

스마트 컨트랙트 개발을 위한 이더리움 개발환경 조사

Index

- 이더리움 네트워크 준비하기
 - Geth, Ganache
- 이더리움 컴파일러 설치
 - Solc
- 스마트 컨트랙트 개발
 - Remix, Truffle
- Web기반 Dapp에서 스마트 컨트랙트 사용
 - Web3.js, Metamask

이더리움 네트워크 준비하기

Geth & Ganache

Geth(Go Ethereum)

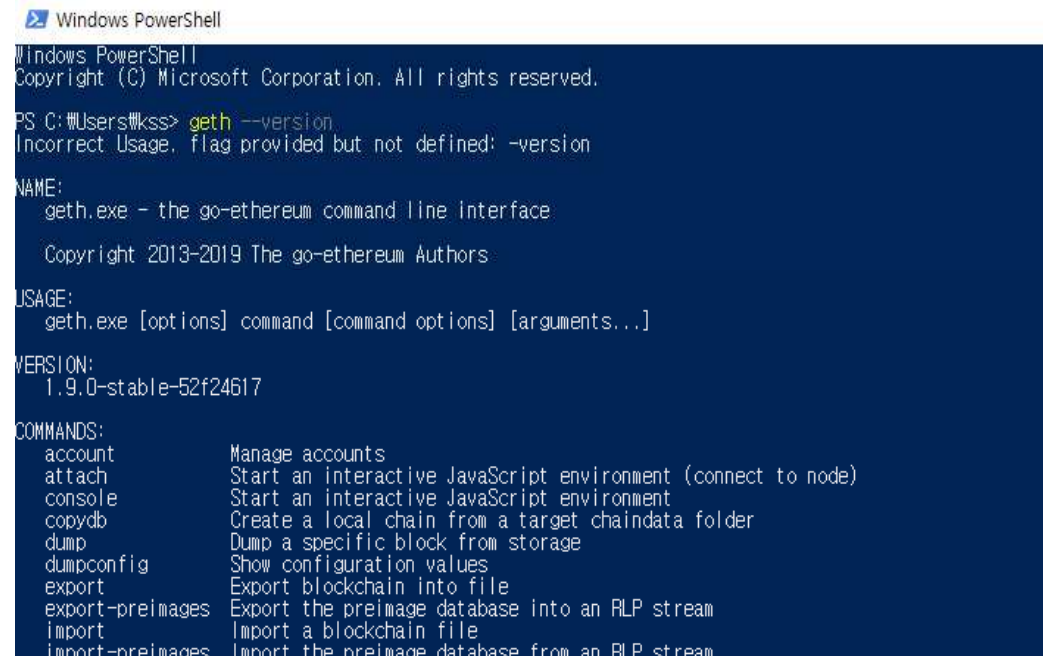
- Ethereum 프로토콜을 구현한, Go언어로 작성된 Ethereum client program
- 설치방법
 - <https://github.com/ethereum/go-ethereum/wiki/Building-Ethereum>
- Geth 설치파일
 - <https://geth.ethereum.org/downloads/>



1. Download zip file
2. Extract geth.exe from zip
3. Open a command prompt
4. chdir
5. open geth.exe

Geth(Go Ethereum) Cont.

- 메인 네트워크 또는 테스트 네트워크에 연결 가능
- EVM을 사용해 Private 네트워크를 위한 채굴 및 EVM 노드 생성 가능
- JSON RPC 프로토콜에 기초
- Command Line Options
 - https://github.com/ethereum/go-ethereum/wiki/Command-Line-Options?source=post_page-----



```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

PS C:\Users\mkss> geth --version
Incorrect Usage. flag provided but not defined: -version

NAME:
  geth.exe - the go-ethereum command line interface

  Copyright 2013-2019 The go-ethereum Authors

USAGE:
  geth.exe [options] command [command options] [arguments...]

VERSION:
  1.9.0-stable-52f24617

COMMANDS:
  account      Manage accounts
  attach       Start an interactive JavaScript environment (connect to node)
  console      Start an interactive JavaScript environment
  copydb       Create a local chain from a target chaindata folder
  dump         Dump a specific block from storage
  dumpconfig   Show configuration values
  export       Export blockchain into file
  export-preimages Export the preimage database into an RLP stream
  import       Import a blockchain file
  import-preimages Import the preimage database from an RLP stream
```

Geth(Go Ethereum) Cont.

