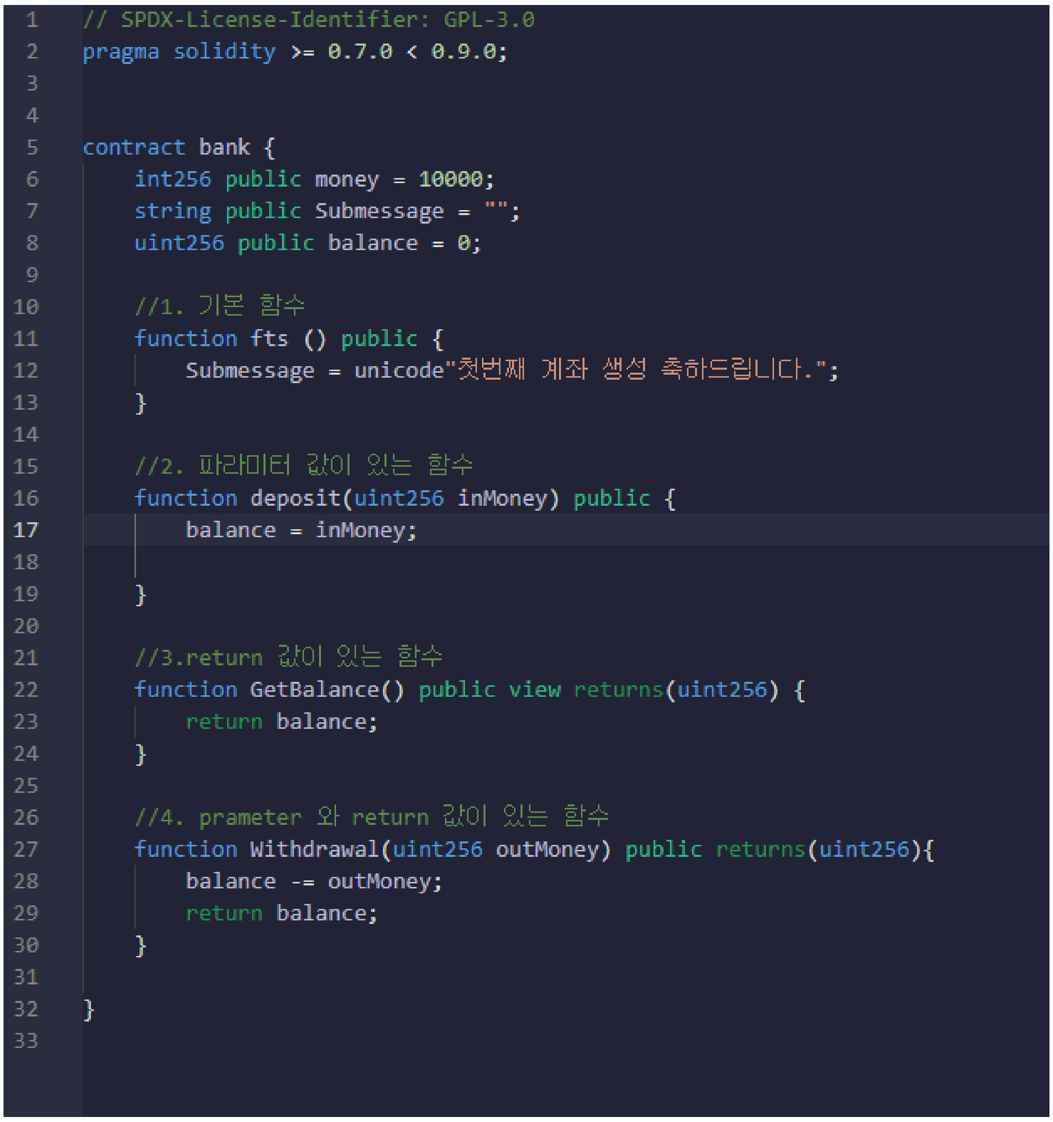
day67-sol

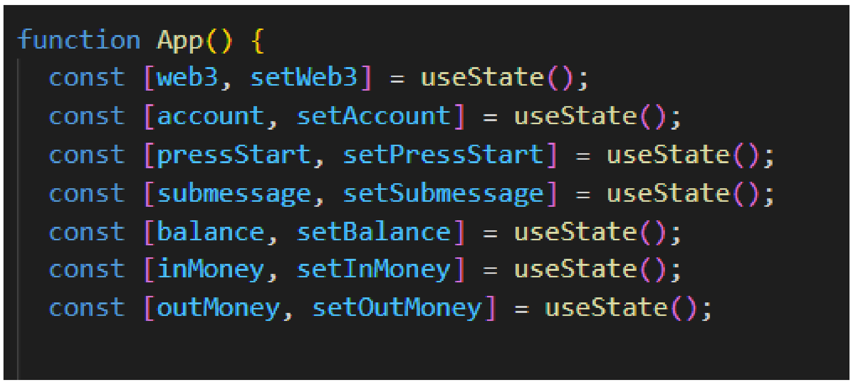
태그	
날짜	@2023년 1월 3일



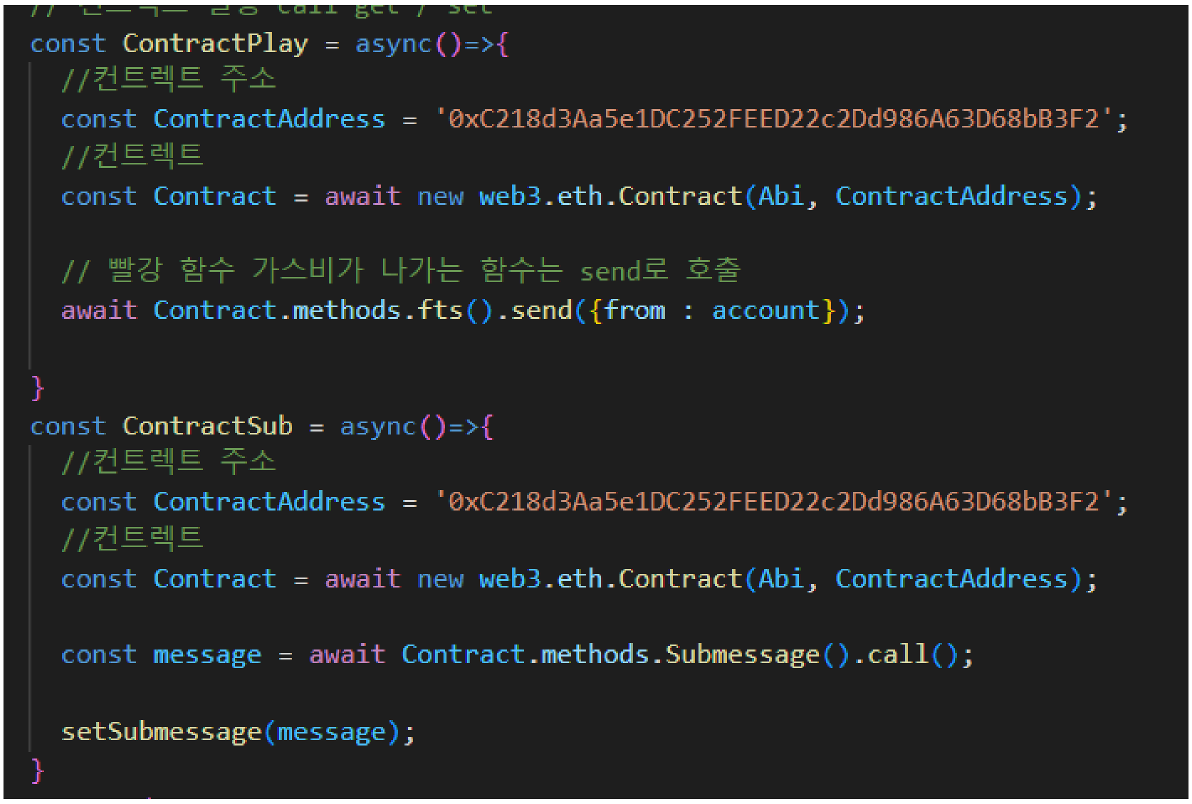
```solidity
// SPDX-License-Identifier: GPL-3.0
pragma solidity >= 0.7.0 < 0.9.0;


contract bank {
    int256 public money = 10000;
    string public Submessage = "";
    uint256 public balance = 0;

    //1. 기본 함수
    function fts () public {
        Submessage = unicode"첫번째 계좌 생성 축하드립니다.";
    }

    //2. 파라미터 값이 있는 함수
    function deposit(uint256 inMoney) public {
        balance = inMoney;

    }

    //3.return 값이 있는 함수
    function GetBalance() public view returns(uint256) {
        return balance;
    }

    //4. prameter 와 return 값이 있는 함수
    function Withdrawal(uint256 outMoney) public returns(uint256){
        balance -= outMoney;
        return balance;
    }

}

