카르다노에서 솔리디티로 스마트 컨트랙트 구현하기

2018. 12. 18

목차

1. 카르다노 테스트 넷

- KEVM
- IELE

2. 지갑 Mallet

- 설치 환경 구성(NVM 설치)
- Mallet 설치 & 테스트 넷 연결

3. 컨트랙트 작성과 테스트넷 Deploy

- KEVM Deployment with Mallet
- IELE Deployment with Mallet
- IELE Deployment with Remix

테스트 넷 KEVM

- 카르다노의 첫번째 테스트넷
- K 프레임워크를 준수하도록 수정된 이더리움 클래식 기반 EVM
- K는 공식적으로 소프트웨어를 검증하여 코드가 자동으로 결함을 검사할수 있는 수단 => 정확한 실행 검증
- KEVM 테스트넷은 K 프레임워크 사양으로 고안된 ETC 클라이언트인 Mantis 클라이언트에서 구현됨.
- JVM 1.8.x 이상이 필요 but JVM 1.9에서는 아직 테스트되지 않음

테스트 넷
IELE

- 두번째 테스트넷
- 스마트 컨트랙트의 공식적인 검증을 쉽게하기 위해 고안
- 카르다노 블록체인 프로토콜을 지원하는 **가상머신**
- 고급언어의 컨트랙트들을 번역하고 실행하기 위한 low-level 플랫폼
- 코드를 자체 IELE 언어로 변환하여 실행
- IELE 언어는 LLVM 중간 표현과 유사한 Human-readable language
- JVM 1.8.x 이상이 필요 but JVM 1.9에서는 아직 테스트되지 않음