- Private 이더리움 네트워크 구성 과정



Cf. Genesis File

- Private network의 **환경설정**
- 채굴 난이도, 체인 환경설정 부분이 들어 있음
- Params
 - chainId : 현재 chain을 식별하는 값. Reply attack을 막기 위해 사용
 - homesteadBlock : ethereum release 버전
 - Difficulty : 채굴 난이도. 값을 낮게 줄수록 block 하나를 mining하는데 걸리는 시간이 짧음
 - gasLimit : block 하나에 포함할 수 있는 gasLimit양
 - Alloc : pre-funded address

```
Windows PowerShell
PS C:\Users\kss\privateNet> dir

디렉터리: C:\Users\kss\privateNet

Mode                LastWriteTime         Length Name
----                -
-a-----      2019-07-25 오전 10:20             491 genesis.json

PS C:\Users\kss\privateNet> geth init .\genesis.json --datadir .\chaindata
INFO [07-25|10:25:53.813] Bumping default cache on mainnet
INFO [07-25|10:25:53.887] Maximum peer count
INFO [07-25|10:25:54.201] Allocated cache and file handles
INFO [07-25|10:25:54.239] Writing custom genesis block
INFO [07-25|10:25:54.241] Persisted trie from memory database
INFO [07-25|10:25:54.250] Successfully wrote genesis state
INFO [07-25|10:25:54.253] Allocated cache and file handles
INFO [07-25|10:25:54.289] Writing custom genesis block
INFO [07-25|10:25:54.291] Persisted trie from memory database
INFO [07-25|10:25:54.299] Successfully wrote genesis state

PS C:\Users\kss\privateNet> geth --datadir .\chaindata --rpc --rpcapi "eth,web3,miner,admin,personal,net" --rpcorsdomain "*" --nodiscover --networkid 15
INFO [07-25|10:29:22.616] Maximum peer count
INFO [07-25|10:29:22.903] Starting peer-to-peer node
INFO [07-25|10:29:22.907] Allocated trie memory caches
INFO [07-25|10:29:22.910] Allocated cache and file handles
INFO [07-25|10:29:23.025] Opened ancient database
INFO [07-25|10:29:23.030] Initialized chain configuration
Byzantium: <nil> Constantinople: <nil> Petersburg: <nil> Engine: unknown
INFO [07-25|10:29:23.037] Disk storage enabled for ethash caches
INFO [07-25|10:29:23.041] Disk storage enabled for ethash DAGs
INFO [07-25|10:29:23.045] Initialising Ethereum protocol
WARN [07-25|10:29:23.049] Upgrade blockchain database version
INFO [07-25|10:29:23.104] Loaded most recent local header
INFO [07-25|10:29:23.108] Loaded most recent local full block
INFO [07-25|10:29:23.112] Loaded most recent local fast block
INFO [07-25|10:29:23.121] Regenerated local transaction journal
INFO [07-25|10:29:23.129] Allocated fast sync bloom
INFO [07-25|10:29:23.132] Initialized fast sync bloom
INFO [07-25|10:29:23.275] New local node record
INFO [07-25|10:29:23.276] IPC endpoint opened
INFO [07-25|10:29:23.279] Started P2P networking
948d26b043b4482de08c8d2099fd46f54fa570862a889980127.0.0.1:30303?disco=0"
INFO [07-25|10:29:23.282] HTTP endpoint opened
WARN [07-25|10:35:43.932] Served eth_coinbase
INFO [07-25|10:36:31.422] Your new key was generated
WARN [07-25|10:36:31.425] Please backup your key file!
21842618be8c993849574a6bcd43ec195e8
WARN [07-25|10:36:31.430] Please remember your password!
INFO [07-25|10:36:28.866] Updated mining threads
INFO [07-25|10:36:28.869] Transaction pool price threshold updated
INFO [07-25|10:36:28.871] Commit new mining work
INFO [07-25|10:36:31.068] Successfully sealed new block
INFO [07-25|10:36:31.072] □□□mined potential block
INFO [07-25|10:36:31.080] Commit new mining work
INFO [07-25|10:36:31.090] Successfully sealed new block
INFO [07-25|10:36:31.094] □□□mined potential block
INFO [07-25|10:36:31.100] Commit new mining work
INFO [07-25|10:36:31.175] Successfully sealed new block
INFO [07-25|10:36:31.178] □□□mined potential block
INFO [07-25|10:36:31.178] Mining too far in the future
INFO [07-25|10:36:33.185] Commit new mining work
INFO [07-25|10:36:33.612] Successfully sealed new block
INFO [07-25|10:36:33.616] □□□mined potential block
INFO [07-25|10:36:33.625] Commit new mining work

Instance: Geth/v1.9.0-stable-52f24617/windows-amd64/go1.12.7
at block: 0 (Thu, 01 Jan 1970 09:00:00 KST)
datadir: C:\Users\kss\privateNet\chaindata
modules: admin:1.0 eth:1.0 miner:1.0 net:1.0 personal:1.0 rpc:1.0 web3:1.0

> personal.newAccount()
Passphrase:
Repeat passphrase:
"21842618be8c993849574a6bcd43ec195e8"
> miner.setEtherbase("0xb235621842618be8c993849574a6bcd43ec195e8")
true
> eth.coinbase
"0xb235621842618be8c993849574a6bcd43ec195e8"
> miner.start()
not
> eth.getBalance(eth.accounts[0])
> eth.getBalance(eth.accounts[0])
> eth.blockNumber
> miner.stop()
null
>
```

채굴

```
Windows PowerShell
PS C:\Users\kss> geth attach rpc:http://localhost:8545
INFO [07-25|10:35:43.428] Bumping default cache on mainnet
Welcome to the Geth JavaScript console!

Instance: Geth/v1.9.0-stable-52f24617/windows-amd64/go1.12.7
at block: 0 (Thu, 01 Jan 1970 09:00:00 KST)
datadir: C:\Users\kss\privateNet\chaindata
modules: admin:1.0 eth:1.0 miner:1.0 net:1.0 personal:1.0 rpc:1.0 web3:1.0

> personal.newAccount()
Passphrase:
Repeat passphrase:
"21842618be8c993849574a6bcd43ec195e8"
> miner.setEtherbase("0xb235621842618be8c993849574a6bcd43ec195e8")
true
> eth.coinbase
"0xb235621842618be8c993849574a6bcd43ec195e8"
> miner.start()
not
> eth.getBalance(eth.accounts[0])
> eth.getBalance(eth.accounts[0])
> eth.blockNumber
> miner.stop()
null
>
```

Ganache(가나슈)

- 가상의 이더리움 네트워크를 생성해서
스마트 컨트랙트를 실행할 수 있도록 해주는 프로그램(TestRPC)
- Ganache 설치파일
 - <https://www.trufflesuite.com/ganache>



Ganache(가나슈) Cont.

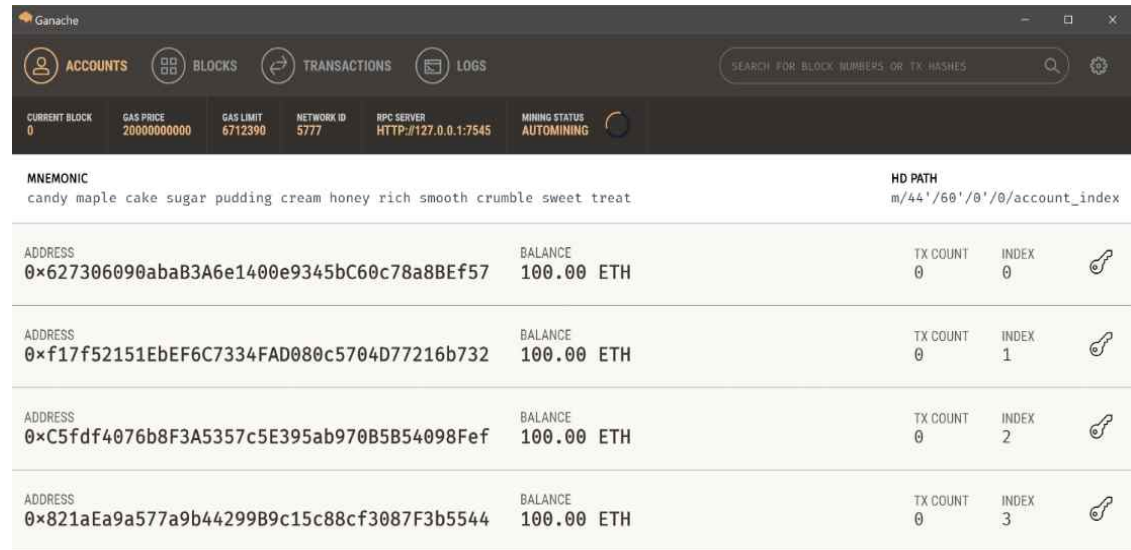
- 이더리움 솔루션 및 스마트 계약의 개발과 테스트 과정을 간소화하기 위해 개발됨

- 자체적으로 이더리움 거래 처리와 채굴 기능을 가짐

> > 거래 채굴 대기 시간이 없음.

