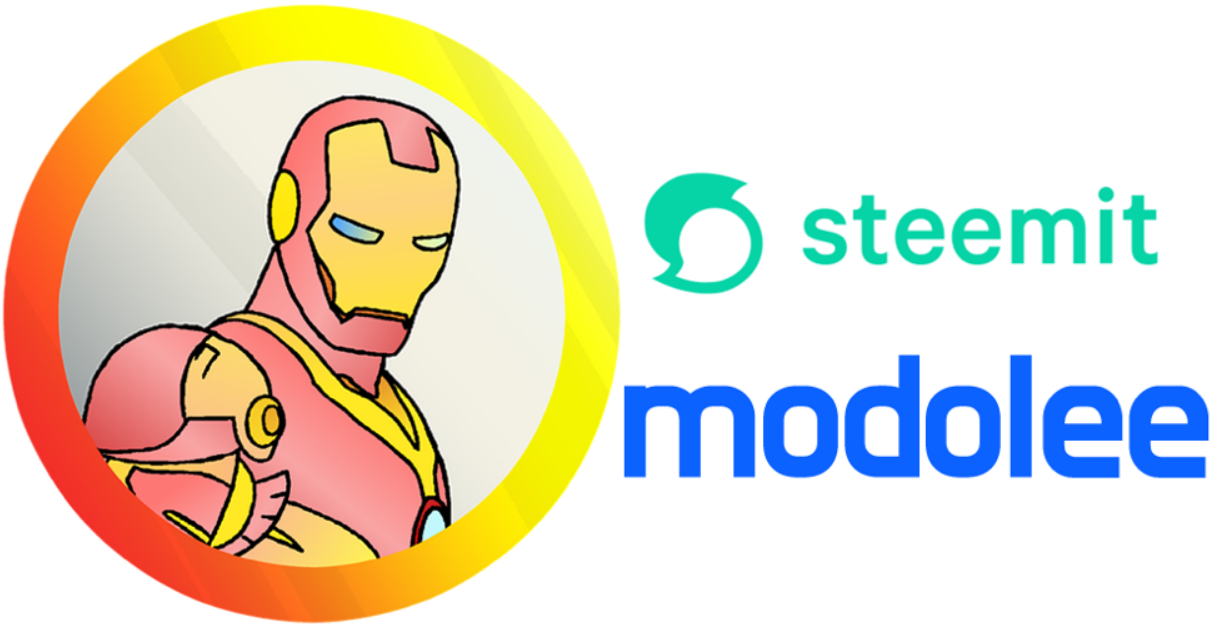
[Ethereum] Ethernaut 풀이 - 0.Hello Ethenaut

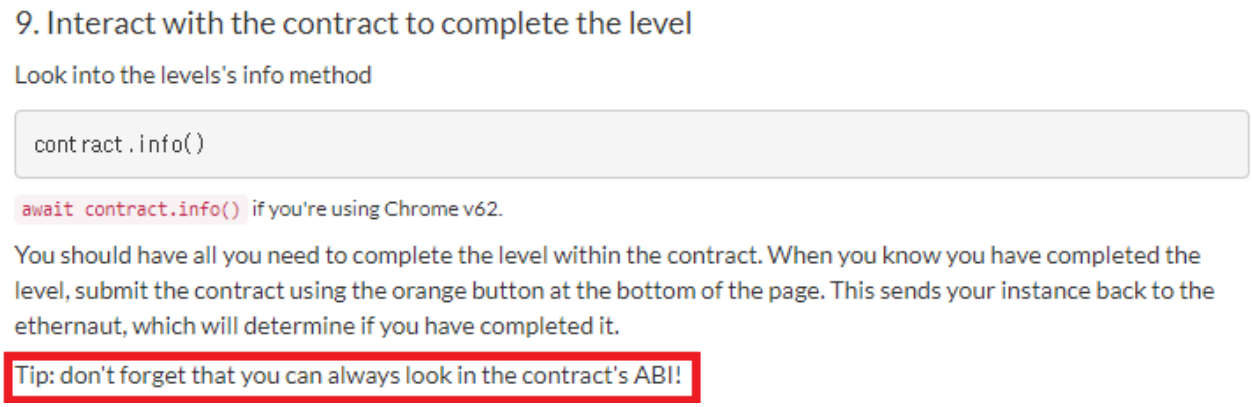
modolee (47) (/@modolee)in #kr-dev (/trending/kr-dev) • 4년 전



안녕하세요. 개발자 모도리입니다.
지난 포스팅(Ethernaut 소개 (https://steemit.com/kr-dev/@modolee/ethereum-ethernaut))에서는 Ethernaut을 소개하고 이용하기 전 준비 사항들을 알아봤습니다. 이번에는 본격적으로 Level 0 문제를 풀어보도록 하겠습니다.