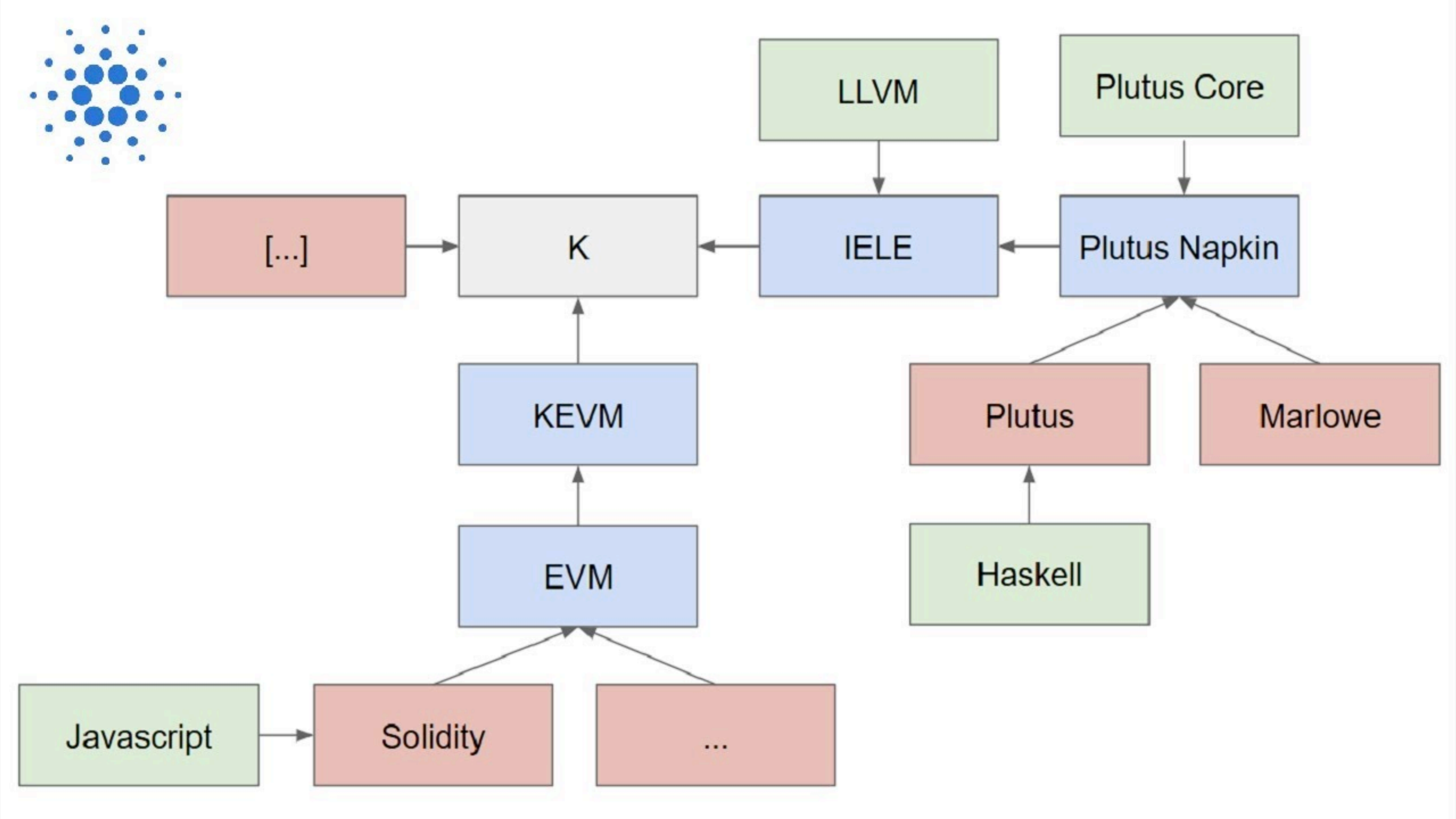
테스트 넷
IELE

KEVM과의
차이점

- IELE 테스트 넷은 레지스터 기반 VM으로 보다 광범위한 분석 및 최적화가 용이(KEVM은 스택 기반)
  - 계약에 대한 가스 비용을 낮출뿐 아니라 정확한 가스 비용 예측을 유도
  - 안정성 및 보안성 검증 용이
- 각 예외 유형에 대해 고유한 오류 코드가 존재 => 디버깅이 쉬움
- 함수를 호출하는 트랜잭션의 보낸 사람은 계정 호출의 정상 반환 값 외에도 반환 값으로 상태 코드를 받음

테스트 넷
IELE

KEVM과의 차이점



🟥 Red : Language

🟩 Green : Inpiration

🟦 Blue : Intermediate

⬜ White : Framework

지갑
Mallet

- Mallet은 카르다노 테스트 넷과의 상호작용을 위한 노드 CLI 도구

- dapp에 라이브러리로 포함도 가능

- git 필요 + **node.js 10.4** 이상 필요

- Package Manager Tool인 homebrew(mac), apt-get(linux), chocolatey(win)을 이용하면 더욱 간편하게 설치 가능

NVM
설치
Mac

```
brew install nvm(mac)

# Add nvm environment variables to your shell
echo "export NVM_DIR='$HOME/.nvm'" >> ~/.bash_profile
echo ". '/usr/local/opt/nvm/nvm.sh'"  >> ~/.bash_profile

# Load variables in current shell
source ~/.bash_profile

# Install 10.4 version of Node
nvm install 10.4

# Verify Version
nvm version
```

NVM 설치 Linux

```
# Install package to need
$ sudo apt-get install build-essential libssl-dev

# if you run install script right below, it will make .nvm directory in /home/ubuntu/
$ curl -o- https://raw.githubusercontent.com/creationix/nvm/v0.33.11/install.sh | bash

# Load variables in current shell
$ source ~/.bashrc

# Install 10.4 version of Node
nvm install 10.4

# Verify Version
nvm version
```

NVM 설치 Windows

# 기존 node 삭제
# 윈도우의 경우 제어판의 프로그램 제거에서 삭제하면 됩니다.

# nvm-setup.zip 다운받아 설치
https://github.com/coreybutler/nvm-windows/releases

# 터미널에서 $ nvm install v4.4.6 처럼 사용할 버전의 노드를 설치합니다.

# 노드 버전을 바꾸어봅니다. $ nvm use 4.4.6
node버전 확인 : $ node -v 만약 node가 설치 안된 것 처럼 나올 때는 터미널을 껐다 켜봅니다.

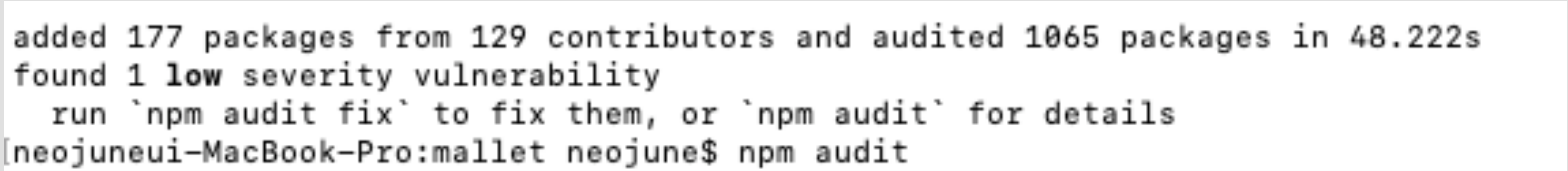
# gulp를 전역에 설치합니다. $ npm install --global gulp
# 주의할 점은 gulp가 버전별로 설치가 되어야 한다는 점입니다.
즉, $ nvm use 4.4.6 에서도 설치, $ nvm use 6.9.2로 바꿔서도 설치해줍니다.