거래는 생성되는 대로 기록됨

- 개발자가 가나슈 CLI를 이더리움 노드로 사용 가능



솔리디티 컴파일러

Solc

Solc (Solidity Compiler)

- 작성한 Smart Contract를 EVM에서 실행 가능하도록 Low level Instruction인 "ByteCode"로 바꿔주는 Compiler
- Compile 결과, 작성한 Smart Contract에 대한 ABI, ByteCode가 생성
 - ByteCode
 - Smart Contract Code를 컴파일한 결과
 - EVM 위에서 실행됨
 - ABI(Application Binary Interface)
 - Smart Contract의 외부/공개 함수와 Parameter에 대한 Metadata 정의
 - Contract의 함수를 호출하는데 사용
- Solc 설치파일
 - <https://github.com/ethereum/solidity/releases>

Solc (Solidity Compiler) Cont.

```
pragma solidity ^0.4.16;

contract testContract {
    uint count;

    constructor() public {
        count=0;
    }

    /// @dev Payable function to modify count with _n
    function setP(uint _n) payable public {
        count = _n;
    }

    /// @dev Function to modify count with _n
    function setNP(uint _n) public {
        count = _n;
    }

    /// @dev Function to get count value
    function get () public view returns (uint) {
        return count;
    }

    /// @dev Function to return cumulative sum between _input and count
    function loopCounter(uint _input) public view returns (uint) {
        uint returnValue;

        for(; _input<count; _input++) {
            returnValue += _input;
        }
        return returnValue;
    }
}
```

Contract Code

```
[
{
    "constant": false,
    "inputs": [
        {
            "name": "_n",
            "type": "uint256"
        }
    ],
    "name": "setNP",
    "outputs": [],
    "payable": false,
    "stateMutability": "nonpayable",
    "type": "function"
},
{
    "constant": true,
    "inputs": [
        {
            "name": "_input",
            "type": "uint256"
        }
    ],
    "name": "loopCounter",
    "outputs": [
        {
            "name": "",
            "type": "uint256"
        }
    ],
    "payable": false,
    "stateMutability": "view",
    "type": "function"
}
...]
```

ABI

ByteCode

Compile example

```
Windows PowerShell

Mode                LastWriteTime         Length Name
-----
d----- 2019-07-25 오전 10:25                chaindata
-a----- 2019-07-25 오전 11:23             1668 contractResult.js
-a----- 2019-07-25 오전 10:20             491 genesis.json
-a----- 2019-07-25 오전 11:11             795 testContract.sol

PS C:\Users\kss\privateNet> solc --abi --bin ./testContract.sol

===== ./testContract.sol:testContract =====
Binary:
608060405234801561001057600080fd5b50600080819055506101a2806100276000396000f3fe60806040526004361061003f5760003560e01c8063159fbd214
610044578063189d61c51461007f5780636d4ce63c146100ce578063c95473db146100f9575b600080fd5b34801561005057600080fd5b5061007d600480360360
2081101561006757600080fd5b8101908080359060200190929190505050610127565b005b34801561008b57600080fd5b506100b8600480360360208110156100
a257600080fd5b8101908080359060200190929190505050610131565b6040518082815260200191505060405180910390f35b3480156100da57600080fd5b5061
00e361015a565b6040518082815260200191505060405180910390f35b6101256004803603602081101561010f57600080fd5b8101908080359060200190929190
505050610163565b005b8060008190555050565b6000805b6000548310156101515782810190508280600101935050610135565b80915050919050565b60008054
905090565b806000819055505056fea265627a7a72305820ed3e4f67338d4c2e1a8cef137870bf703da23637a8e43dd3add46814cf4978c264736f6c634300050a
0032
Contract JSON ABI
[{"constant":false,"inputs":[{"name":"_n","type":"uint256"}],"name":"setNP","outputs":[],"payable":false,"stateMutability":"nonpay
able","type":"function"},{"constant":true,"inputs":[{"name":"_input","type":"uint256"}],"name":"loopCounter","outputs":[{"name":"","
","type":"uint256"}],"payable":false,"stateMutability":"view","type":"function"},{"constant":true,"inputs":[],"name":"get","outputs
":[{"name":"","","type":"uint256"}],"payable":false,"stateMutability":"view","type":"function"},{"constant":false,"inputs":[{"name":"
_n","type":"uint256"}],"name":"setP","outputs":[],"payable":true,"stateMutability":"payable","type":"function"},{"inputs":[],"paya
ble":false,"stateMutability":"nonpayable","type":"constructor"}]
```

컴파일
산출물

```
Windows PowerShell
PS C:\Users\kss\privateNet> geth attach rpc:http://localhost:8545
INFO [07-25]13:18:47.308] Bumping default cache on mainnet provided=1024 updated=4096
Welcome to the Geth JavaScript console!

instance: Geth/v1.9.0-stable-52f24617/windows-amd64/go1.12.7
coinbase: 0xb235621842618be8c993849574ab6cd43ec195e8
at block: 765 (Thu, 25 Jul 2019 11:43:06 KST)
datadir: C:\Users\kss\privateNet\chaindata
modules: admin:1.0 eth:1.0 miner:1.0 net:1.0 personal:1.0 rpc:1.0 web3:1.0

> loadScript('contractResult.js')
true
> var testContract = eth.contract(contractAbi)
undefined
> personal.unlockAccount(eth.accounts[0]);
Unlock account 0xb235621842618be8c993849574ab6cd43ec195e8
Passphrase:
true
> var contractObj = testContract.new({from:eth.accounts[0], data:contractBin, gas:2000000});
undefined
> contractObj.get()
TypeError: 'get' is not a function
    at <anonymous>:1:1
> miner.start()
null
> contractObj.get()
> miner.stop()
null
>
```