0.Hello Ethernaut

지난 번 포스팅에서는 9번 항목의 맛보기까지만 진행을 했습니다.



TIP!을 잘 생각하면서 진행을 해보겠습니다.

contract.info()

- info 함수를 호출하면 아래와 같은 메세지가 나옵니다.

```
> await contract.info()
"You will find what you need in info1()."
```

- info1 함수를 호출해 봐야겠죠?

contract.info1()

- info1 함수를 호출한 화면입니다.

```
> await contract.info1()
"Try info2(), but with "hello" as a parameter."
```

- 이번에는 info2를 호출하라고 하는데, parameter로 "Hello"를 넣으라고 하네요.

contract.info2()

- 시키는데로 info2 함수에 parameter로 "Hello"를 넣어서 호출합니다.

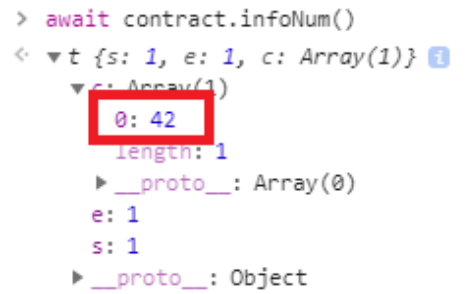
```
> await contract.info2("hello")
"The property infoNum holds the number of the next info method to call."
```

- infoNum이라는 property가 다음 info 함수의 숫자를 가지고 있는데, infoNum 이라는게 있는건지 Num 대신 무슨 숫자를 넣어야 될지 조금 헷갈립니다.
- 이때 아까 TIP!!!이 있었죠. contract의 ABI를 확인해 보겠습니다. `contract` 명령어를 입력합니다.

- 실제로 infoNum이라는 함수가 존재하네요.

contract.infoNum()

- 존재를 확인했으니 infoNum 을 호출해 봅니다.



- 분명 infoNum이 다음 info 함수의 숫자를 가지고 있다고 했는데, 그냥 봐서는 잘 모르겠습니다. 그래서 결과 값을 한번 펼쳐봤습니다. 숫자가 몇 개 있는데, 그 중 42가 눈에 띕니다.
- 혹시 모르니 ABI를 확인해 보겠습니다.

- info42 함수가 존재하네요.

contract.info42()

- info42 함수를 호출합니다.

```
> await contract.info42()
"theMethodName is the name of the next method."
```

- 메소드(함수) 이름이 theMethodName 이라니 설마... 했지만 정말 있습니다.

```
> contract
< ▼n {constructor: ƒ, abi: Array(11), contract: u, password: ƒ, infoNum: ƒ, …}
  ▼abi: Array(11)
    ▶0: {constant: true, inputs: Array(0), name: "password", outputs: Array(1), payable:
    ▶1: {constant: true, inputs: Array(0), name: "infoNum", outputs: Array(1), payable: f
    ▶2: {constant: true, inputs: Array(0), name: "theMethodName", outputs: Array(1), paya
    ▶3: {inputs: Array(1), payable: false, stateMutability: "nonpayable", type: "construc
    ▶4: {constant: true, inputs: Array(0), name: "info", outputs: Array(1), payable: fals
    ▶5: {constant: true, inputs: Array(0), name: "info1", outputs: Array(1), payable: fal
    ▶6: {constant: true, inputs: Array(1), name: "info2", outputs: Array(1), payable: fal
    ▶7: {constant: true, inputs: Array(0), name: "info42", outputs: Array(1), payable: fa
    ▶8: {constant: true, inputs: Array(0), name: "method7123949", outputs: Array(1), paya
    ▶9: {constant: false, inputs: Array(1), name: "authenticate", outputs: Array(0), paya
    ▶10: {constant: true, inputs: Array(0), name: "getCleared", outputs: Array(1), payabl
     length: 11
    ▶__proto__: Array(0)
   address: "0x660755e287a33e8f536fbf0d6d07951ebaf94a5f"
  ▶allEvents: ƒ ()
  ▶authenticate: ƒ ()
  ▶constructor: ƒ ()
  ▶contract: u {_eth: n, transactionHash: null, address: "0x660755e287a33e8f536fbf0d6d079
  ▶getCleared: ƒ ()
  ▶info: ƒ ()
  ▶info1: ƒ ()
  ▶info2: ƒ ()
  ▶info42: ƒ ()
  ▶infoNum: ƒ ()
  ▶method7123949: ƒ ()
  ▶password: ƒ ()
  ▶send: ƒ (e)
  ▶sendTransaction: ƒ ()
  ▶theMethodName: ƒ ()
```

contract.theMethodName()

- 호출합니다. (도대체 언제까지 해야 될까요? ㅠㅠ)

```
> await contract.theMethodName()
< "The method name is method7123949."
```

contract.method7123949()

- 아시죠? 또 호출합니다.

```
> await contract.method7123949()
< "If you know the password, submit it to authenticate()."
```

- 갑자기 password를 알고 있냐고 물어봅니다. 하.... 설마....

