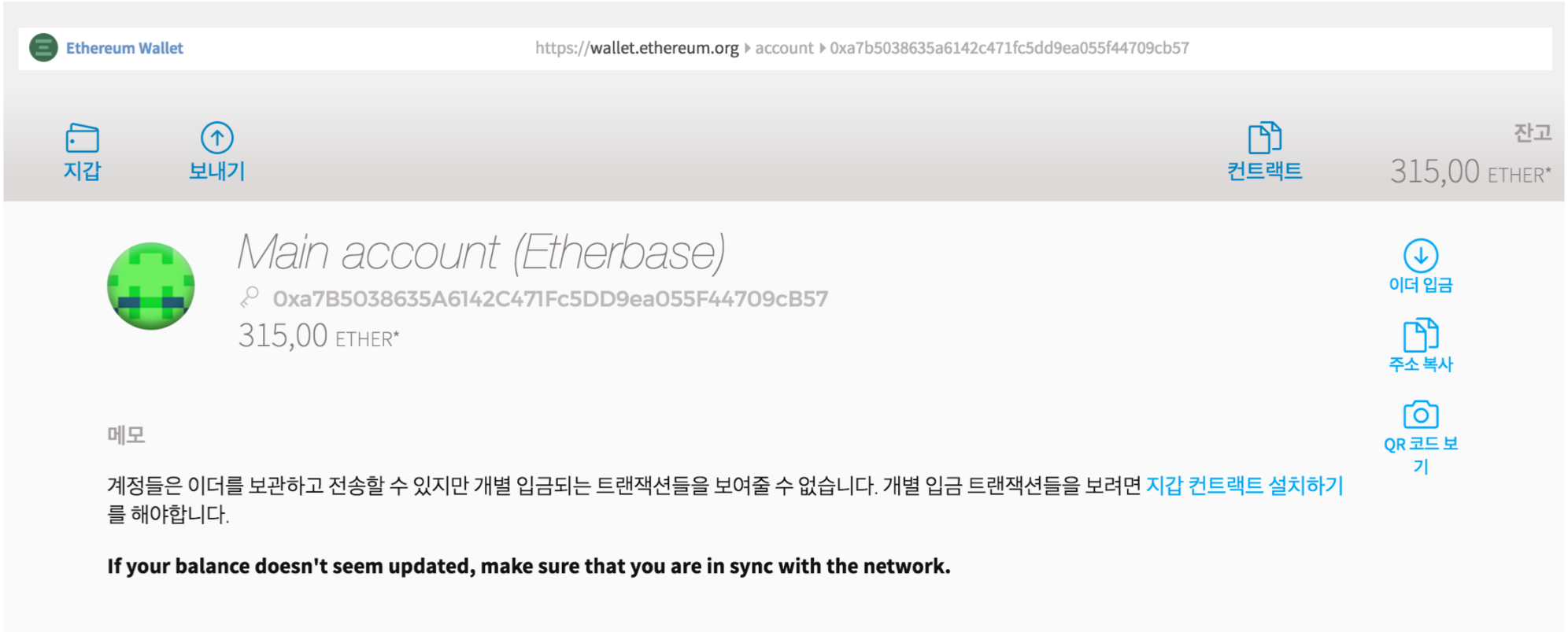
# Mist에 스마트 컨트랙트 배포

- **스마트 컨트랙트를 사용하기 전 ether 필요**
  - geth --datadir=./chaindata/ console 명령어로 geth 에 접근
  - miner.start(); 로 채굴 시작

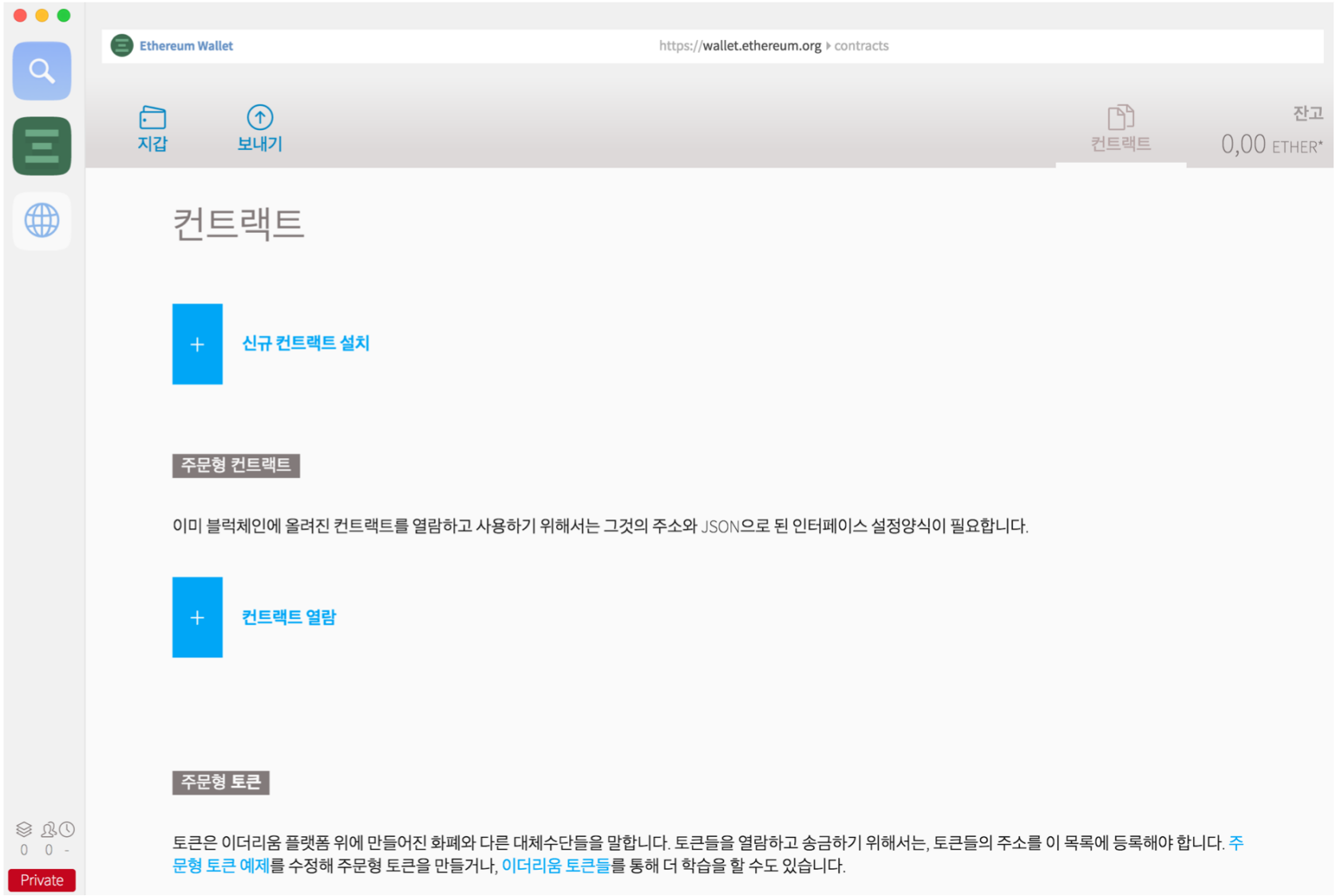
```
INFO [03-07|08:41:34] 🔗block reached canonical chain          number=57 hash=1e4b72…100cb0
INFO [03-07|08:41:34] 🔨mined potential block                  number=62 hash=285f5b…1c84a9
INFO [03-07|08:41:34] Commit new mining work                   number=63 txs=0 uncles=0 elapsed=118.134µs
INFO [03-07|08:41:34] Successfully sealed new block            number=63 hash=8fdd79…0b167d
INFO [03-07|08:41:34] 🔗block reached canonical chain          number=58 hash=d9f203…14861b
INFO [03-07|08:41:34] 🔨mined potential block                  number=63 hash=8fdd79…0b167d
INFO [03-07|08:41:34] Commit new mining work                   number=64 txs=0 uncles=0 elapsed=123.479µs

true
> INFO [03-07|08:41:34] Generating DAG in progress               epoch=1 percentage=95 elapsed=1m7.025s
INFO [03-07|08:41:34] Generating DAG in progress               epoch=1 percentage=96 elapsed=1m7.507s
INFO [03-07|08:41:35] Generating DAG in progress               epoch=1 percentage=97 elapsed=1m7.983s
INFO [03-07|08:41:35] Generating DAG in progress               epoch=1 percentage=98 elapsed=1m8.462s
INFO [03-07|08:41:36] Generating DAG in progress               epoch=1 percentage=99 elapsed=1m9.540s
INFO [03-07|08:41:36] Generated ethash verification cache      epoch=1 elapsed=1m9.543s
```