실행중인 게스 노드에 연결

ABI, Bytecode를 담은 js 파일 로드

Contract Object 생성을 위한 배포 준비

Contract 배포 후 Contract Object 생성

채굴 시작 시킨 후, 컨트랙트 함수 호출

```
contractResult.js
1 contractAbi =
  * [{"constant":false,"inputs":[{"name":"_n","type":"uint256"}],"name":"setNP","outputs":[],"payable":false,"stateMutability":"nonpayable","type":"function"}, {"constant":true,"inputs":[{"name":"_input","type":"uint256"}],"name":"loopCounter","outputs":[{"name":"","type":"uint256"}],"payable":false,"stateMutability":"view","type":"function"}, {"constant":true,"inputs":[{"name":"","type":"uint256"}],"name":"get","outputs":[{"name":"","type":"uint256"}],"payable":false,"stateMutability":"view","type":"function"}, {"constant":false,"inputs":[{"name":"_n","type":"uint256"}],"name":"setP","outputs":[],"payable":true,"stateMutability":"payable","type":"function"}, {"inputs":[{"name":"","type":"uint256"}],"payable":false,"stateMutability":"nonpayable","type":"constructor"}];
2
3 contractBin =
  * "0x608060405234801561001057600080fd5b50600080819055506101a2806100276000396000f3fe6080604052
  * 6004361061003f5760003560e01c8063159fbdb214610044578063189d61c51461007f5780636d4ce63c146100c
  * e578063c95473db146100f9575b600080fd5b34801561005057600080fd5b5061007d6004803603602081101561
  * 006757600080fd5b81019080803590602001909291905050610127565b005b34801561008b57600080fd5b506
  * 100b8600480360360208110156100a257600080fd5b81019080803590602001909291905050610131565b6040
  * 518082815260200191505060405180910390f35b3480156100da57600080fd5b506100e361015a565b604051808
  * 2815260200191505060405180910390f35b6101256004803603602081101561010f57600080fd5b810190808035
  * 90602001909291905050610163565b005b8060008190555050565b60008054905090565b80600081905550565b
  * 0508280600101935050610135565b80915050919050565b60008054905090565b80600081905550565bfea26562
  * 7a7a72305820ed3e4f67338d4c2e1a8cef137870bf703da23637a8e43dd3add46814cf4978c264736f6c6343000
  * 50a0032";
  * 50a0032";
```

스마트 컨트랙트 개발

Remix & Truffle

Remix

- Solidity 기반 스마트 컨트랙트 개발을 위한 브라우저기반 IDE
- Smart Contract 작성, 컴파일, 디버깅, 배포 가능
- 이더리움 메인넷, 테스트넷, 로컬 가상머신에서 연결 가능
- Remix IDE 홈페이지
 - <https://remix.ethereum.org>
- Remix IDE Doc
 - <https://remix-ide.readthedocs.io/en/latest/index.html>

ETRI-업무포털

Remix - Ethereum IDE

이더리움 Dapp만들기 5. 스마트

남에게 제공할 수 있어야 비로소

https://remix.ethereum.org/#optimize=false&evmVersion=null&version=soljson-v0.4.26+commit.4563c3fc.js

Color Scriptor

SW Expert Academy

인공지능 - 충북대...

충북대학교 - 수강...

블로그

전공 교과

인턴십

프로젝트

졸업작품

2019 ETRI 인턴

기타 생활

remix

SOLIDITY COMPILER

Compiler0.4.26+commit.4563c3fc

LanguageSolidity

EVM Versioncompiler default

Compile testContract.sol

Compiler Configuration

☒ Auto compile

☐ Enable Optimization

☐ Hide warnings

ContracttestContract (testCon

ABI

Bytecode

Compilation Details

Publish on Swarm

testContract.sol

```
1 pragma solidity ^0.4.16;
2
3 contract testContract {
4     uint count;
5
6     constructor() public {
7         count=0;
8     }
9
10    /// @dev Payable function to modify count with _n
11    function setP(uint _n) payable public {
12        count = _n;
13    }
14
15    /// @dev Function to modify count with _n
16    function setNP(uint _n) public {
17        count = _n;
18    }
19
20    /// @dev Function to get count value
21    function get () public view returns (uint) {
22        return count;
23    }
24
25
26    /// @dev Function to return cumulative sum between _input and count
27    function loopCounter(uint _input) public view returns (uint) {
28        uint returnValue;
29
30        for(; _input<count; _input++) {
31            returnValue += _input;
32        }
33        return returnValue;
34    }
35
36 }
37
38
39
```

listen on network

Search with transaction hash or address

remix.debugHelp(): Display help message for debugging

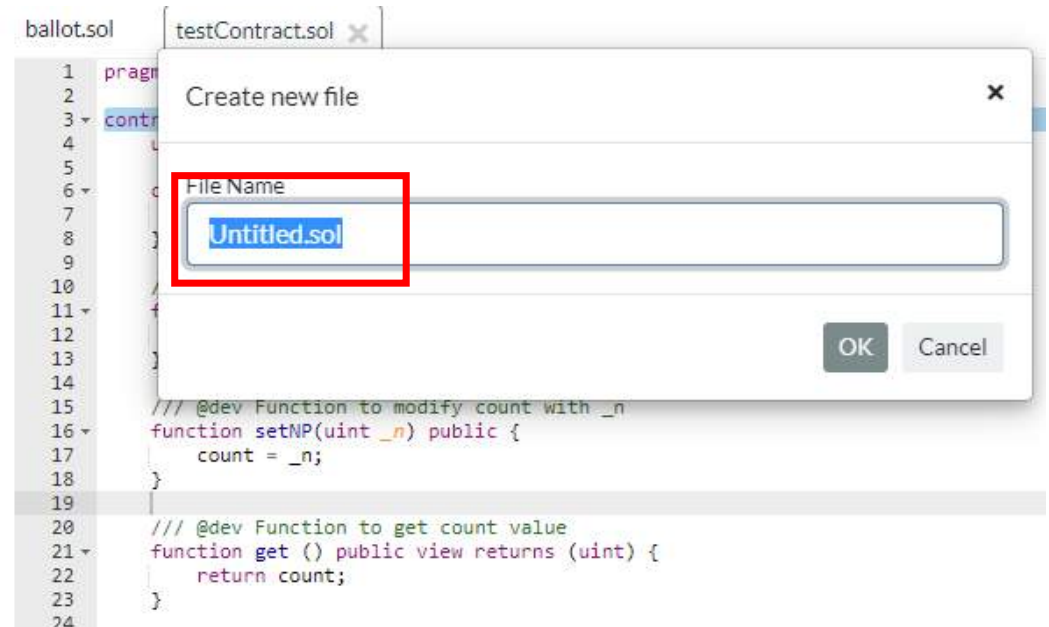
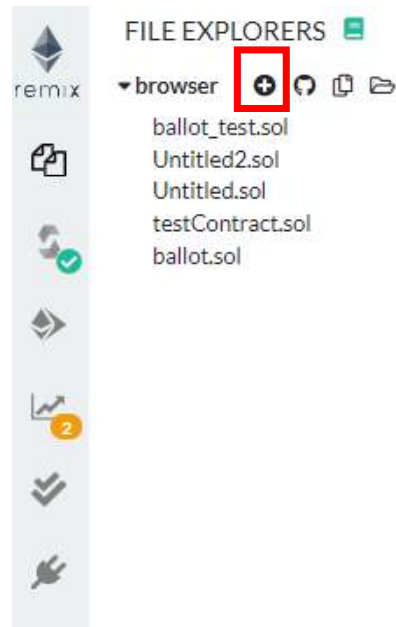
- Welcome to Remix v0.8.5 -