```

//빨강함수 가스비가 나가는 함수는 send
await Contract.methods.FirstSubscriber().send({from:account});

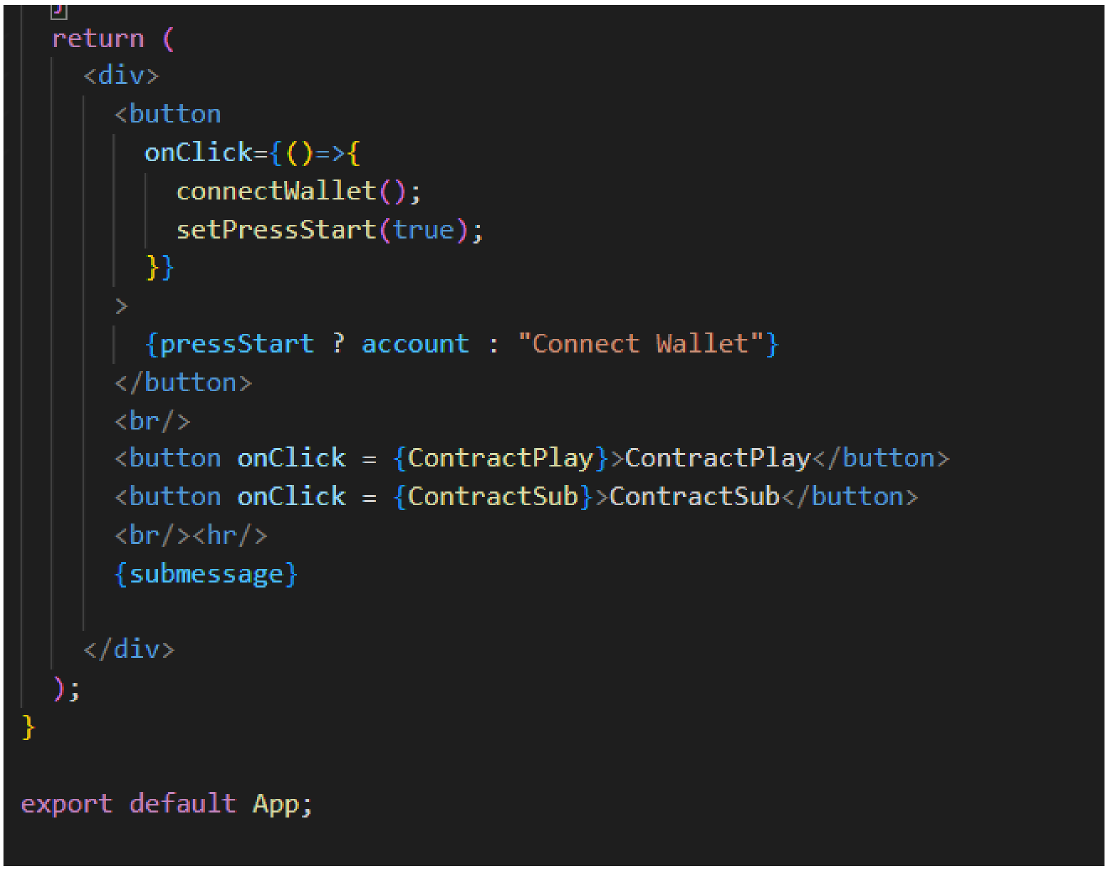
ContractSubMessage



```
function App() {
  const [web3, setWeb3] = useState();
  const [account, setAccount] = useState();
  const [pressStart, setPressStart] = useState();
  const [submessage, setSubmessage] = useState();
  const [balance, setBalance] = useState();
  const [inMoney, setInMoney] = useState();
  const [outMoney, setOutMoney] = useState();
```



```
// 컨트랙트 실행 call get / set
const ContractPlay = async()=>{
  //컨트렉트 주소
  const ContractAddress = '0xC218d3Aa5e1DC252FEED22c2Dd986A63D68bB3F2';
  //컨트렉트
  const Contract = await new web3.eth.Contract(Abi, ContractAddress);

  // 빨강 함수 가스비가 나가는 함수는 send로 호출
  await Contract.methods.fts().send({from : account});

}
const ContractSub = async()=>{
  //컨트렉트 주소
  const ContractAddress = '0xC218d3Aa5e1DC252FEED22c2Dd986A63D68bB3F2';
  //컨트렉트
  const Contract = await new web3.eth.Contract(Abi, ContractAddress);

  const message = await Contract.methods.Submessage().call();

  setSubmessage(message);
}
```



```
  return (
    <div>
      <button
        onClick={()=>{
          connectWallet();
          setPressStart(true);
        }}
      >
        {pressStart ? account : "Connect Wallet"}
      </button>
      <br/>
      <button onClick = {ContractPlay}>ContractPlay</button>
      <button onClick = {ContractSub}>ContractSub</button>
      <br/><hr/>
      {submessage}

    </div>
  );
}

export default App;
```