- 친절하게도 public 함수로 password를 부를 수 있게 되어 있습니다.
- password를 확인해 보겠습니다.

```
> await contract.password()
"ethernaut0"
```

- 그리고 authenticate 함수도 확인해 보겠습니다.

```
> contract
< ▼n {constructor: ƒ, abi: Array(11), contract: u, password: ƒ, infoNum: ƒ, …}
  ▼abi: Array(11)
    ▶0: {constant: true, inputs: Array(0), name: "password", outputs: Array(1), payable:
    ▶1: {constant: true, inputs: Array(0), name: "infoNum", outputs: Array(1), payable: f
    ▶2: {constant: true, inputs: Array(0), name: "theMethodName", outputs: Array(1), paya
    ▶3: {inputs: Array(1), payable: false, stateMutability: "nonpayable", type: "construc
    ▶4: {constant: true, inputs: Array(0), name: "info", outputs: Array(1), payable: fals
    ▶5: {constant: true, inputs: Array(0), name: "info1", outputs: Array(1), payable: fal
    ▶6: {constant: true, inputs: Array(1), name: "info2", outputs: Array(1), payable: fal
    ▶7: {constant: true, inputs: Array(0), name: "info42", outputs: Array(1), payable: fa
    ▶8: {constant: true, inputs: Array(0), name: "method7123949", outputs: Array(1), paya
    ▶9: {constant: false, inputs: Array(1), name: "authenticate", outputs: Array(0), paya
    ▶10: {constant: true, inputs: Array(0), name: "getCleared", outputs: Array(1), payabl
     length: 11
    ▶__proto__: Array(0)
   address: "0x660755e287a33e8f536fbf0d6d07951ebaf94a5f"
  ▶allEvents: ƒ ()
  ▶authenticate: ƒ ()
  ▶constructor: ƒ ()
  ▶contract: u {_eth: n, transactionHash: null, address: "0x660755e287a33e8f536fbf0d6d079
  ▶getCleared: ƒ ()
  ▶info: ƒ ()
  ▶info1: ƒ ()
  ▶info2: ƒ ()
  ▶info42: ƒ ()
  ▶infoNum: ƒ ()
  ▶method7123949: ƒ ()
  ▶password: ƒ ()
  ▶send: ƒ (e)
  ▶sendTransaction: ƒ ()
  ▶theMethodName: ƒ ()
```

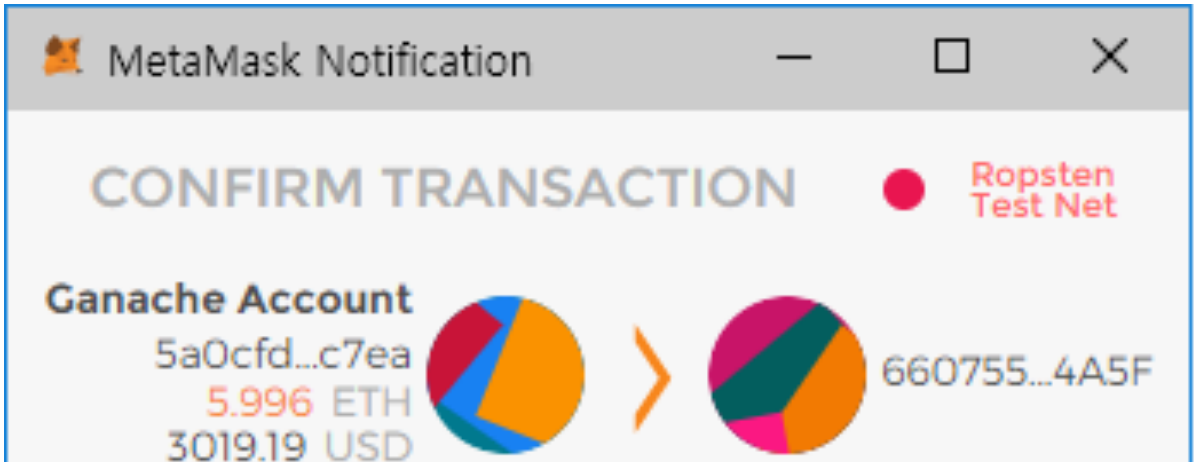
- authenticate 함수는 input으로 1개를 받는데, 아마도 password 겠죠?