Mallet
설치
&
테스트넷
연결

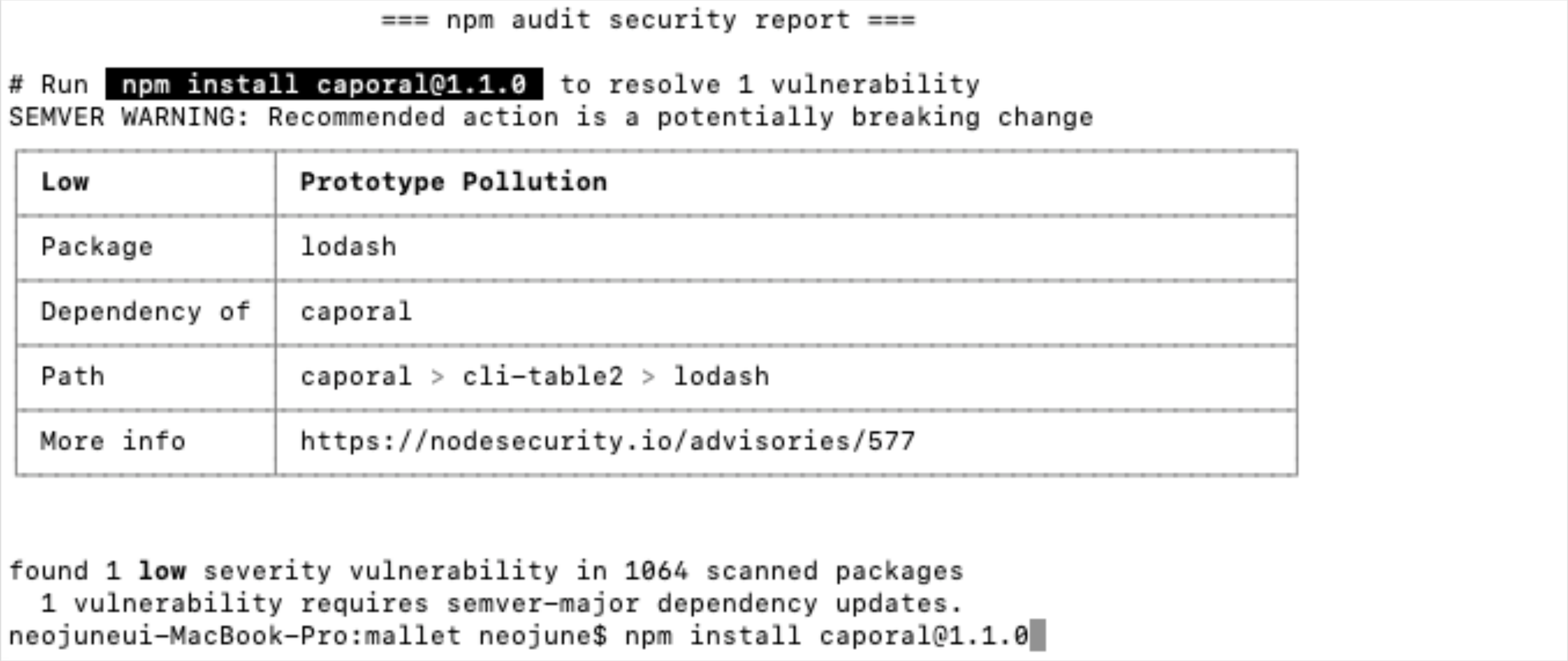
```
# Go to a Projects dir
cd <Cool Projects Dir>

# Pull Mallet
git clone https://github.com/input-output-hk/mallet && cd mallet

# Install dependencies
npm i

# Verify installation
./mallet --help

# Connect to the KEVM testnet
./mallet kevm -d <data_dir>

# Connect to the IELE testnet
./mallet iele -d <data_dir>
```

Mallet
설치
&
테스트넷
연결

# npm i 문제시 해결 방법



```
added 177 packages from 129 contributors and audited 1065 packages in 48.222s
found 1 low severity vulnerability
  run `npm audit fix` to fix them, or `npm audit` for details
[neojuneui-MacBook-Pro:mallet neojune$ npm audit
```



```
                       === npm audit security report ===

# Run  npm install caporal@1.1.0  to resolve 1 vulnerability
SEMVER WARNING: Recommended action is a potentially breaking change

┌───────────────┬──────────────────────────────────────────────────────────────┐
│ Low           │ Prototype Pollution                                          │
├───────────────┼──────────────────────────────────────────────────────────────┤
│ Package       │ lodash                                                       │
├───────────────┼──────────────────────────────────────────────────────────────┤
│ Dependency of │ caporal                                                      │
├───────────────┼──────────────────────────────────────────────────────────────┤
│ Path          │ caporal > cli-table2 > lodash                                │
├───────────────┼──────────────────────────────────────────────────────────────┤
│ More info     │ https://nodesecurity.io/advisories/577                       │
└───────────────┴──────────────────────────────────────────────────────────────┘


found 1 low severity vulnerability in 1064 scanned packages
  1 vulnerability requires semver-major dependency updates.
neojuneui-MacBook-Pro:mallet neojune$ npm install caporal@1.1.0
```