You can use this terminal for:

- Checking transactions details and start debugging.
- Running JavaScript scripts. The following libraries are accessible:
 - web3 version 1.0.0
 - ethers.js
 - swarm
 - remix (run remix.help() for more info)
- Executing common command to interact with the Remix interface (see list of commands above). Note that these commands can also be included and run from a JavaScript script.
- Use exports.register(key, obj).remove(key).clear() to register and reuse object across script executions.

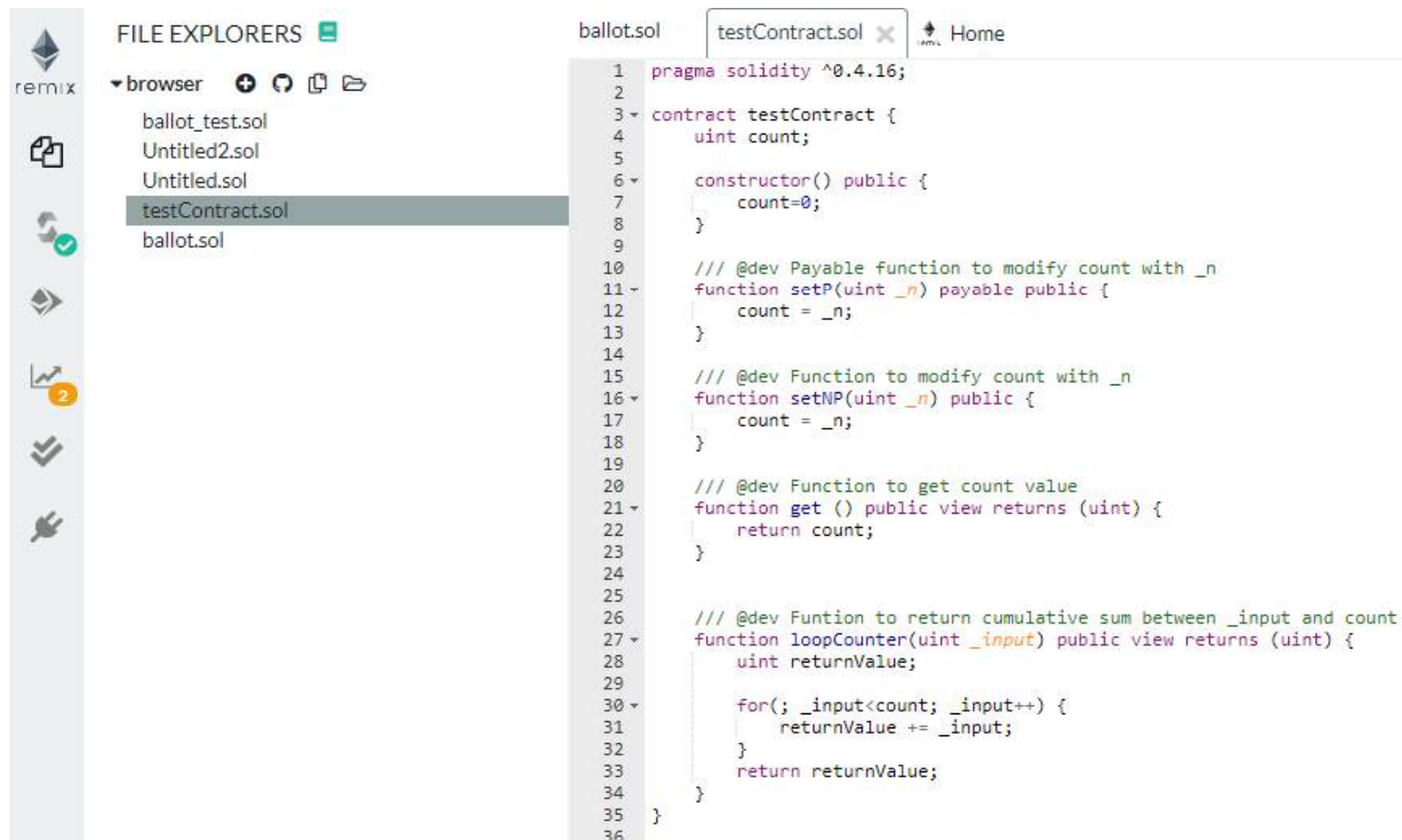
Remix Cont.

- Contract 파일 생성



Remix Cont.

- Contract 작성



Remix Cont.

- Contract 컴파일

1. 컴파일 메뉴 클릭

2. 컴파일러 버전 선택

3. 컴파일 버튼 클릭

Opt. 컴파일 산출물 확인

The screenshot displays the Remix IDE interface. On the left, the 'SOLIDITY COMPILER' panel is visible. It includes a 'Compiler' dropdown menu set to '0.4.26+commit.4563c', a 'Language' dropdown set to 'Solidity', and an 'EVM Version' dropdown set to 'compiler default'. A red box highlights the 'Compiler' dropdown. Below these, a large 'Compile testContract.sol' button is also highlighted with a red box. Underneath the button, there are checkboxes for 'Auto compile' (checked), 'Enable Optimization' (unchecked), and 'Hide warnings' (unchecked). At the bottom of the compiler panel, there is a 'Contract' dropdown set to 'testContract (testCon'. Below the compiler panel, there are buttons for 'Publish on Swarm' and 'Compilation Details'. A red box highlights the 'ABI' and 'Bytecode' links at the bottom of the compiler panel. On the right, the source code editor shows the 'testContract.sol' file with the following code:

```
1 pragma solidity ^0.4.16;
2
3 contract testContract {
4     uint count;
5
6     constructor() public {
7         count=0;
8     }
9
10    /// @dev Payable function to modify count with _n
11    function setP(uint _n) payable public {
12        count = _n;
13    }
14
15    /// @dev Function to modify count with _n
16    function setNP(uint _n) public {
17        count = _n;
18    }
19
20    /// @dev Function to get count value
21    function get () public view returns (uint) {
22        return count;
23    }
24
25    /// @dev Funtion to return cumulative sum between _input and
26    function loopCounter(uint _input) public view returns (uint)
27    {
28        uint returnValue;
29
30        for(; _input<count; _input++) {
31            returnValue += _input;
32        }
33        return returnValue;
34    }
35 }
36
37
38
```

Remix Cont.