※프로그래밍에서 중요한 것

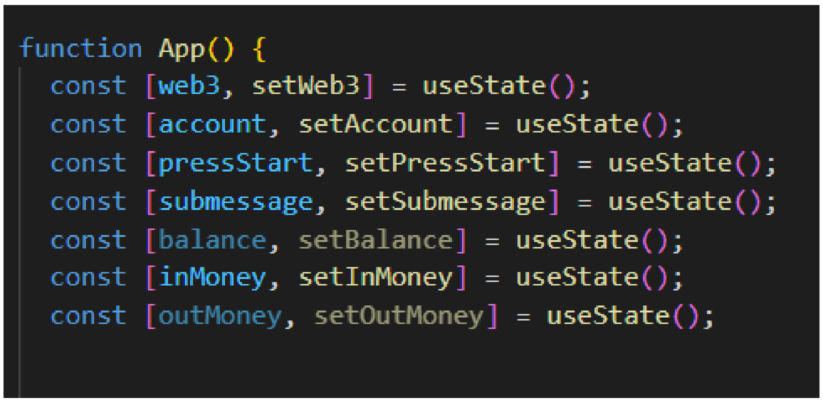
입력과 출력

변수

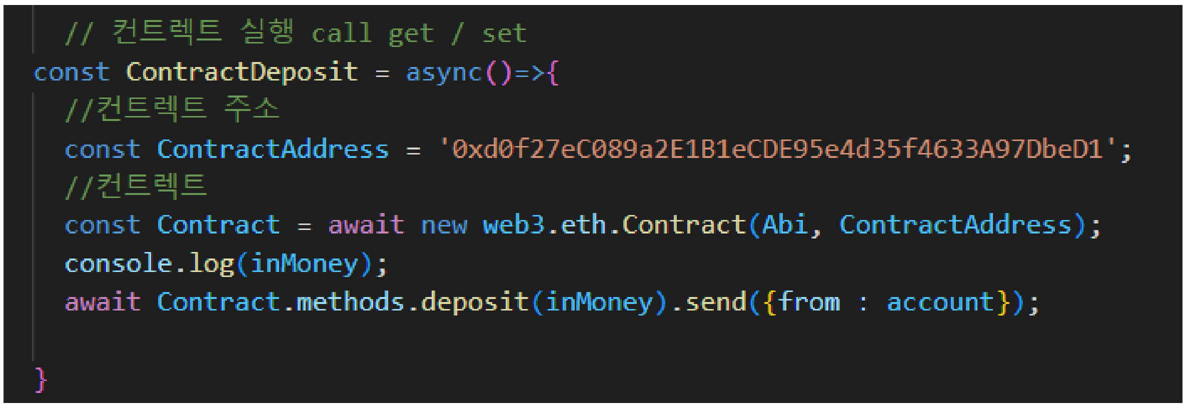
함수

클래스

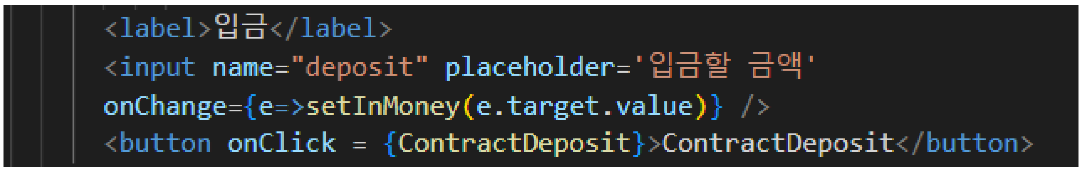
deposit



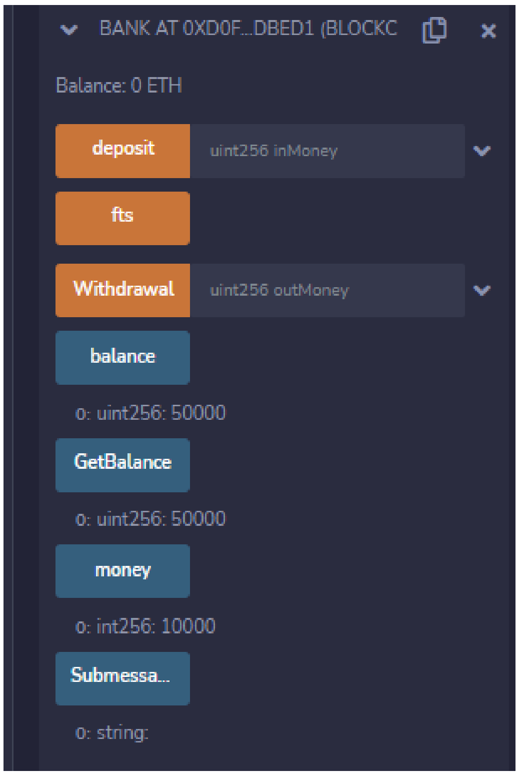
```
function App() {
  const [web3, setWeb3] = useState();
  const [account, setAccount] = useState();
  const [pressStart, setPressStart] = useState();
  const [submessage, setSubmessage] = useState();
  const [balance, setBalance] = useState();
  const [inMoney, setInMoney] = useState();
  const [outMoney, setOutMoney] = useState();
```