Contract 작성

```
# Create a project dir in the same root as your mallet repo
mkdir contract && cd contract

# You should have
# <Project Dir>
#    - mallet
#    - contract

# Save the hellocardano.sol file from the gist into the contract dir

# You should have
# <Project Dir>
#    - mallet
#    - contract
#      - hellocardano.sol
```

Contract 작성

```
pragma solidity ^0.4.21;

contract HelloCardano{
    uint num=1;

    function getNum() external view returns(uint){
        return num;
    }

    function setNum(uint param) external {
        num = param;
    }
}
```

KEVM Deployment with Mallet

```
# Install the solc npm package
npm install -g solc

# Verify it was installed
solc --help

# Compile hellocardano.sol
solc --bin --abi hellocardano.sol

# You should have
# <Project Dir>
#    - mallet
#    - contract
#      - hellocardano.sol
#      - hellocardano.abi
#      - hellocardano.bin
```

KEVM Deployment with Mallet

```
# Make sure you are using Node 10.4.x(See previous article)
nvm use 10

# Open Mallet CLI

cd ../mallet && mkdir kevm && ./mallet kevm -d kevm

# Create an Account - note we are assigning a variable to use later
mallet> account1 = newAccount("test0")

# Select account - note we are using our variable we set earlier
mallet> selectAccount(account1)

# Request Funds from faucet - note we are using the selected acct
mallet> requestFunds()
```

KEVM Deployment with Mallet

```
# Get balance - make sure you have a balance before proceeding
mallet> getBalance()

# Import the node.js fs module
mallet> fs = require('fs')

# Get the binary code
mallet> myContract = fs.readFileSync('../contract/hellocardano.bin', 'utf8')

# Create the transaction
mallet> tx = {
// gas limit, mandatory
    gas: 470000,
// the variable with our smart contract binary
    data: myContract
};
```

KEVM Deployment with Mallet

```
# Deploy the contract
mallet> deploymentHash = sendTransaction(tx)


# Verify the contract deployed
mallet> myContractAddress =
getReceipt(deploymentHash).contractAddress



mallet> sendTransaction({to: myContractAddress,
    gas:470000, data:0x67e0badb});
```

IELE Deployment with Mallet

```
# Make sure you are using Node 10.4.x(See previous article)
nvm use 10

# Open Mallet CLI
./mallet iele -d ./iele

# Create an Account - note we are assigning a variable to use later
mallet> account1 = newAccount("test0")

# Select account - note we are using our variable we set earlier
mallet> selectAccount(account1)

# Request Funds from faucet - note we are using the selected acct
mallet> requestFunds()
```

IELE Deployment with Mallet

```
# Get balance - make sure you have a balance before proceeding
mallet> getBalance()

# Compile the Solidity contract
# 자주 접속 문제가 발생하고 있음
mallet> myBytecode = iele.compile('../contract/hellocardano.sol').bytecode

# Deploy contract
mallet> iele.deployContract ({gas: 1000000, value: 0, code: myBytecode , args: []})
```