# Mist에 스마트 컨트랙트 배포

- miner.stop(); 로 채굴 정지



# Mist에 스마트 컨트랙트 배포



# Mist에 스마트 컨트랙트 배포

솔리더티 컨트랙트 소스 코드 | 컨트랙트 바이트 코드

```
1  pragma solidity ^0.4.18;
2
3  contract SimpleStorage {
4      uint storedData;
5
6      function set(uint x) public {
7          storedData = x;
8      }
9
10     function get() constant public returns (uint) {
11         return storedData;
12     }
13 }
```

설치할 컨트랙트를 선택하세요

Simple Storage

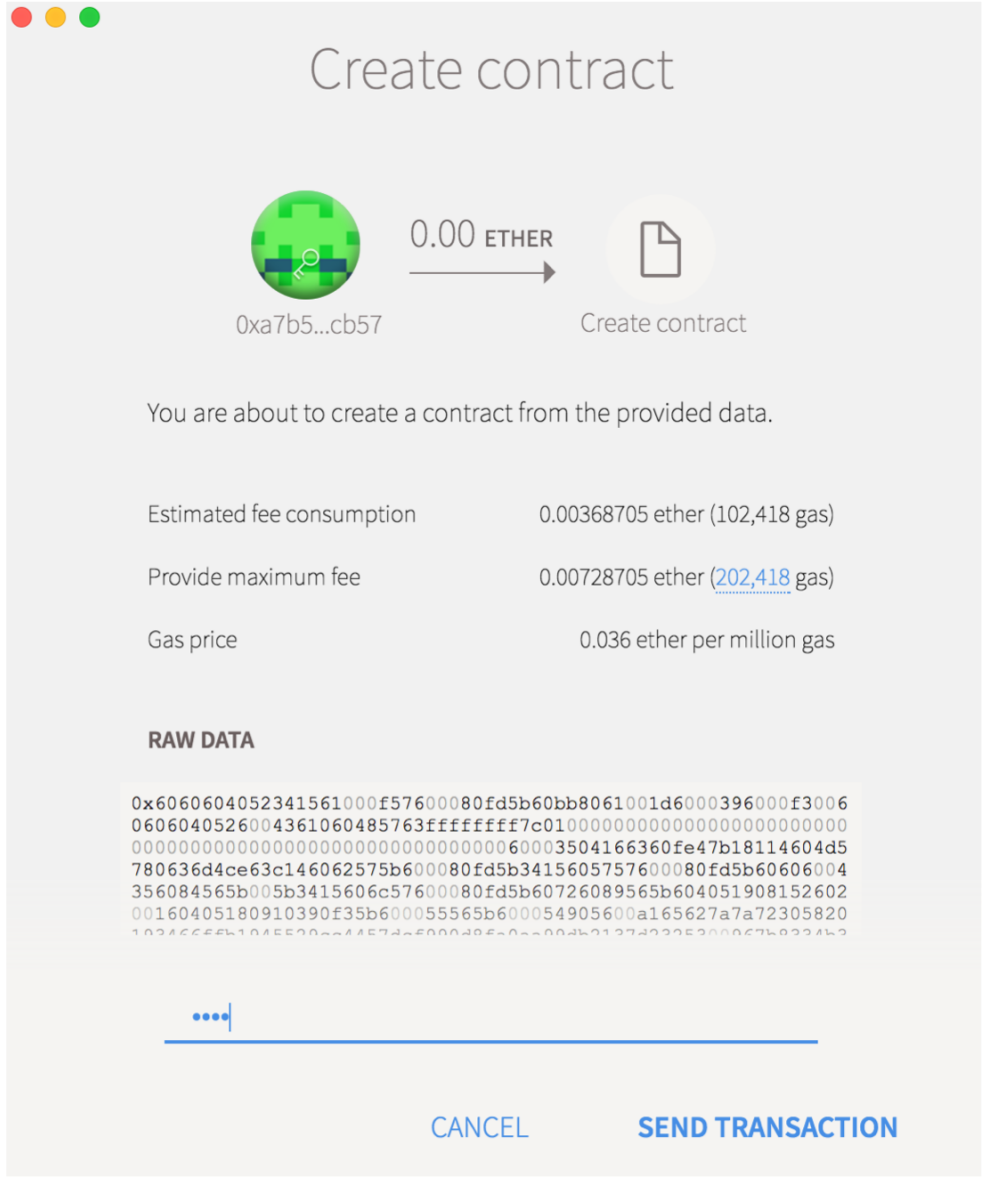
수수료 선택

0,003687048 ETHER

낮음(느린 전송) 높음(빠른 전송)

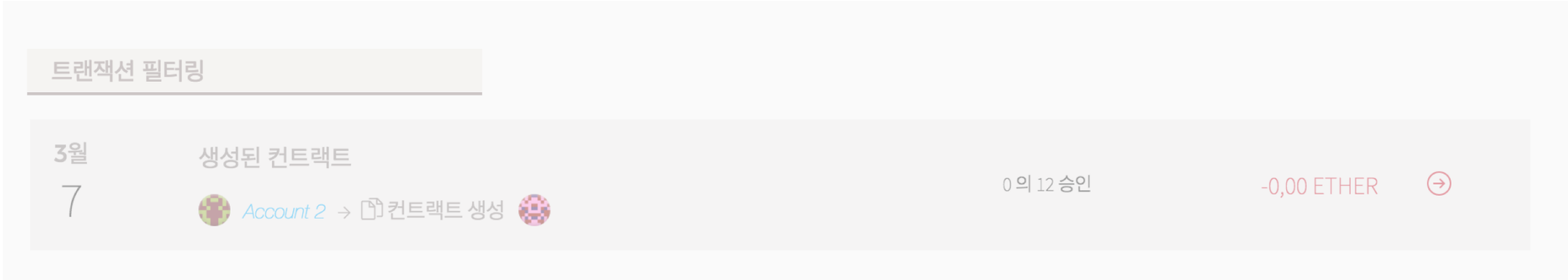
이것은 이 트랜잭션을 처리하기 위해 사용될 최대 금액입니다. 당신의 트랜잭션은 **대략 30초 이내에**에 블럭체인에 등록될 것입니다.