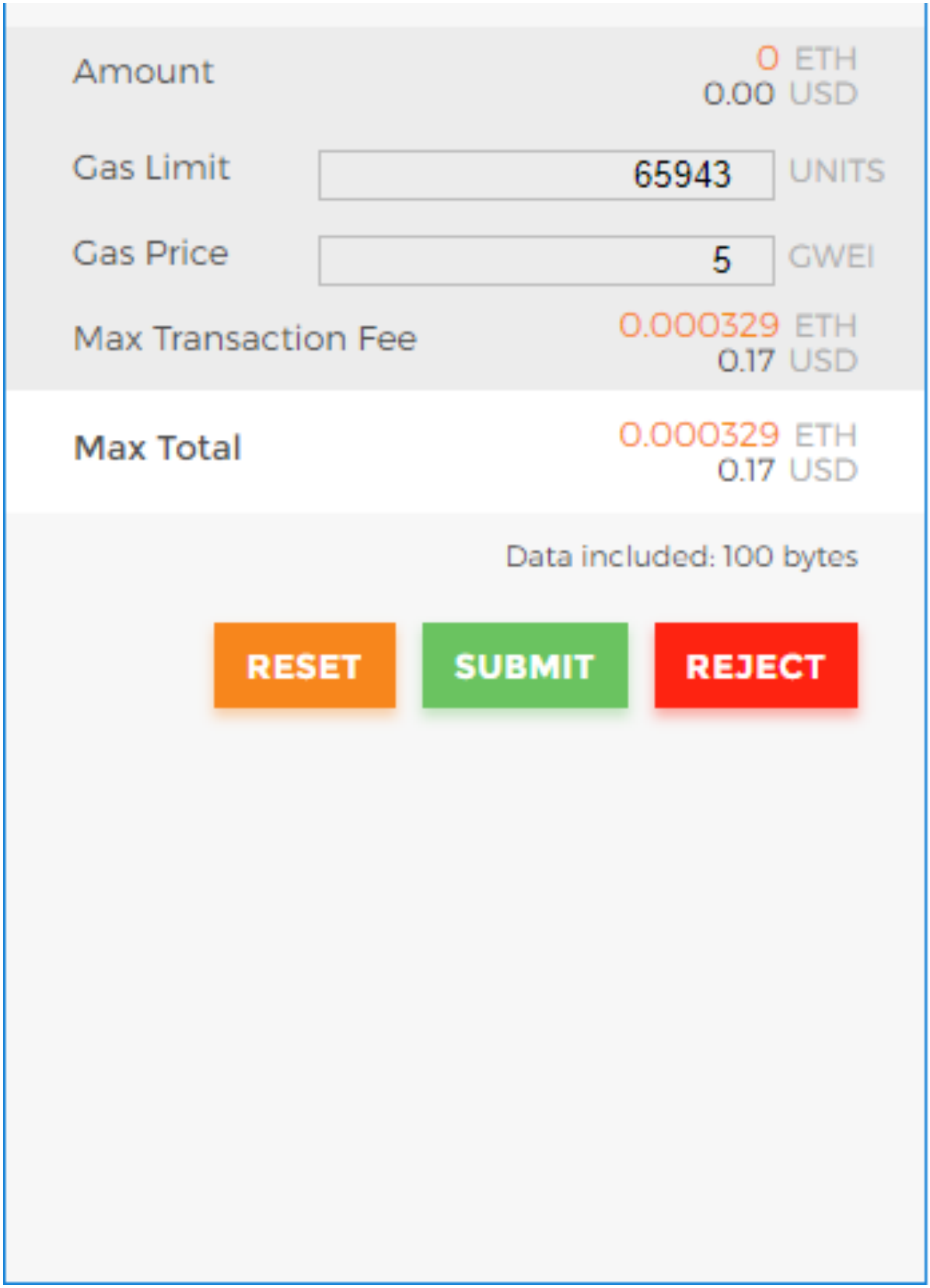
contract.authenticate()

- 앞서 확인한 password를 parameter로 해서 authenticate 함수를 호출합니다.