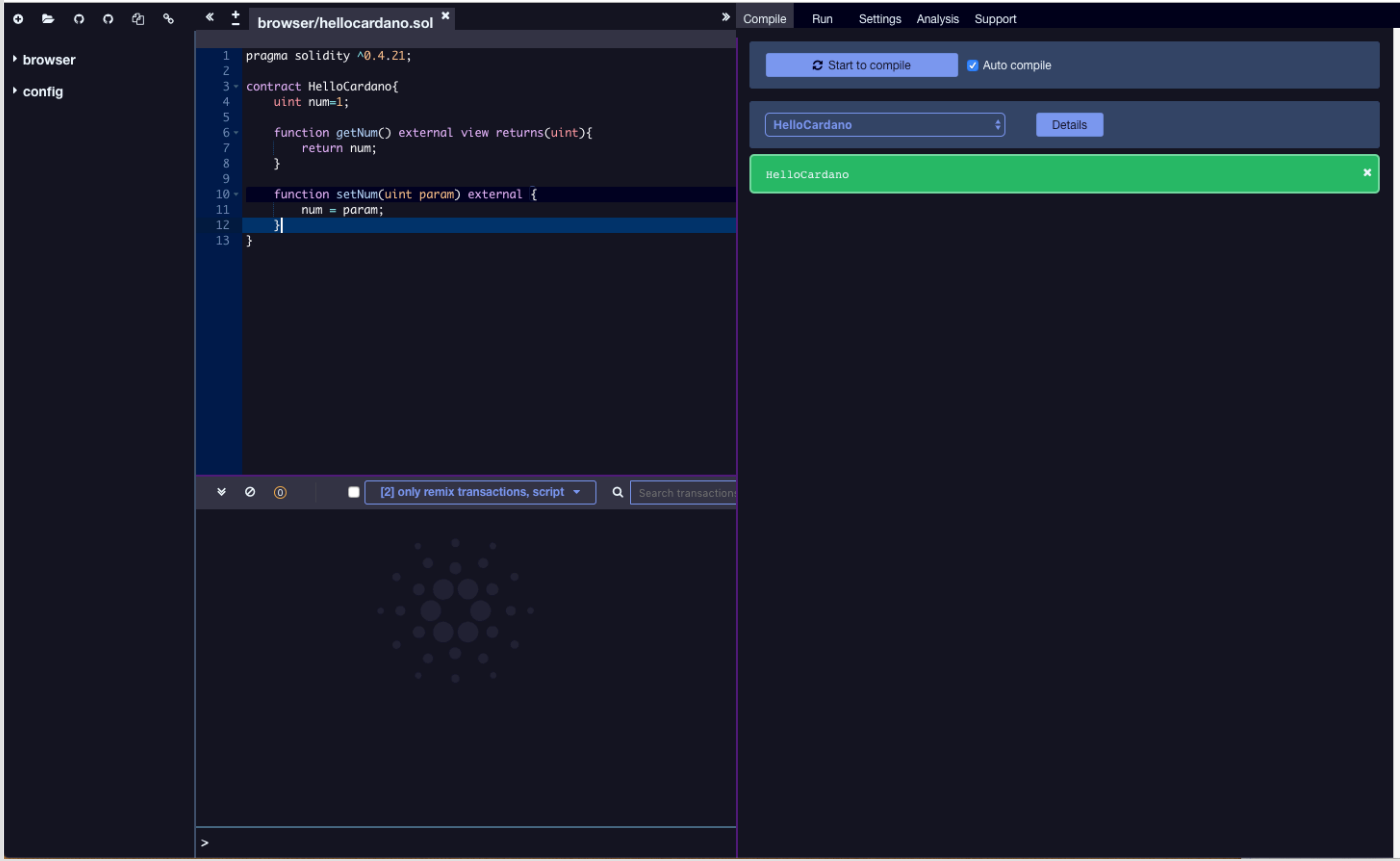
IELE Deployment with Mallet

```
# Get Deployed contract address

mallet> myContractAddress = getReceipt().contractAddress


# Interact with deployed contract

mallet> iele.callContract({to:myContractAddress, gas:1000000,

func:'getNum()', args:[]})


mallet> iele.callContract({to:myContractAddress, gas:1000000,

func:'setNum(uint)', args:[2]})
```