# Mist에 스마트 컨트랙트 배포




- 계정 초기에 설정한 비밀번호
  - 초기에 비밀번호를 설정하지 않은 경우 새 계정을 만든 후 채굴 과정부터 다시 시작

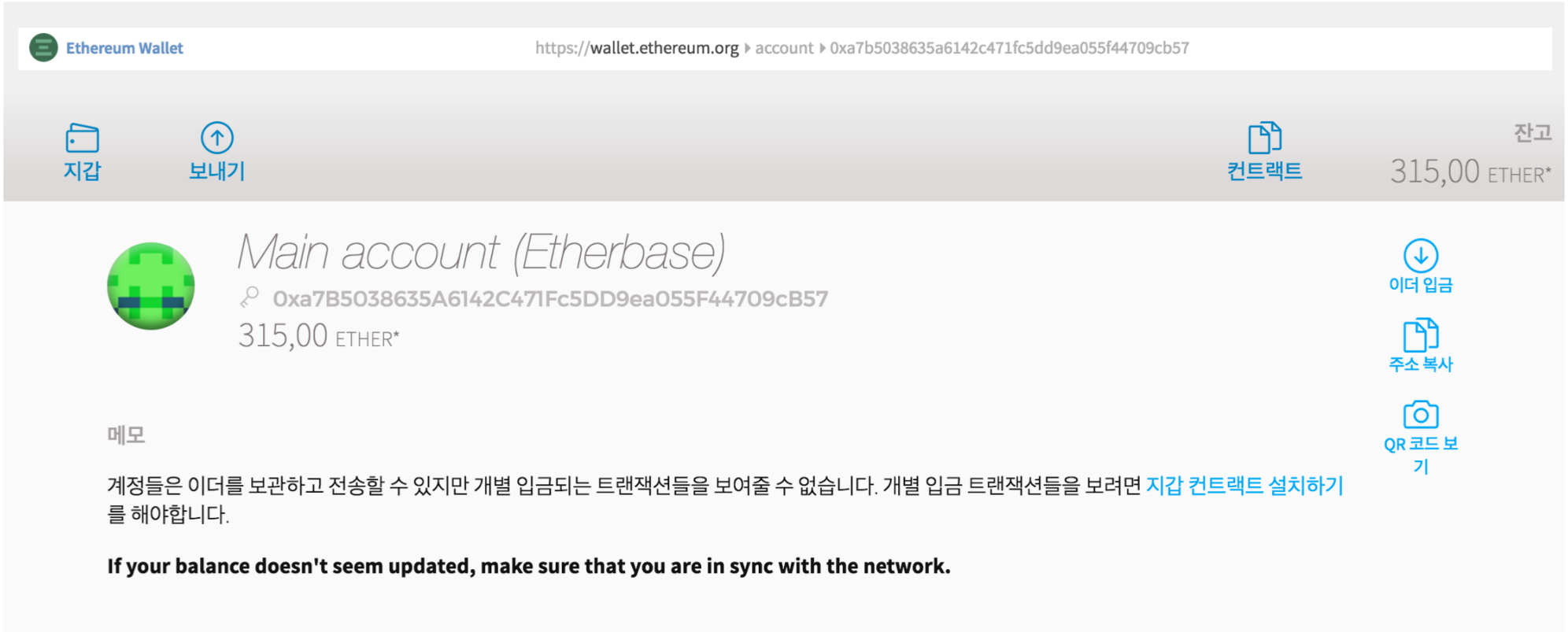
# Mist에 스마트 컨트랙트 배포



# Mist에 스마트 컨트랙트 배포

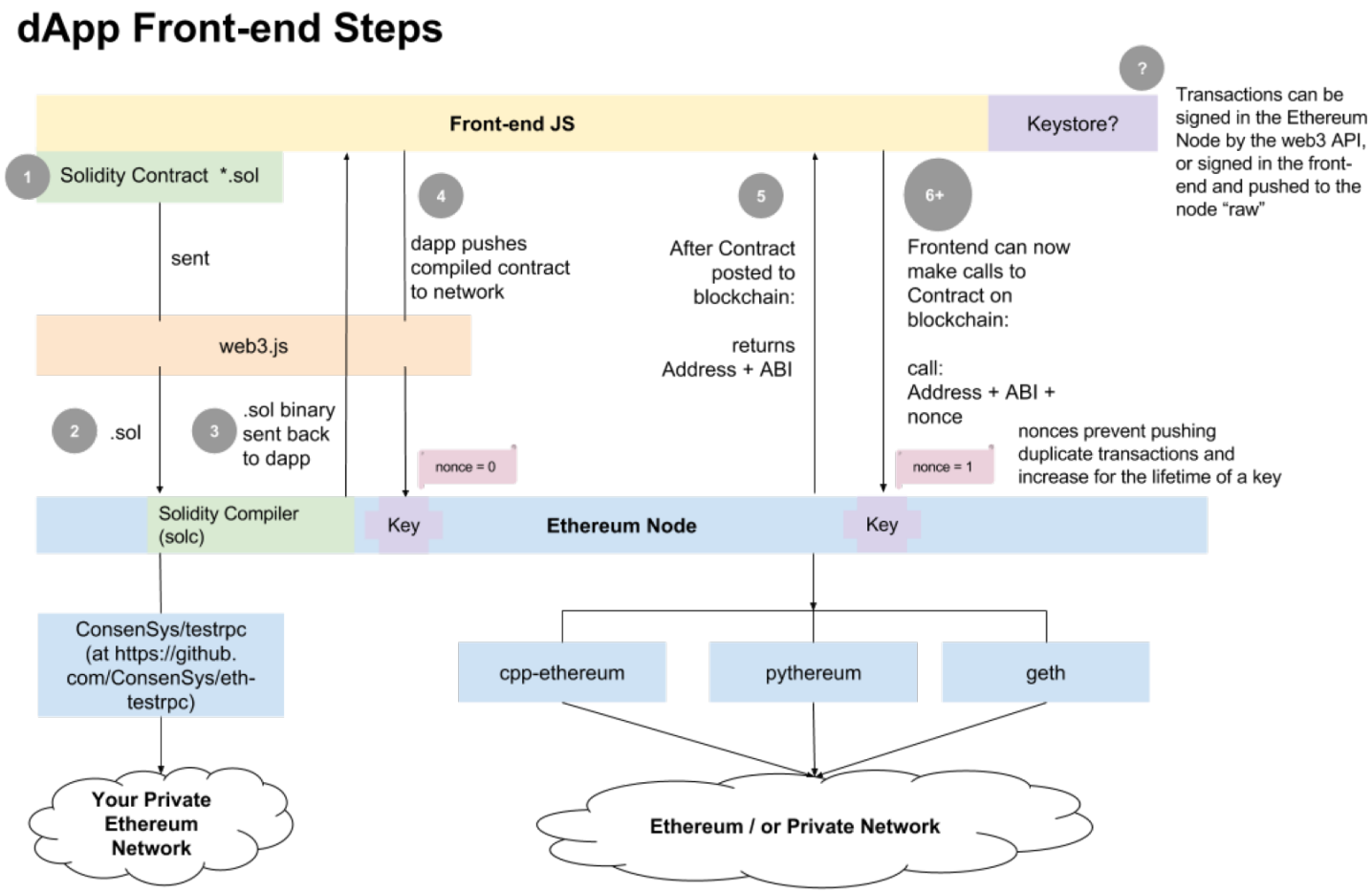
# Mist에 스마트 컨트랙트 배포

- miner.stop(); 로 채굴 정지



# Solidity

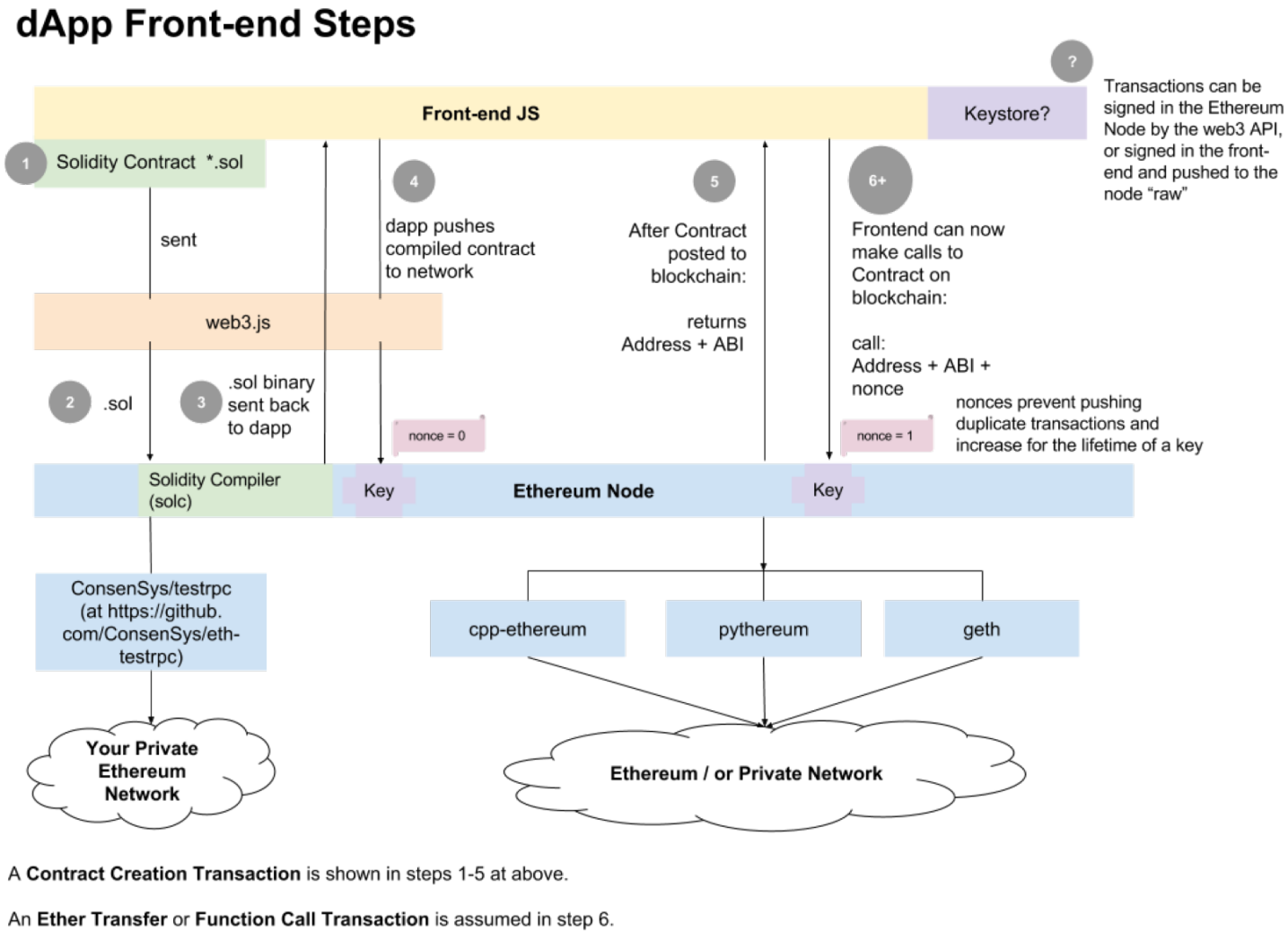
# Solidity 구조




https://cdn-images-1.medium.com/max/1600/0*B_K9HNktRm_DrEZT.png

# Solidity 구조

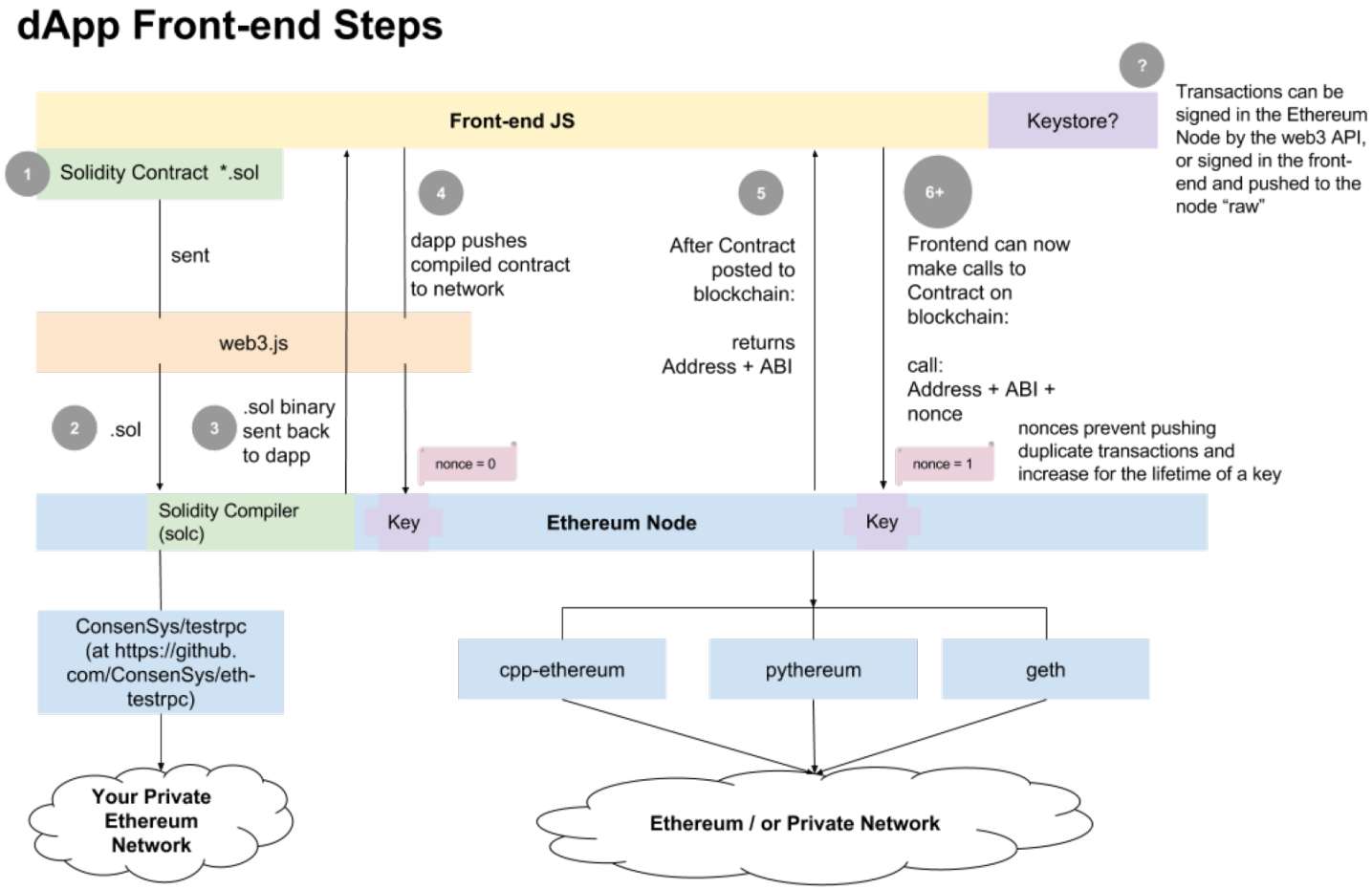
1. 스마트 컨트랙트 작성




https://cdn-images-1.medium.com/max/1600/0*B_K9HNktRm_DrEZT.png

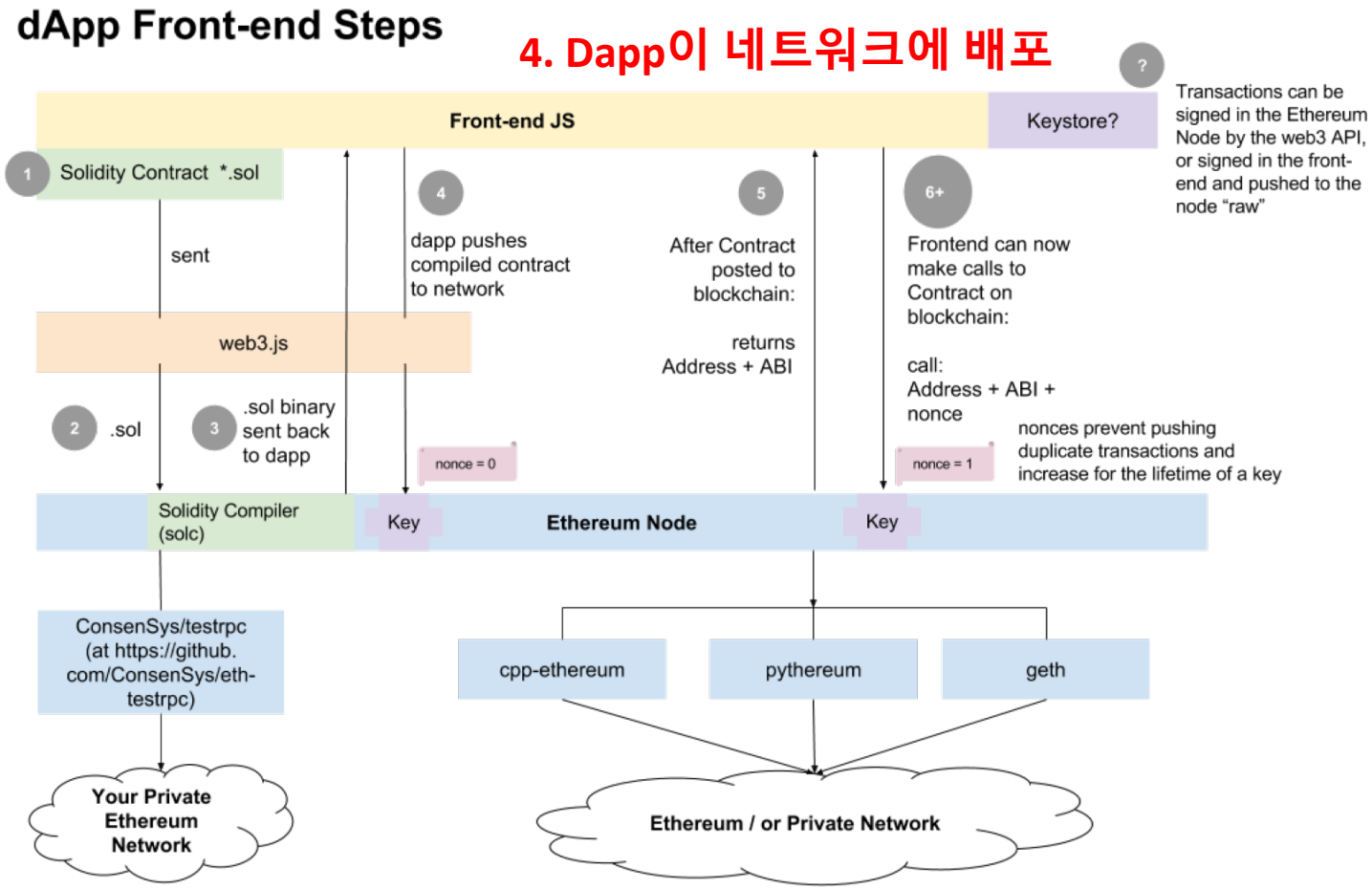
# Solidity 구조

## 2 & 3. 컴파일



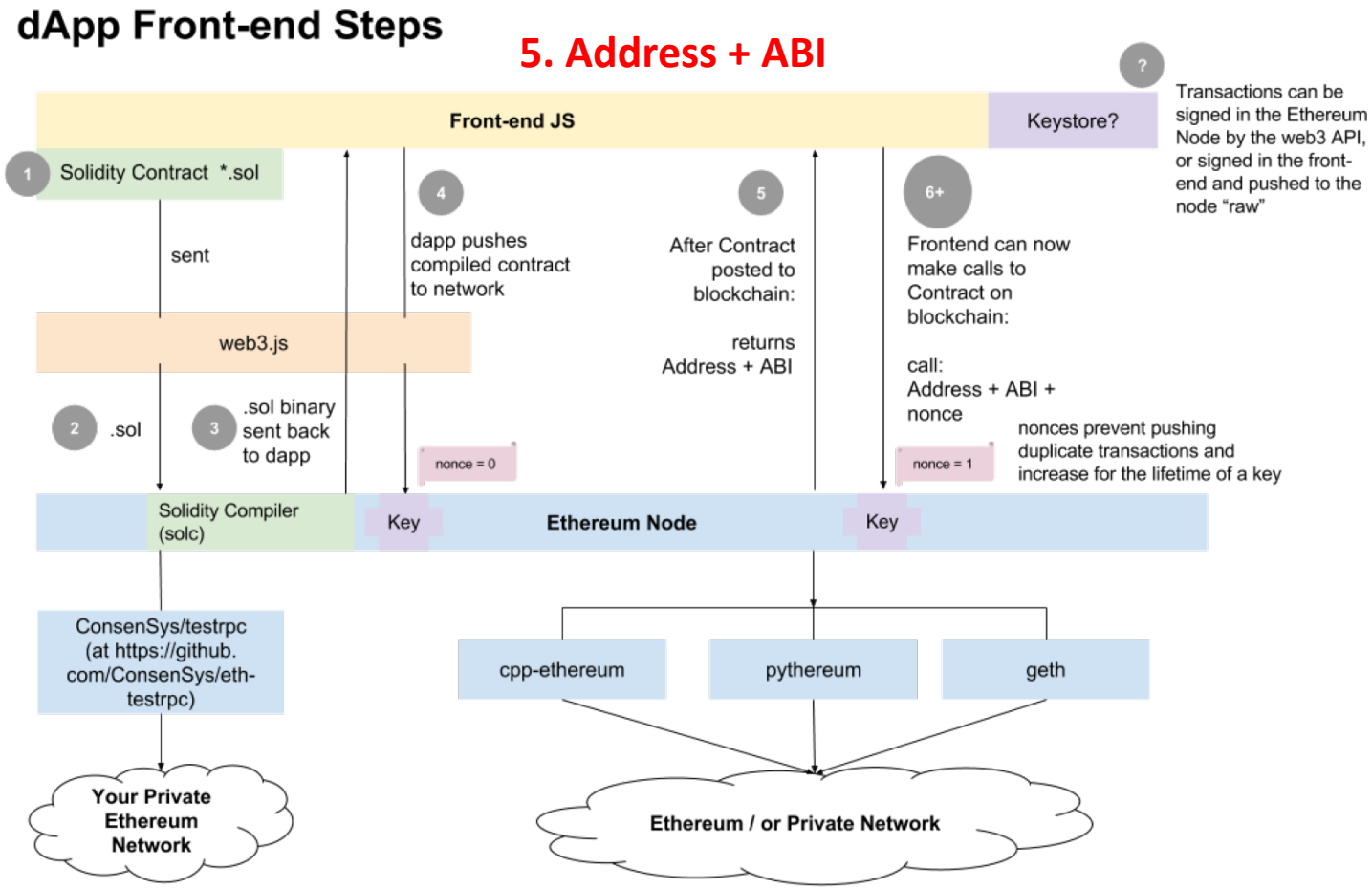

https://cdn-images-1.medium.com/max/1600/0*B_K9HNktRm_DrEZT.png