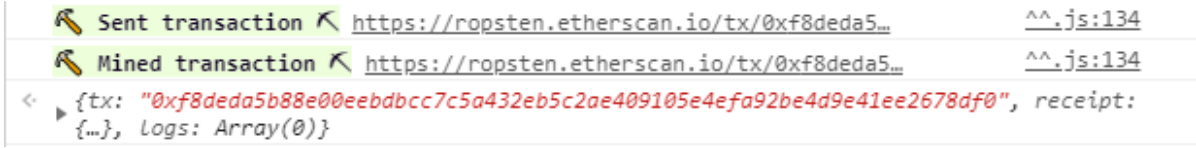
```
> await contract.authenticate("ethernaut0")
```

- 이거는 쓰기 함수 인가보네요. gas fee를 달라고 합니다.





- 적당한 가격을 넣고, submit을 누릅니다.
- 잠시 후 트랜잭션이 보내지고, 채굴까지 완료가 됩니다.



- 더 이상 요구하는게 없으니 완료된 것 같죠?

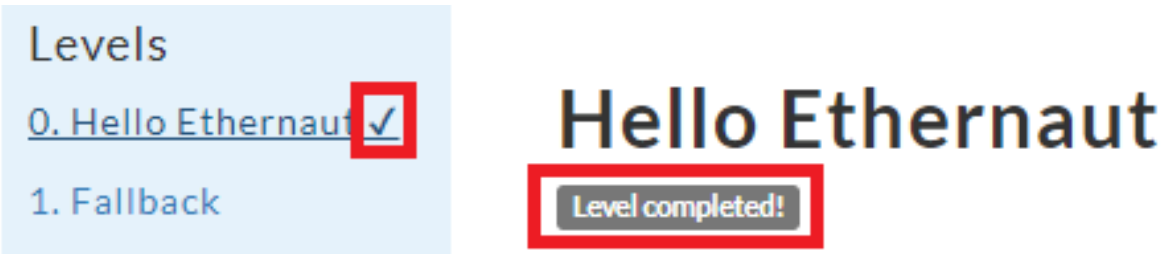
Submit instance

- level을 완료했다는 표시로 submit instance버튼을 누릅니다.

- 또 gas fee를 요구하네요. 이건 ethernaut smart contract에 현재 지갑 주소가 level을 complete 했다를 저장하기 위해 발생시키는 트랜잭션입니다.
- 채굴까지 정상적으로 완료가 되면, 아래와 같이 level completed 화면이 콘솔에 뜹니다.

Level Completed

- 임수를 완수 했다는 표시로 몇 가지 변화가 있습니다.
- 우선 체크 표시와 completed 배지가 생깁니다.



- 그리고 해당 instance의 smart contract 코드로 맨 아래 보여집니다.

```
pragma solidity ^0.4.18;

contract Instance {

  string public password;
  uint8 public infoNum = 42;
  string public theMethodName = 'The method name is method7123949.';
  bool private cleared = false;

  // constructor
  function Instance(string _password) public {
    password = _password;
  }

  function info() public pure returns (string) {
    return 'You will find what you need in info1().';
  }

  function info1() public pure returns (string) {
    return 'Try info2(), but with "hello" as a parameter.';
  }

  function info2(string param) public pure returns (string) {
    if(keccak256(param) == keccak256('hello')) {
      return 'The property infoNum holds the number of the next info method to call.';
    }
    return 'Wrong parameter.';
  }

  function info42() public pure returns (string) {
    return 'theMethodName is the name of the next method.';
  }


  function method7123949() public pure returns (string) {
    return 'If you know the password, submit it to authenticate().';
  }

  function authenticate(string passkey) public {
    if(keccak256(passkey) == keccak256(password)) {
      cleared = true;
    }
  }

  function getCleared() public view returns (bool) {
    return cleared;
  }
}
```

- 그리고 제일 중요한 다음 level로 넘어갈 수 있는 버튼이 새롭게 생겼습니다.

명령어 하나 하나씩을 설명해 가면서 문제를 풀어보았는데, 혹시나 중간에 이해가 안되는 부분이 있으시다면 편하게 댓글 남겨주시면 제가 아는 한도 내에서 정성껏 답변 드리겠습니다. 다음 문제도 계속 올리도록 하겠습니다. ^^

#kr (/trending/kr) #ethereum (/trending/ethereum) #solidity (/trending/solidity) #ethernaut (/trending/ethernaut)

4년 전 in #kr-dev (/trending/kr-dev) by

modolee (47) (/@modolee)

$0.11 | 3 보팅

댓글 달기 | 0

(/kr-dev/@modolee/ethereum-ethernaut-0-hello-ethenaut)