IELE Deployment with Remix

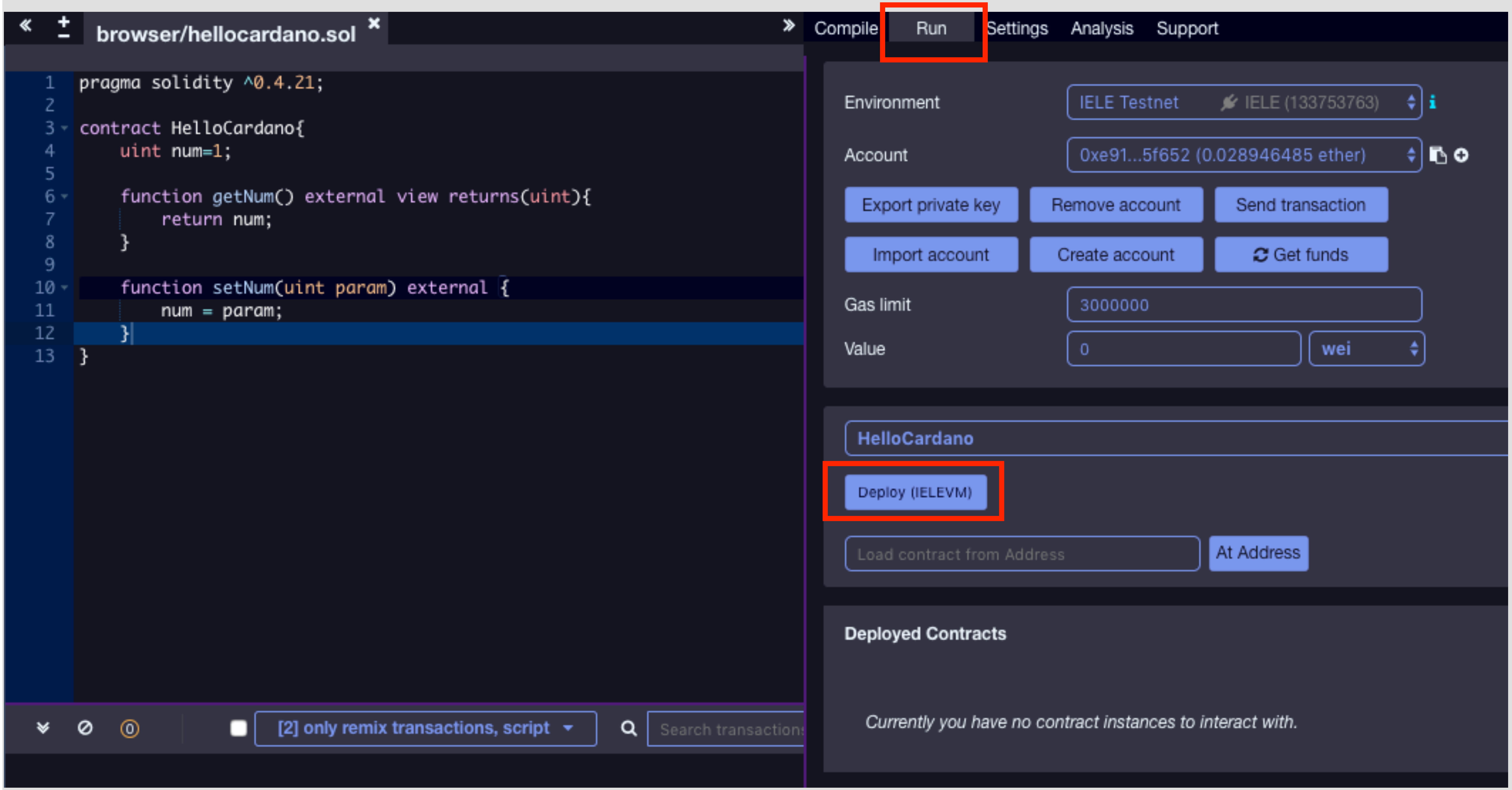
리믹스 주소

https://iele-testnet.iohkdev.io/remix/



```
pragma solidity ^0.4.21;

contract HelloCardano{
    uint num=1;

    function getNum() external view returns(uint){
        return num;
    }

    function setNum(uint param) external {
        num = param;
    }
}
```

IELE Deployment with Remix



```
pragma solidity ^0.4.21;

contract HelloCardano{
    uint num=1;

    function getNum() external view returns(uint){
        return num;
    }

    function setNum(uint param) external {
        num = param;
    }
}
```

Environment: IELE Testnet IELE (133753763)

Account: 0xe91...5f652 (0.028946485 ether)