# Solidity 구조



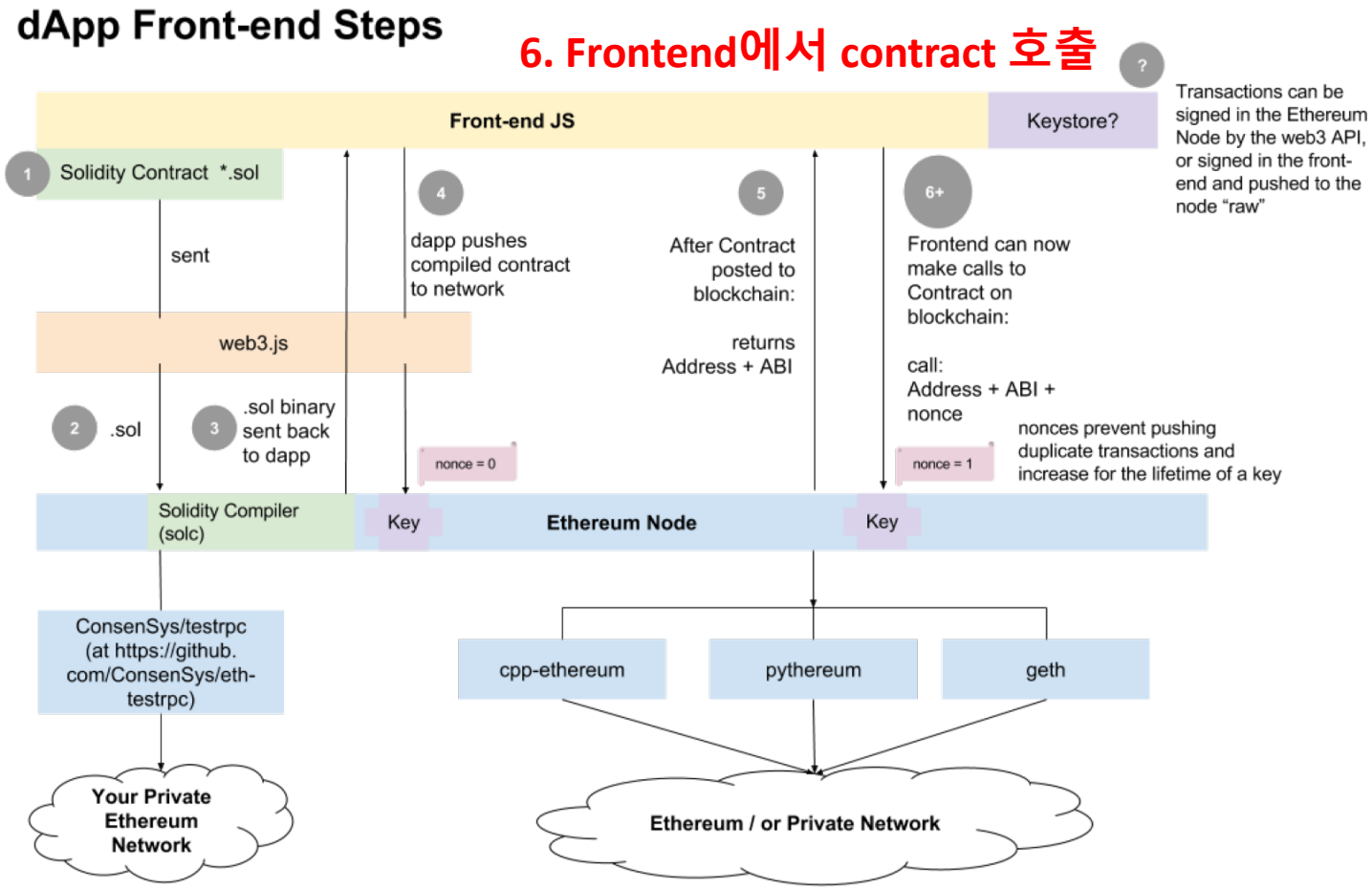

https://cdn-images-1.medium.com/max/1600/0*B_K9HNktRm_DrEZT.png

# Solidity 구조




https://cdn-images-1.medium.com/max/1600/0*B_K9HNktRm_DrEZT.png

# Solidity 구조




https://cdn-images-1.medium.com/max/1600/0*B_K9HNktRm_DrEZT.png

# Solidity 구문

- 자료형
  - uint – unsigned int
  - address – Ethereum address
  - bool, int(1~32)
  - String
  - No float in Solidity
  - Solidity는 소수(float)를 지원하지 않음
    - 소수는 부정확 하기 때문

# Solidity 구문

- State variables

```
contract SimpleStorage {
  uint storedData; // State variable // ...
}
```

- Function

```
pragma solidity ^0.4.0;
contract SimpleAuction {
  function bid() public payable { // Function // ...

