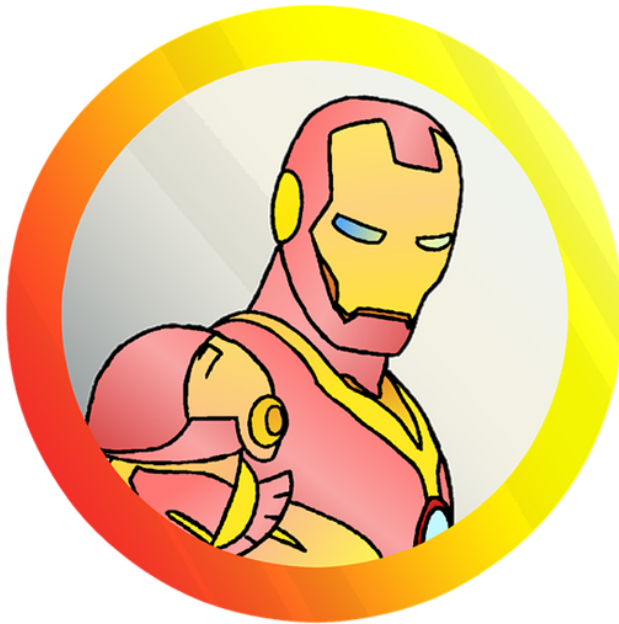


[Ethereum] Ethernaut 풀이 -

1.Fallback

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



 steemit
modolee

안녕하세요. 개발자 모도리입니다.

Ethernaut 문제 풀이 시리즈를 계속 진행해 보려고 합니다. 이번에는 Level 1 문제를 풀어보겠습니다. 지난 게시물은 아래를 확인해 주세요.

- Ethernaut 소개 (<https://steemit.com/kr-dev/@modolee/ethereum-ethernaut>)
- Ethernaut 풀이 - 0.Hello Ethenaut (<https://steemit.com/kr-dev/@modolee/ethereum-ethernaut-0-hello-ethernaut>)

1.Fallback

임무 확인

총 2가지 조건을 만족 시켜야 임무를 완수할 수 있습니다.

Look carefully at the contract's code below.

You will beat this level if

1. you claim ownership of the contract
2. you reduce its balance to 0

Things that might help

- How to send ether when interacting with an ABI
- How to send ether outside of the ABI
- Converting to and from wei/ether units -see help() command-
- Fallback methods

Get new instance

- contract의 ownership을 내 것으로 만들어라.
- contract의 balance를 0으로 만들어라.

새로운 인스턴스 생성

우선 기본적으로 콘솔을 띄운 상태에서, **Get new instance** 버튼을 누르면, MetaMask 창이 뜨는데 거기에서 submit 버튼을 눌러 트랜잭션을 발생시킵니다.

Fallback

^^.js:59

Type help() for a listing of custom web3 addons

^^.js:116

Annoying 'Slow network detected' message? Try Dev Tools settings -> User messages only or disable 'chrome://flags/#enable-webfont-intervention-v4'

^^.js:124

=> Level address

^^.js:38

0x234094aac85628444a82dae0396c680974260be7

=> Player address

^^.js:38

0x5a0cfd716b8a53ba093ee55d2115a29735dec7ea

=> Ethernaut address

^^.js:38

0xc833a73d33071725143d7cf7dfd4f4bba6b5ced2

^^.js:48

Requesting new instance from level...

< < <<PLEASE WAIT>> > >

Sent transaction <https://ropsten.etherscan.io/tx/0x260884>

^^.js:134

E...

Mined transaction <https://ropsten.etherscan.io/tx/0x260884>

^^.js:134

4e...

^^.js:31

{tx: "0x260884ec4b0f86265b2a0a77c0133d58de1b2ae79ad4570227a57e91d75a44b1", receipt: {...}, Logs: Array(1)}