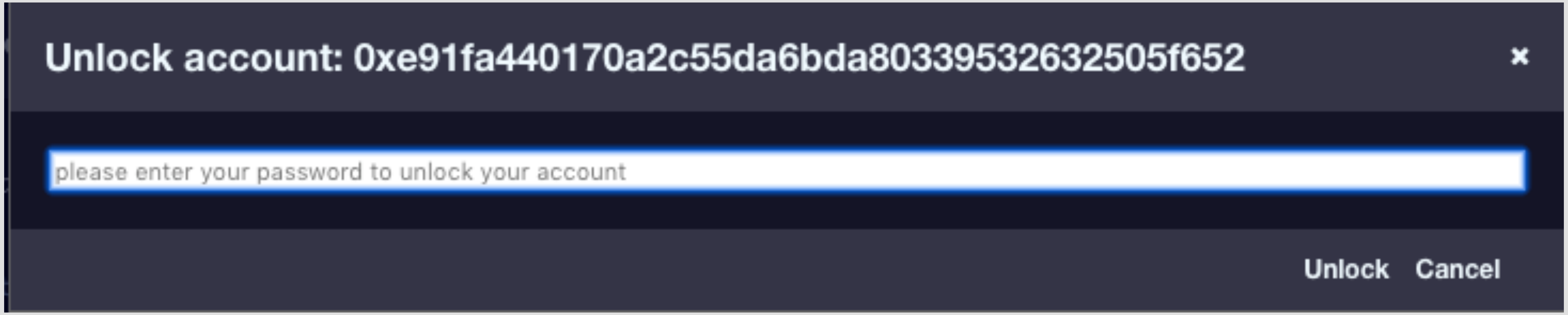
Gas limit: 3000000

Value: 0 wei

Deployed Contracts

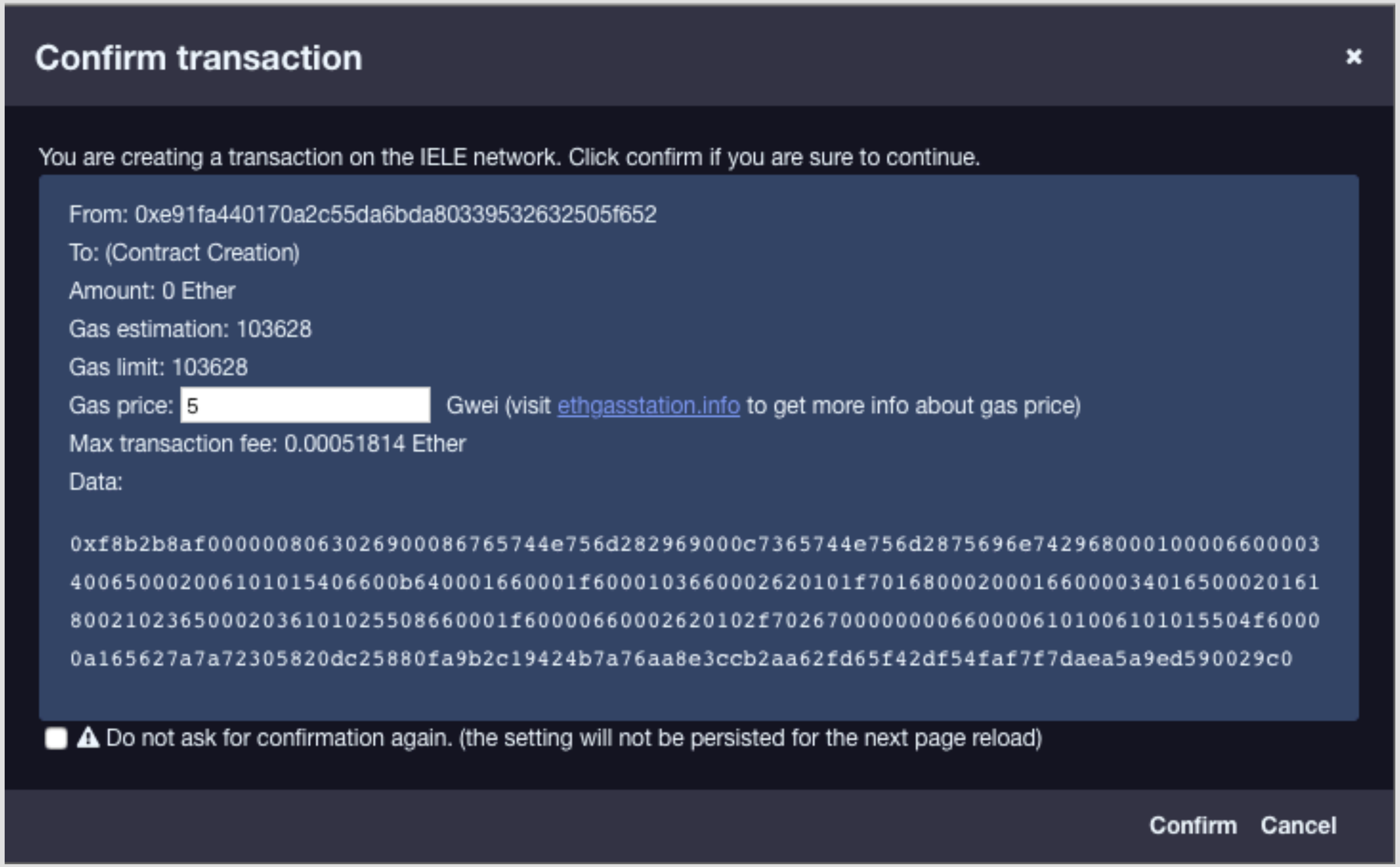
Currently you have no contract instances to interact with.