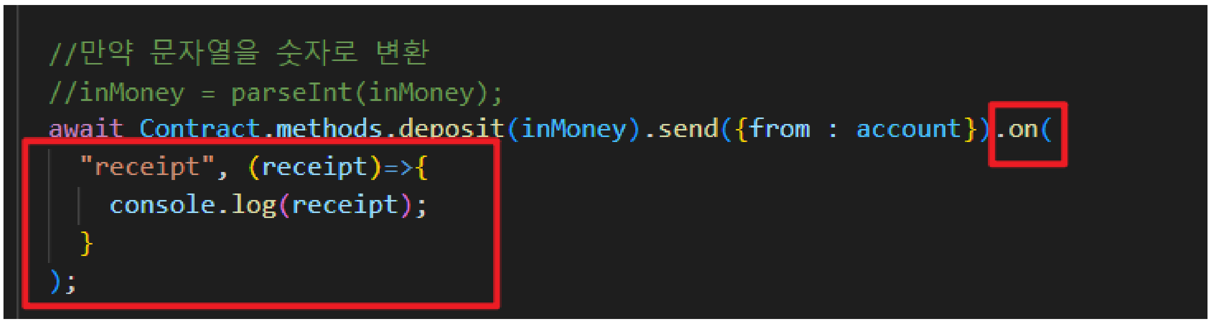
```
    // 컨트렉트 실행 call get / set
const ContractDeposit = async()=>{
    //컨트렉트 주소
    const ContractAddress = '0xd0f27eC089a2E1B1eCDE95e4d35f4633A97DbeD1';
    //컨트렉트
    const Contract = await new web3.eth.Contract(Abi, ContractAddress);
    console.log(inMoney);
    await Contract.methods.deposit(inMoney).send({from : account});

}
```



```
        <label>입금</label>
        <input name="deposit" placeholder='입금할 금액'
        onChange={e=>setInMoney(e.target.value)} />
        <button onClick = {ContractDeposit}>ContractDeposit</button>
```