  }
}
```

# Solidity 구문

- Function modifier : 구문을 명확히 하는 데 사용

```
pragma solidity ^0.4.11;
contract Purchase {
  address public seller;
  modifier onlySeller() { //Modifier
    require(msg.sender == seller);
    _;
  }
 function abort() public onlySeller { // Modifier usage // ... }
}
```

# Solidity 구문

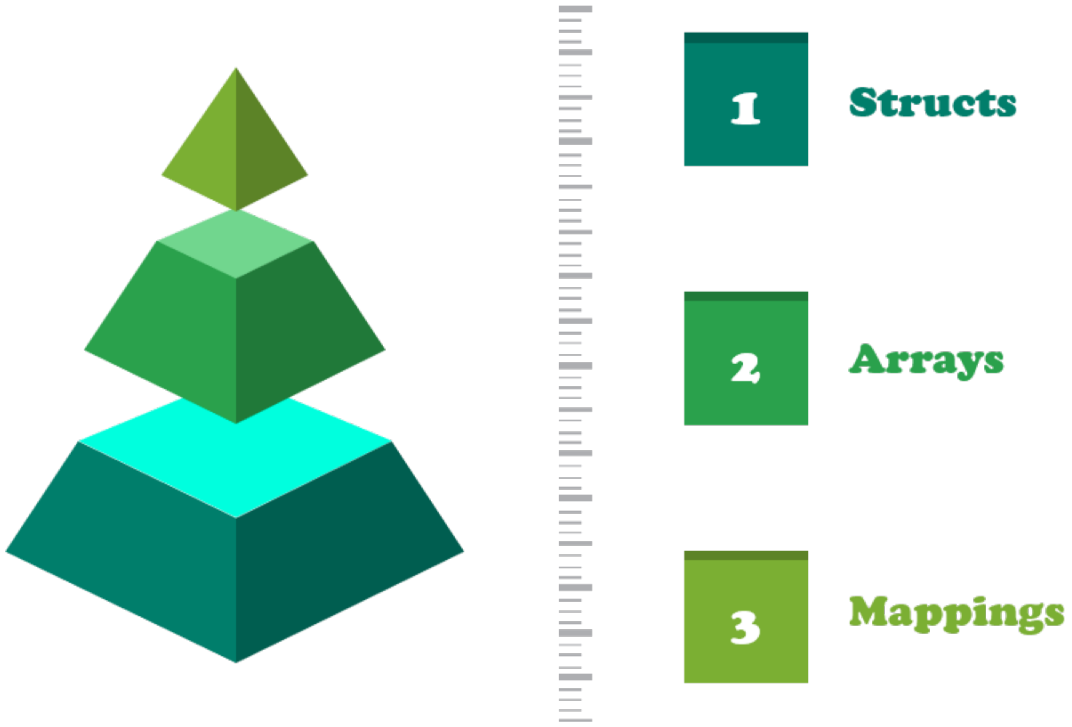
- Events : EVM 로깅 기능과 연결되는 구문

```
pragma solidity ^0.4.0;
contract SimpleAuction {
  event HighestBidIncreased(address bidder, uint amount); // Event
  function bid() public payable { // ...
     HighestBidIncreased(msg.sender, msg.value); // Triggering event
  }
}
```

# Solidity 구문

- https://github.com/shwarzes89/solidity101