• Contract 배포

1. 배포 환경 선택

- JavaScript VM(Remix 메모리상)
- Injected Provider(MetaMask)
- Web3 Provider(Geth)

2. 컨트랙트 배포

DEPLOY & RUN TRANSACTIONS

Environment: JavaScript VM

Account: 0xca3...a733c (100 ether)

Gas limit: 3000000

Value: 0 wei

testContract

Deploy

or

At Address Load contract from Address

Transactions recorded: 0

Deployed Contracts

Currently you have no contract instances to interact with.

Remix Cont.

✓ 배포 환경에 대한 공식 Doc

Run Setup


The following settings allow you to directly influence the transaction execution:




Environment:

- **JavaScript VM** : All the transactions will be executed in a sandbox blockchain in the browser. This means nothing will be persisted and a page reload will restart a new blockchain from scratch, the old one will not be saved.
- **Injected Provider** : Remix will connect to an injected web3 provider. **Metamask** is an example of providers that inject web3, thus can be used with this option.
- **Web3 Provider** : Remix will connect to a remote node. You will need to provide the URL address to the selected provider: geth, parity or any Ethereum client.
- **Account**: the list of accounts associated with the current environment (and their associated balances).
- **Gas Limit**: the maximum amount of gas that can be set for all the transactions created in Remix.
- **Value**: the amount of value for the next created transaction (this value is always reset to 0 after each transaction execution).


remix



DEPLOY & RUN TRANSACTIONS

Environment: JavaScript VM 

Account:  0xca3...a733c (99.9999999%)  

Gas limit: 3000000

Value: 0 wei 

testContract  




Deploy


or


At Address

Transactions recorded: **1** 


Deployed Contracts


 testContract at 0x692...77b3a (memory)  

setNP uint256 _n 

setP uint256 _n 



get

loopCounter uint256 _input 

Home testContract.sol 

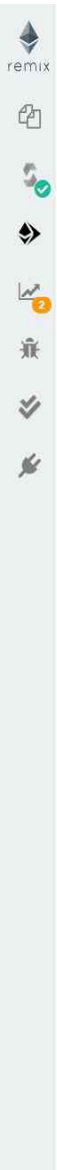
```
1 pragma solidity ^0.4.16;
2
3 contract testContract {
4     uint count;
5
6     constructor() public {
7         count=0;
8     }
9
10    /// @dev Payable function to modify count with _n
11    function setP(uint _n) payable public {
12        count = _n;
13    }
14
15    /// @dev Function to modify count with _n
16    function setNP(uint _n) public {
17        count = _n;
18    }
19
20    /// @dev Function to get count value
21    function get () public view returns (uint) {
```

  0 ☐ listen on network  Search with transaction hash or address

 [vm] from:0xca3...a733c to:testContract.(constructor) value:0 wei data:0x608...80029 logs:0 **Debug** 

hash:0xd57...3facd

status	0x1 Transaction mined and execution succeed
transaction hash	0xd57dd7a220adbe4f52f8306263cc562dc4174aa0e7e0b50c809f2071d593facd 
contract address	0x692a70d2e424a56d2c6c27aa97d1a86395877b3a 
from	0xca35b7d915458ef540ade6068dfe2f44e8fa733c 
to	testContract.(constructor) 
gas	3000000 gas 
transaction cost	159827 gas 
execution cost	80743 gas 
hash	0xd57dd7a220adbe4f52f8306263cc562dc4174aa0e7e0b50c809f2071d593facd 
input	0x608...80029 
decoded input	{ } 
decoded output	- 
logs	[]  
value	0 wei 



DEPLOY & RUN TRANSACTIONS

Environment: JavaScript VM

Account: 0xca3...a733c (99.999999999999798462 e)

Gas limit: 3000000

Value: 0 wei

testContract

Deploy

or

At Address: 0x692a70d2e424a56d2c6c27aa97d1a86395677b3a

Transactions recorded: 2

Deployed Contracts

testContract at 0x692...77b3a (memory)

setNP uint256_n

setP 10

get

loopCounter uint256_input

0: uint256: 24

```
1 pragma solidity ^0.4.16;
2
3 contract testContract {
4     uint count;
5
6     constructor() public {
7         count=0;
8     }
9
10    /// @dev Payable function to modify count with _n
11    function setP(uint _n) payable public {
12        count = _n;
13    }
14
15    /// @dev Function to modify count with _n
16    function setNP(uint _n) public {
17        count = _n;
18    }
19
20    /// @dev Function to get count value
21    function get () public view returns (uint) {
22        return count;
23    }
24
25    /// @dev Funtion to return cumulative sum between _input and count.
26    function loopCounter(uint _input) public view returns (uint) {
27        uint returnValue;
28        for (uint i=0; i<=_input; i++) {
29            returnValue += count;
30        }
31    }
32 }
```

ContractDefinition testContract 0 reference(s)

0 listen on network Search with transaction hash or address

[va] from:0xca3...a733c to:testContract.(constructor) value:0 wei data:0x608...80029 logs:0 hash:0xd57...3facd

transact to testContract.setP pending ...

[va] from:0xca3...a733c to:testContract.setP(uint256) 0x692...77b3a value:0 wei data:0xc95...0000a logs:0 hash:0x0c8...3ebfc

status: 0x1 Transaction mined and execution succeed

transaction hash: 0x0c82ba7832425b1742be0d89ab46a4d10fae97fadbd66b6192a7912493ebfc

from: 0xca3b7d915456e1540ade60681fe2144e6fa733c

to: testContract.setP(uint256) 0x692a70d2e424a56d2c6c27aa97d1a86395677b3a

gas: 3000000 gas

transaction cost: 41711 gas

execution cost: 20247 gas

hash: 0x0c82ba7832425b1742be0d89ab46a4d10fae97fadbd66b6192a7912493ebfc

input: 0xc95...0000a

decoded input: { "uint256 _n": { "_hex": "0x0a" } }

decoded output: {}

logs: []

value: 0 wei

Remix Cont.