=> Instance address

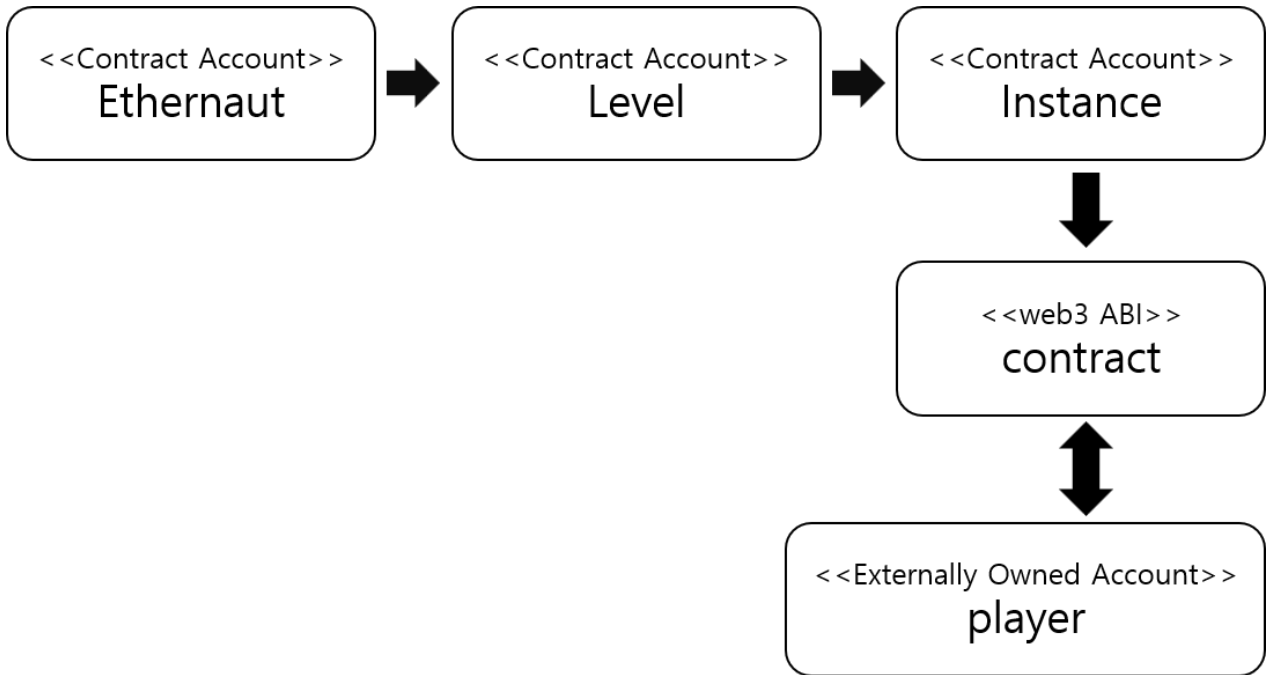
^^.js:38

0x1dc0ebf6f22fa474e5f62637a42f696c36176c1f

채굴까지 완료가 되면 인스턴스의 주소가 나옵니다.

여러 주소(계정)들의 관계 분석

콘솔 창에 여러가지 주소들이 나오는데 관계를 한번 분석해 보겠습니다.



- Ethernaut - Contract Account (CA)
 - Ethernaut 라는 해당 서비스 전체를 관장하는 컨트랙트입니다.
 - 각 Level 별 컨트랙트를 생성하고 임무 완수 여부 상태를 저장합니다.
 - 콘솔 명령 : ethernaut
- Level - Contract Account (CA)
 - Level 컨트랙트는 사용자가 요청하면 Instance 컨트랙트를 생성합니다.
 - 콘솔 명령 : level
- Instance - Contract Account (CA)
 - Level의 임무를 담고 있는 컨트랙트입니다.
 - 콘솔 명령 : instance
- Contract - Javascript 변수
 - Instance 컨트랙트를 web3와 ABI를 이용해서 console에서 접근할 수 있게 한 것입니다.
 - 콘솔 명령 : contract
- Player - Externally Owned Account (EOA)
 - MetaMask에 있는 계정입니다.
 - 콘솔 명령 : player

Solidity 코드 분석

아래 쪽에 보시면 현재 Level의 instance 배포에 사용된 Solidity 코드가 있습니다. 이 코드를 분석해서 임무를 완수해야 됩니다.

```
pragma solidity ^0.4.18;

import 'zeppelin-solidity/contracts/ownership/Ownable.sol';

contract Fallback is Ownable {

    mapping(address => uint) public contributions;

    function Fallback() public {
        contributions[msg.sender] = 1000 * (1 ether);
    }

    function contribute() public payable {
        require(msg.value < 0.001 ether);
        contributions[msg.sender] += msg.value;
        if(contributions[msg.sender] > contributions[owner]) {
            owner = msg.sender;
        }
    }

    function getContribution() public view returns (uint) {
        return contributions[msg.sender];
    }

    function withdraw() public onlyOwner {
        owner.transfer(this.balance);
    }

    function() payable public {
        require(msg.value > 0 && contributions[msg.sender] > 0);
        owner = msg.sender;
    }
}
```