# Solidity 자료형




- Mapping : 해쉬테이블
- Array : 컴파일 시 정적으로 크기가 결정되거나 동적 사이즈 설정 가능
- Struct : 다양한 타입을 속성으로 가지는 데이터 타입
  - 16개의 멤버 제한

# 예제

# ERC20 Specification

- Token Methods:
  - totalSupply
  - balanceOf
  - transfer
  - transferFrom
  - approve
  - allowance

- https://theethereum.wiki/w/index.php/ERC20_Token_Standard

# Web3.js를 이용한 스마트 컨트랙트와 UI 연동

- Node.js 설치 (https://nodejs.org/ko)

# Web3.js를 이용한 스마트 컨트랙트와 UI 연동

- 정상적으로 설치되었는지 확인
  - node –v
  - npm –v

- Ganache cli 설치
  - npm install ganache-cli

# Web3.js를 이용한 스마트 컨트랙트와 UI 연동

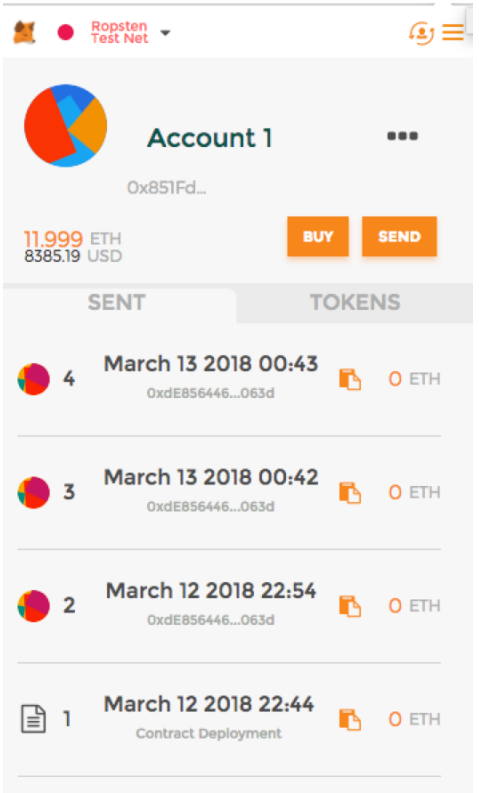
- 정상적으로 설치되었는지 확인
  - node –v
  - npm –v
- Web3.js 설치
  - npm install ethereum/web3.js --save
- Ganache cli 설치
  - npm install ganache-cli