• Contract 디버깅

함수 실행을 위한 바이트 코드를
Opcodes 형태로 표현

Transaction 실행 단계 슬라이더

Transaction 실행 단계 이동 버튼

- 이전 행으로 이동(Step over back)
- 뒤로 들어가기(Step back)
- 들어가기(Step into)
- 다음 행으로 이동(Step over forward)
- 이전 중단점으로 이동
- 밖으로 점프
- 다음 중단점으로 이동

DEBUGGER

0x0c82ba7832425b1742be0d89ab46a4d10fae97fadbbdbdb6192a...

Stop debugging

349 DUP1
350 PUSH1 00
352 DUP2
353 SWAP1
354 SSTORE
355 POP
356 POP
357 JUMP

vm trace step: 60

execution step: 60

add memory:

gas: 3

remaining gas: 2978314

loaded address: 0x692a70d2e424a56d2c6c27aa97d1a86395877b3a



Solidity Locals


_n: 10uint256








Solidity State

count: 0uint256

Remix Cont.

- Unit Testing


remix

SOLIDITY UNIT TESTING

Test your smart contract by creating a foo_test.sol file (open ballot_test.sol to see the example). You will find more informations in the [documentation](#). Then use the stand alone NPM module remix-tests to run unit tests in your Continuous Integration <https://www.npmjs.com/package/remix-tests>. For more details, see How to test smart contracts guide in our documentation.

Generate test file

Run Tests ☒ Check/Uncheck all

☐ browser/ballot_test.sol

☒ browser/testContract_test.sol

Results:

browser/testContract_test.sol (testContractTest)

✓ (Check get)

browser/testContract_test.sol

1 passing (0s)

Home testContract.sol ballot_test.sol ballot.sol testContract_test.sol

```
1
2 pragma solidity ^0.4.16;
3 import "remix_tests.sol"; // this import is automatically injected by Remix.
4 import "./testContract.sol";
5
6 contract testContractTest {
7
8     testContract testContractToTest;
9     function beforeAll () public {
10         testContractToTest = new testContract();
11     }
12
13     function checkGet () public {
14         Assert.equal(testContractToTest.get(), uint(0), "count is not zero");
15     }
16 }
17
18
```

Truffle(트러플)

- 스마트 컨트랙트 및 애플리케이션 개발을 위한 Node.js 기반 프레임워크
- 개발, 배포, 테스트 속도, 개발 생산성을 증가시키는 도구
- 설치방법
 - Node Package Manager(NPM)을 통해 노드 모듈로서 설치
C:\Users> npm install -g truffle

```
PS C:\vscode\ballot> truffle init

✓ Preparing to download
✓ Downloading
✓ Cleaning up temporary files
✓ Setting up box

Unbox successful. Sweet!

Commands:

  Compile:      truffle compile
  Migrate:      truffle migrate
  Test contracts: truffle test

PS C:\vscode\ballot>
```

Truffle(트러플) Cont.

• 사용방법

① Project 생성

③ Contract 폴더

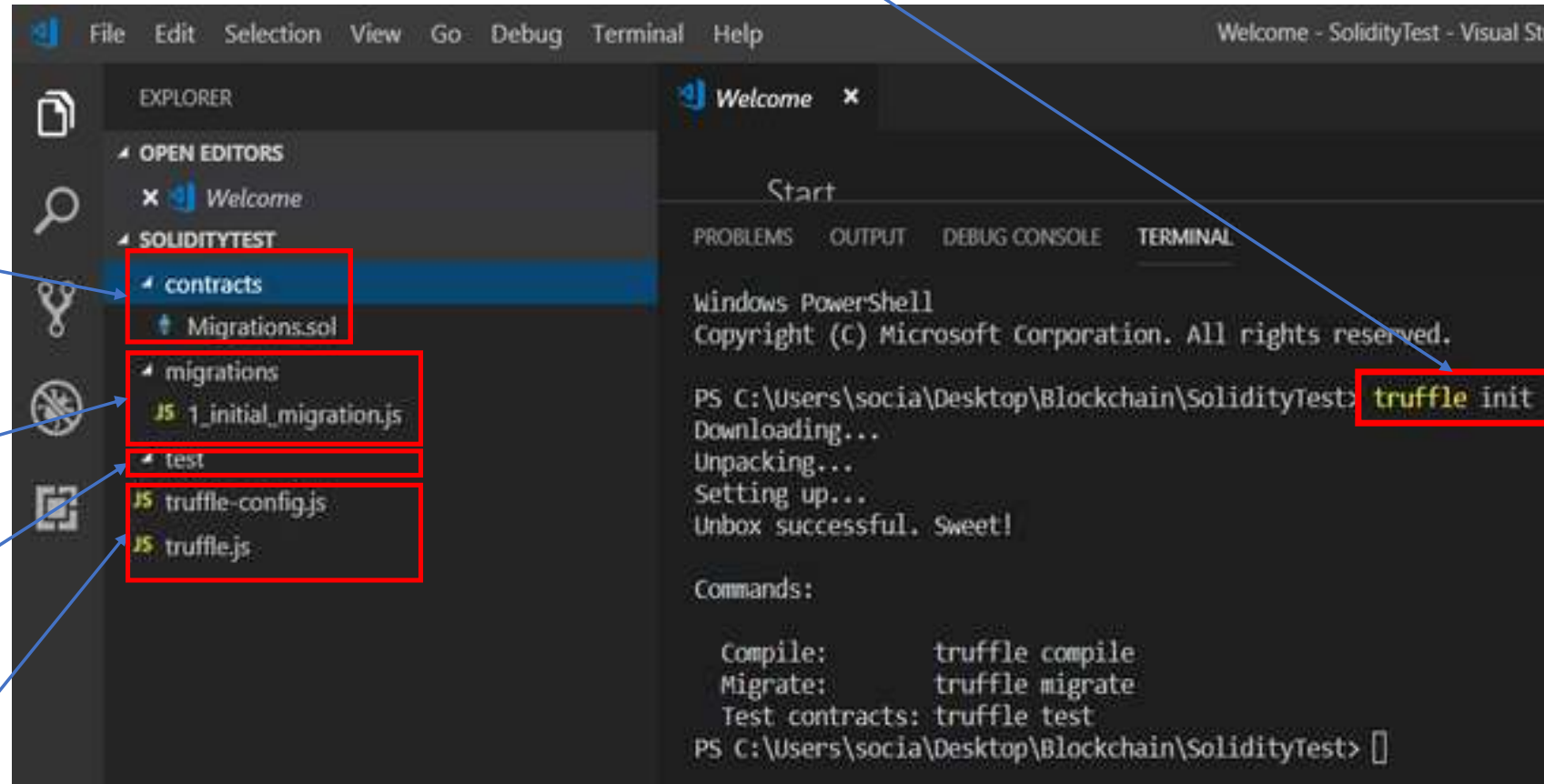
④ JS 파일들

For Contract 배포

Test Scripts

② Project 환경 구성

For 네트워크 연결, 계약 배포 등



Truffle(트러플) Cont.