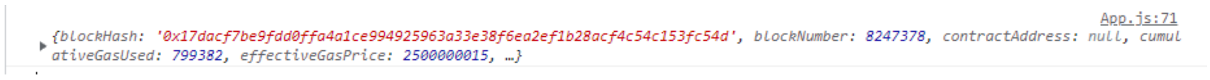


```
Balance: 0 ETH
0: uint256: 50000
0: uint256: 50000
0: int256: 10000
0: string:
```



```
//만약 문자열을 숫자로 변환
//inMoney = parseInt(inMoney);
await Contract.methods.deposit(inMoney).send({from : account}).on(
  "receipt", (receipt)=>{
    console.log(receipt);
  }
);
```

잘 보내졌는지 확인하는 방법



```
                                                                                                   App.js:71
▸ {blockHash: '0x17dacf7be9fdd0ffa4a1ce994925963a33e38f6ea2ef1b28acf4c54c153fc54d', blockNumber: 8247378, contractAddress: null, cumul
  ativeGasUsed: 799382, effectiveGasPrice: 2500000015, …}
```

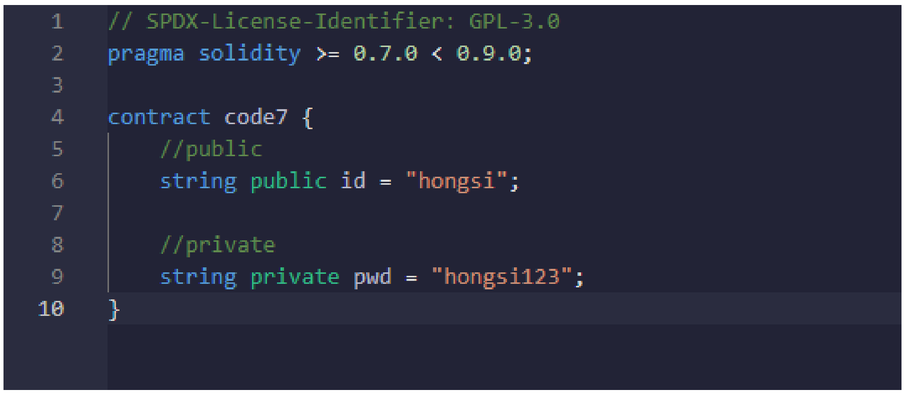
접근지정자

Public : 모든 곳에서 접근 가능

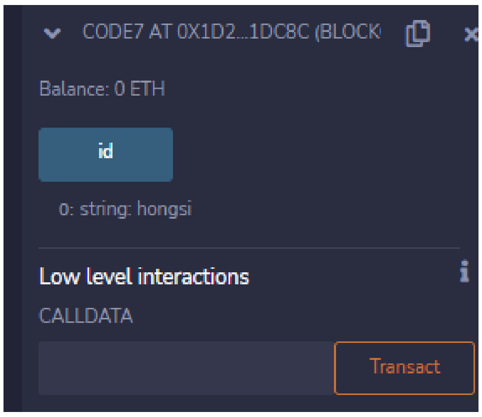
private : 자기 자신의 스마트컨트렉트

external : 외부에서 접근 가능, 자기자신 x

internal : 자기자신 가능 , 상속 자식도 가능



```
// SPDX-License-Identifier: GPL-3.0
pragma solidity >= 0.7.0 < 0.9.0;

contract code7 {
    //public
    string public id = "hongsi";

    //private
    string private pwd = "hongsi123";
}
```