- 버전 선언, 파일 import, 컨트랙트 선언

```
pragma solidity ^0.4.18;

import 'zeppelin-solidity/contracts/ownership/Ownable.sol';

contract Fallback is Ownable {

    mapping(address => uint) public contributions;
```

- Solidity 0.4.18 버전을 기준으로 작성되었습니다.(0.4.23 버전 이전에는 생성자를 컨트랙트 이름과 동일하게 선언했었는데, 0.4.23 버전부터는 constructor라는 이름으로 선언합니다.)
- Ownable.sol 파일을 import 합니다.
- Ownable은 OpenZeppelin에서 제공하는 표준 컨트랙트로 컨트랙트의 owner를 지정하고, 컨트랙트 함수의 실행 권한을 제한할 수 있습니다.

```
/**
 * @dev The Ownable constructor sets the original `owner` of
 * the contract to the sender account.
 */
constructor() public {
    owner = msg.sender;
}

/**
 * @dev Throws if called by any account other than the owner.
 */
modifier onlyOwner() {
    require(msg.sender == owner);
    _;
}
```

- 컨트랙트 생성 시 owner를 msg.sender로 지정하고, onlyOwner modifier를 이용하는 함수는 owner만 실행 가능하게 만들 수 있습니다.
- Fallback 컨트랙트는 Ownable 컨트랙트를 상속 받아서 선언합니다.
- address를 uint로 mapping 하는 contributions 상태 변수를 선언합니다.
- 생성자

```
function Fallback() public {
    contributions[msg.sender] = 1000 * (1 ether);
}
```

- msg.sender(컨트랙트 배포자)의 contributions 에 1000 ether를 저장합니다.
- 그리고 Ownable을 상속 받았기 때문에, owner가 msg.sender가 됩니다.
- await contract.owner() 명령으로 owner를 확인해 봅니다.
- ```
> await contract.owner()
```

```
< "0x234094aac85628444a82dae0396c680974260be7"
```
- Level contract가 instance를 배포 했으므로, owner로 설정되어 있습니다.
- **contribute 함수**
  - payable로 선언되어 ether를 전송받을 수 있습니다.
  - msg.value(보낸 ether)가 0.001 ether 미만이어야 아래 코드를 실행합니다.
  - 전송 받은 ether 만큼을 msg.sender(보낸 계정)의 contributions 값에 더합니다.
  - 만약 msg.sender의 contributions 값이 owner의 contributions 값보다 클 경우 *owner를 msg.sender에게 넘겨줍니다.*
- **getContribution 함수**
  - gas fee가 소모되지 않는, 단순 상태변수 읽기 함수이기 때문에 view로 선언되었습니다.
  - msg.sender의 contributions 값을 반환합니다.
- **withdraw 함수**
  - onlyOwner modifier가 붙어 있으므로, 컨트랙트 owner만 실행할 수 있습니다.
  - 현재 컨트랙트가 가지고 있는 *모든 ether 잔고를 owner에게 전송합니다.*
- **fallback 함수**
  - payable로 선언되어 ether를 전송받을 수 있습니다.



- msg.value(보낸 ether)가 0 초과이고, msg.sender(보낸 계정)의 contributions 값이 0 초과 일 때 **owner**를 **msg.sender**로 변경합니다.

## 문제 풀이

### 임무 완료 조건 다시 확인

1. owner가 되어라. -> owner를 변경하는 코드가 있는 contribute, fallback 을 살펴봐야 합니다.
2. 컨트랙트의 ether 잔고를 0으로 만들어라. -> owner가 된 후 withdraw를 호출하면 잔고를 0으로 만들 수 있습니다.

### owner가 되어 보겠습니다.

- contribute 함수를 통해서 owner가 되려고 한다면, 엄청난 노가다가 필요합니다.
- contribute에서 owner가 될 수 있는 조건은 현재 owner보다 contributions 값이 커져야 하는데, 생성자에서 최초 owner의 contributions을 1000 ether로 설정했습니다.
- 그리고 contribute의 코드 실행 조건은 0.001 ether 미만을 전송해야 하기 때문에, 0.0009 ether를 1,120,000번 전송해야 contributions이 1,008이 되어 owner가 될 수 있습니다. 설마 이 방법은 아니겠죠...
- 그러면 fallback 함수를 보겠습니다.
- owner가 될 수 있는 조건은 0 초과 ether를 전송하고, contributions도 0 초과이면 가능합니다.
- msg.value는 ether 전송시에 지정해 주면 되는데, contributions는 어떻게 해야 될까요?
- contribute 함수를 다시 보면 owner를 얻지는 못하더라도 전송한 ether 만큼을 contributions에 더해줍니다.
- 방법은! contribute 함수를 호출해서 contributions를 0 초과 값으로 만들어 놓은 상태에서, fallback 함수를 호출하면서 0 초과하는 ether를 전송하

면 owner가 될 수 있습니다.