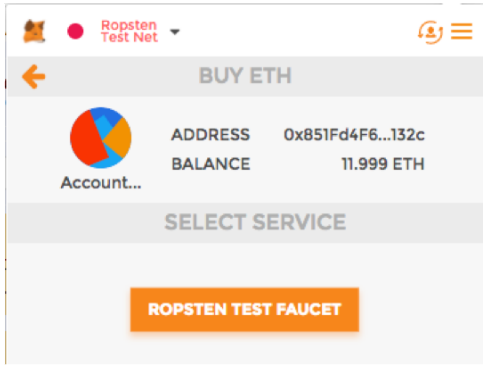
# Web3.js를 이용한 스마트 컨트랙트와 UI 연동

- 샘플 스마트 컨트랙트 작성
  - https://gist.github.com/shwarzes89/0a275daefb2a190ce6a78a87db9563a4
  - TODO 부분 실습

# Web3.js를 이용한 스마트 컨트랙트와 UI 연동

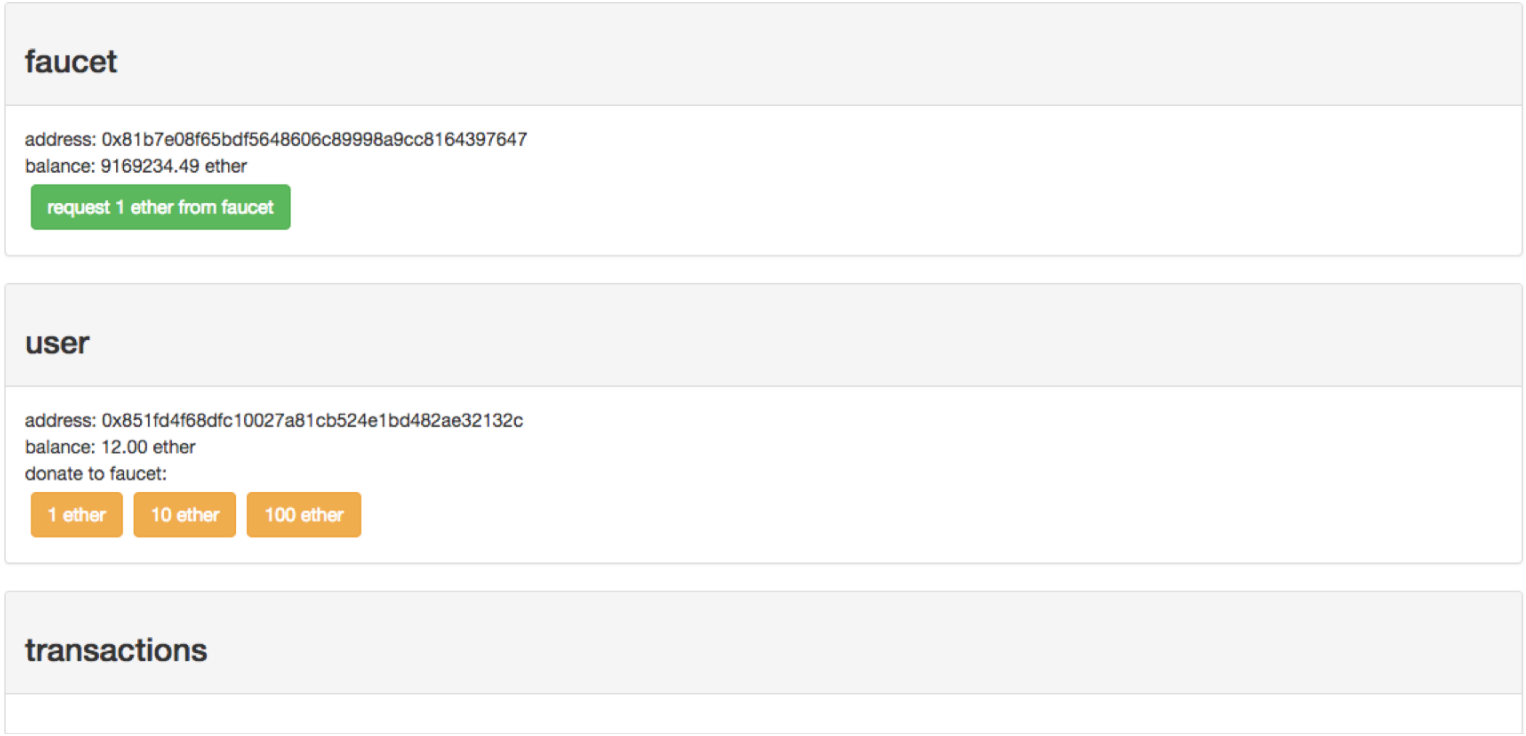
1. MetaMask 실행
2. Ropsten network 연결
3. ‘Buy’ 클릭
4. ‘Ropsten test faucet’ 클릭