IELE Deployment with Remix



Unlock account: 0xe91fa440170a2c55da6bda80339532632505f652

please enter your password to unlock your account



Confirm transaction

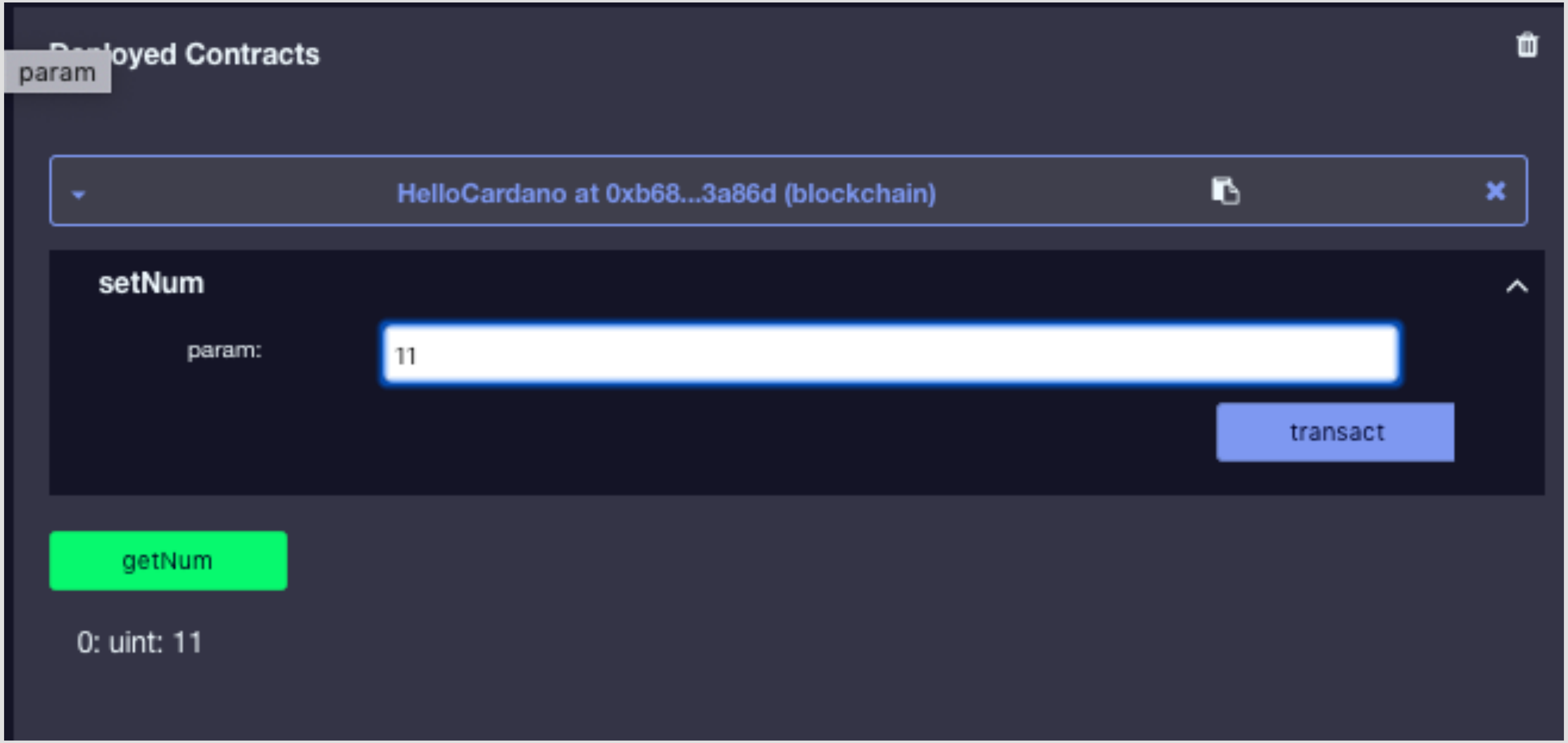
You are creating a transaction on the IELE network. Click confirm if you are sure to continue.

From: 0xe91fa440170a2c55da6bda80339532632505f652
To: (Contract Creation)
Amount: 0 Ether
Gas estimation: 103628
Gas limit: 103628
Gas price: 5 Gwei (visit ethgasstation.info to get more info about gas price)
Max transaction fee: 0.00051814 Ether
Data:

```
0xf8b2b8af000000806302690008676574 4e756d282969000c7365744e756d2875696e742968000100006600003
4006500020061010154066 00b640001660001f6000103660002620101f70168000200016600003401650002 0161
8002102365000203610102550866 0001f60000660002620102f70267000000000660000610100610101550 4f6000
0a165627a7a72305820dc25880fa9b2c19424b7a76aa8e3ccb2aa62fd65f42df54faf7f7daea5a9ed590029c0
```

☐ ⚠ Do not ask for confirmation again. (the setting will not be persisted for the next page reload)

IELE Deployment with Remix



HelloCardano at 0xb68...3a86d (blockchain)

param: 11

0: uint: 11

참조 사이트

IOHK 테스트넷 공식 튜토리얼
https://testnet.iohkdev.io/

카르다노 클라이언트 (kevm, iele) 설치 / 테스트넷 접속
https://medium.com/coinmonks/cardano-smart-contracts-101-testnets-f9dc7ac24635

스마트컨트랙트 hello world
https://medium.com/coinmonks/cardano-101-your-first-contract-ab22ec32e870

K Framework
https://runtimeverification.com/blog/k-framework-an-overview/

mallet 사용 가이드
https://github.com/input-output-hk/mallet/blob/master/README.md

IELE Remix
https://iele-testnet.iohkdev.io/remix