- **contribute 함수 호출**

- `contract.contribute.sendTransaction({value:toWei(0.0009)})` 를 콘솔창에서 실행하면, contribute 함수를 호출하면서 0.0009 ether를 전송합니다.

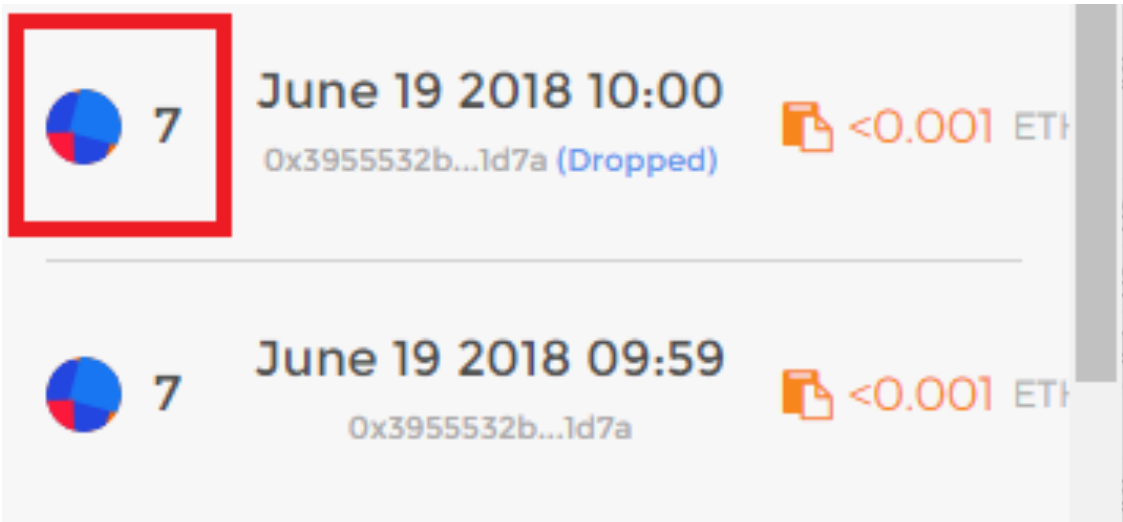
- MetaMask 창이 뜨면 Submit을 눌러주면 됩니다. 채굴이 빨리 되게 하려면 gas price를 넉넉하게 줍니다.

```
> contract.contribute.sendTransaction({value:toWei(0.0009)})
```

- `< ▶ Promise {<pending>}`

 Sent transaction  <https://ropsten.etherscan.io/tx/0xd1e2890...> [^^.js:134](#)

- 가끔 채굴이 되었는데, Mined transation 이라는 메시지가 안나오는 경우가 있습니다. 그때에는 MetaMask에서 transation이 ...이 아니라 여러 색상이 칠해져 있는 동그라미가 나온다면 transaction이 처리 된 것입니다.



- 그러면 `getContribution`, `getBalance` 함수를 이용해서 정상적으로 ether가 전송되었는지 확인해 보겠습니다.
- player의 contributions를 확인하기 위해 `fromWei(await contract.getContribution())` 명령을 입력합니다.
- `getContribution`은 기본적으로 wei 값을 반환하기 때문에 큰 숫자가 나옵니다. 그래서 ether로 표현하기 위해서 `fromWei` 함수를 이용했습니다. (`help()` 명령을 통해 확인할 수 있습니다.) contributions의 값이 0을 초과했습니다.
- instance의 ether 잔고를 확인하기 위해 `await`

getBalance(instance) 명령을 입력합니다.