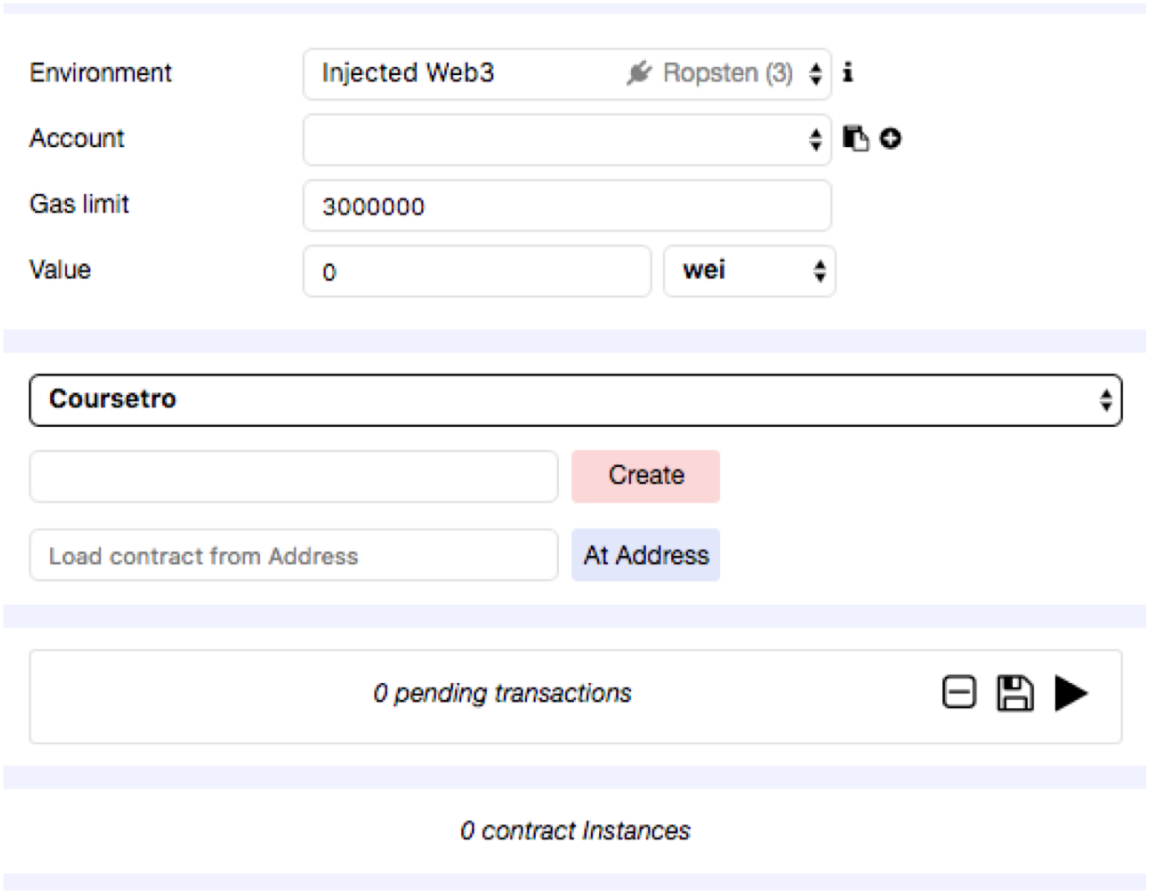
# Web3.js를 이용한 스마트 컨트랙트와 UI 연동

5. ‘Request 1 ether from faucet’ - Ether 요청



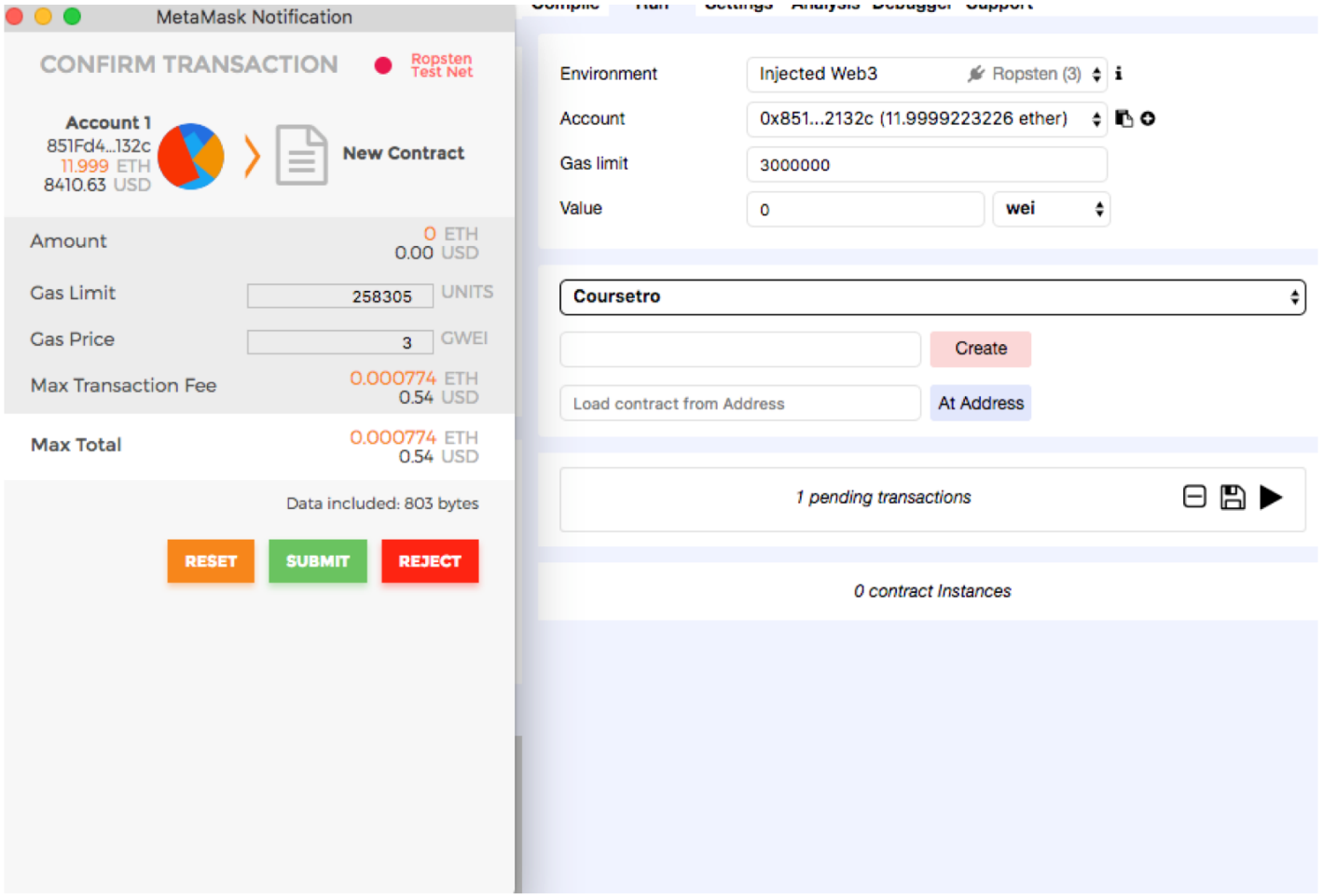
# Web3.js를 이용한 스마트 컨트랙트와 UI 연동

- 스마트 컨트랙트 생성



# Web3.js를 이용한 스마트 컨트랙트와 UI 연동

- 스마트 컨트랙트 생성



# Web3.js를 이용한 스마트 컨트랙트와 UI 연동

- Remix의 Compile 탭 – ‘Details’
- ABI 복사

# Web3.js를 이용한 스마트 컨트랙트와 UI 연동

- Web page 작성
  - https://gist.github.com/shwarzes89/469bfbf53356c03da04c24ccfb764b97
  - TODO 부분 실습

# ERC20 Specification

- https://gist.github.com/shwarzes89/ea754daef5dd805aaa11bdd98a308c8b

# Greeter

https://gist.github.com/shwarzes89/fe25c9756b8506906906d45b6c4ecbe9

# Coin example

- 이더리움으로 살 수 있는 코인
- 1eth = 20 MyCoin
- 구매
- 환불

# My Coin

https://gist.github.com/shwarzes89/e8bcaffc8bd0d1f43535a99fe0aab94a

# My Coin

mapping(address=>uint) balances;

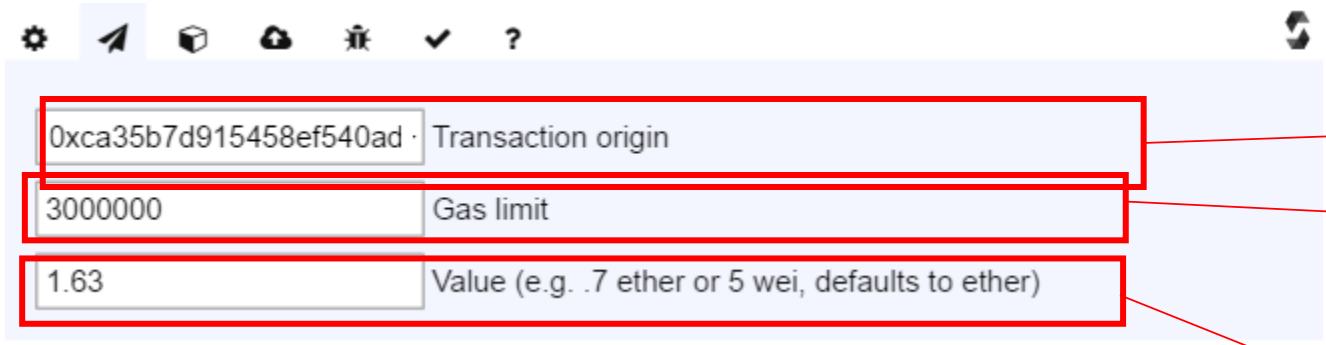
- 잔액 저장

uint COIN_WEI = 50000000000000000;

- 1Eth는 너무 크기 때문에 Ehtereum에선 WEI로 바꿔서 사용한다.
- 1 eth = 1000000000000000000 wei
- 1 eth = 20 coin
- 1 coin = 50000000000000000 wei

# My Coin

트랜잭션 세팅하기




보내는 사람 주소
드래그 가능

실행 Gas 제한,
limit를 넘어가는 함수를 실행 불가

보내는 돈의양
단위 Ether

# My Coin


컨트렉트 생성
같은 소스코드로 여러
컨트렉트를 생성 가능


아이콘을 누르면 함수 실행,
파라미터는 옆에 기제

# 안전 거래 서비스

- 스마트 컨트렉트는 돈을 보관한다.
- 물건이 확인되면 돈을 보낸다.

# 안전거래 서비스

https://gist.github.com/shwarzes89/0526d0302e044496c618ebcbd9b493ec

- value = 물건 가격
- seller = 판매자
- Buyer = 구매자
- State = 생성, 잠금,물건 배송, 완료

# 경매

https://gist.github.com/shwarzes89/0c089125246ea814b19e98b93d0c4e96

# 참조

- Solidity 문서
  - https://solidity.readthedocs.io/en/v0.4.20

- Solidity 샘플
  - https://github.com/fivedogit/solidity-baby-steps
  - https://github.com/bellaj/smart-contract