- 컨트랙트 Compile
 - C:\Users> truffle compile
- 컨트랙트 deploy
 - C:\Users> truffle migrate
- 컨트랙트 Unit Testing
 - Test 폴더 안에 테스트 파일을 생성하여 테스트
 - Test 파일은 Solidity와 Js로 작성 가능
 - Test 예시는 생략...

Web기반 Dapp에서 스마트 컨트랙트 사용

Web3.js (자바스크립트 라이브러리)

- 웹 페이지 및 질의에서 사용할 수 있는 클라이언트 측 라이브러리
- 웹 애플리케이션에서 백엔드 이더리움 노드에 연결 가능
- 이더리움 노드에 거래 제출 가능
- Web3.js 공식홈페이지
 - <https://web3js.readthedocs.io/en/1.0/>
- 설치방법
 - Node Package Manager(NPM)을 통해 노드 모듈로서 설치
C:\Users> npm install web3@버전

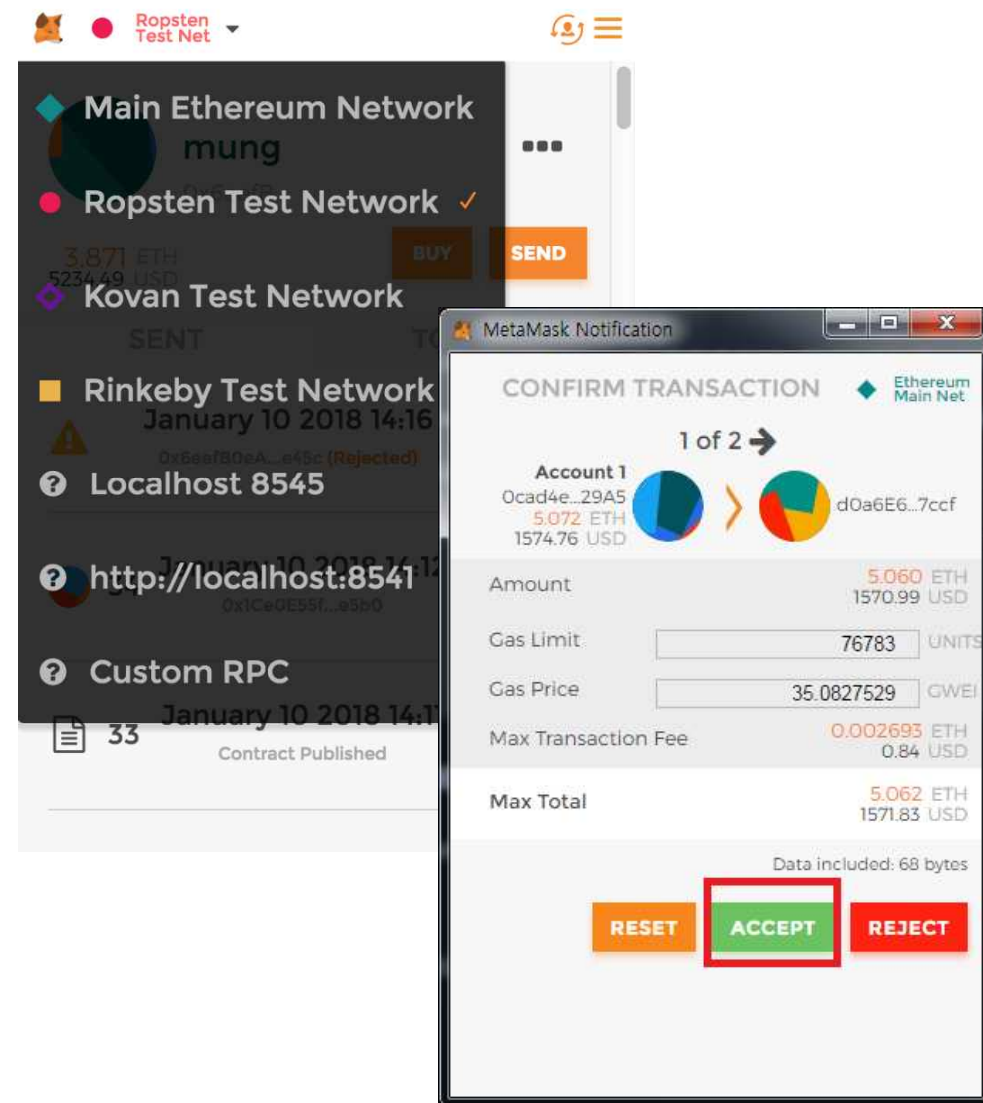
Web3.js Cont.

사용 예시)

```
const Web3 = require('web3')
const url = 'YOUR INFURA URL'
const web3 = new Web3(url)
const testabi = 'YOUR ABI'
const contractAddress = 'YOUR SMARTCONTRACT ADDRESS'
async function func() {
  const test_contract = new web3.eth.Contract(testabi, contractAddress)
  const record = await test_contract.methods.get_data().call()
  console.log(record)
  //console.log('record: ' + record)
}
func()
```

MetaMask(메타마스크)

- 이더를 주고받는 것을 돕는 지갑
- 크롬 브라우저 확장 프로그램 형태
- 전체 체인데이터를 로컬에 다운로드하지 않고 이더리움 네트워크와 상호작용 할 수 있음
- Main Net, Ropsten/Kovan/Rinkeby TestNet, Local, Custom RPC 연결 가능
- MetaMask 설치파일
 - <https://metamask.io/>



Q & A

추가 참고자료

- Geth로 스마트 컨트랙트 배포하기
 - <https://busy.org/@pangol/dapp-5-solc>
- Truffle을 이용하여 Dapp 개발하기
 - <https://medium.com/returnvalues/%EC%9D%B4%EB%8D%94%EB%A6%AC%EC%9B%80-dapp-%EA%B0%9C%EB%B0%9C-rental-car-1-2-7268e2e0060a>