- getBalance를 호출해서 컨트랙트의 ether 잔고를 확인하면 contribute를 통해서 전송 받은 ether 만크이 있습니다.

```
> fromWei(await contract.getContribution())
< "0.0009"
•
> await getBalance(instance)
< "0.0009"
```

## • fallback 함수 호출

- fallback 함수 호출 조건을 만족 시키려면 0 초과 ether를 전송해야 합니다.
- fallback 함수는 어떻게 호출할 수 있을까요?
- 가장 쉬운 방법은 그냥 MetaMask를 이용해서 instance address로 ether 전송 트랜잭션을 발생시키는 것입니다. (0 ether 전송도 가능합니다.)
- MetaMask 창을 열어서 SEND 버튼을 누르면 Recipient Address, Amount를 넣을 수 있는 칸이 있습니다.
- Recipient Address에는 instance의 address를 Amount에는 전송하고자 하는 ether 량(wei가 아닌 ether 단위)을 적고 NEXT를 누릅니다. 저는 contribute에 전송량과 동일하게 0.0009 ether를 전송했습니다.
- MetaMask에서 트랜잭션 처리가 완료 되었는지 확인한 후, owner를 확인해 보겠습니다.
- await contract.owner() 와 player 명령의 결과 값이 같다면, owner가 된 것입니다. (맨 처음 나왔던 player 주소와 다른 것은... 제가 여러 군데에서 작업을 해서 MetaMask 주소가 바뀌어서 그런 것입니다. 오해 없으시길 ^^;)

```
> await contract.owner()
< "0xf4690d756fa9dbf696da5c321b5260a6c48fbc6f"
•
> player
< "0xf4690d756fa9dbf696da5c321b5260a6c48fbc6f"
```

instance의 ether 잔고를 0으로 만들겠습니다.

- 우선 await getBalance(instance) 명령으로 잔고가 얼마 있는지 확인





해 보겠습니다.

- ```
> await getBalance(instance)
```



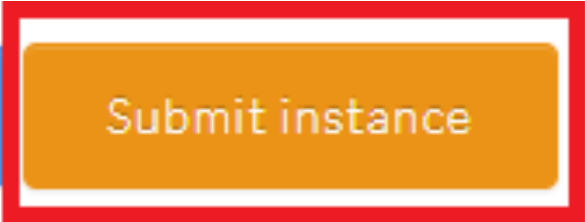
```
< "0.0018"
```
- contribute 호출 시 0.0009 ether, fallback 호출 시 0.0009 ether를 전송했으니 0.0018 ether가 있으면 정상입니다.
- owner가 되었으니 withdraw 함수를 호출할 수 있습니다.
- `contract.withdraw()` 명령을 실행하고, MetaMask로 트랜잭션을 발생시킵니다. 그리고 다시 한번 `await getBalance(instance)` 명령을 통해서 잔고를 확인합니다.

```
> contract.withdraw()
< ▶ Promise {<pending>}
```

-  Sent transaction  <https://ropsten.etherscan.io/tx/0x50d01df...> `^^.js:134`
 -  Mined transaction  <https://ropsten.etherscan.io/tx/0x50d01df> `^^.js:134`
- ```
...
> await getBalance(instance)
< "0"
```

## 답안 제출

- 2가지 임무를 모두 완수 했으니 Submit instance 버튼을 눌러서 답안을 제출합니다.

-  

- 트랜잭션을 발생시키고, 채굴이 완료되면 임무 완수 메시지가 나옵니다.



- 최근에 문제가 되었던 ICON의 ICX ERC-20 토큰 컨트랙트 코드에서도 사소한 부호 실수로 엄청난 불편을 일으키고 있습니다.

```
modifier onlyFromWallet {
 require(msg.sender != walletAddress);
 _;
}
```

- msg.sender가 walletAddress인 경우에만 실행할 수 있게 modifier를 만드려고 했는데, 부호를 잘못 사용하여 walletAddress를 제외한 모두(아무나) 접근할 수 있게 되어버렸습니다.

문제를 풀 때랑 이걸 포스팅으로 작성할 때 들어가는 시간의 차이가 많이 나네요

ㅠㅠ

하루 한 문제를 예상하고 있었는데, 문제가 어려워지면 기간이 더 걸릴 수도 있을 것 같습니다.

어차피 공부 목적으로 작성 중이니 기간에 압박 받지 않고 열심히 올려보겠습니다.

감사합니다.

[#kr \(/trending/kr\)](#)

[#ethereum \(/trending/ethereum\)](#)

[#solidity \(/trending/solidity\)](#)

[#ethernaut \(/trending/ethernaut\)](#)

🕒 4년 전 in [#kr-dev \(/trending/kr-dev\)](#) by

[modolee \(47\)](#) ▾ [./ \(@modolee\)](#)

📈 📉 \$0.13 ▾ 5 보팅 ▾

🔗 [댓글 달기](#) | 💬 0

[./kr-dev/@modolee/ethereum-ethernaut-1-fallback\)](#)