CODE7 AT 0X1D2...1DC8C (BLOCK

Balance: 0 ETH

0: string: hongsi

Low level interactions

CALLDATA

public 은 보이지만 private은 안보인다.

```
// SPDX-License-Identifier: GPL-3.0
pragma solidity >= 0.7.0 < 0.9.0;

contract code7 {
    //public
    string public id = "hongsi";

    //private
    string private pwd = "hongsi123";

}

contract public_test {
    uint256 public att = 10;

    function setAtt(uint256 _att) public {
        att = _att;
    }

    function getAtt() view public returns(uint256){
        return att;
    }
}

contract private_test {
    uint256 private password = 1234;

    function setPass(uint256 _pass) public {
        password = _pass;
    }

    function getPass() view public returns (uint256){
        return password;
    }
}
```

솔리디티 접근지정자 중 Get함수에서 주로 쓰이는 접근지정자

view : function 밖의 변수들을 읽을 수 있으나 변경 불가

pure : function 밖의 변수들을 읽지 못하고, 변경도 불가

view 와 pure 명시 안할 때 : funtion 밖의 변수들을 읽어서 변경을 해야한다.

리빌 컨트렉트

public : 주소

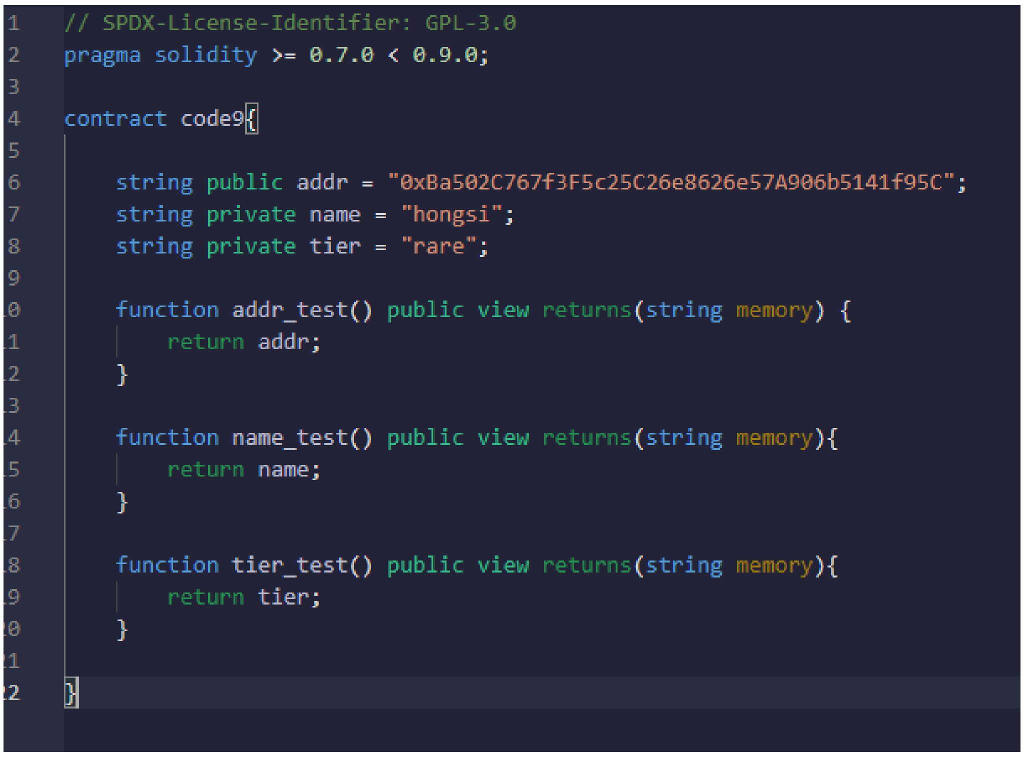
private : 캐릭터 이름

private : 등급

함수를 만들어서 리액트에서 dapp 만들어보기

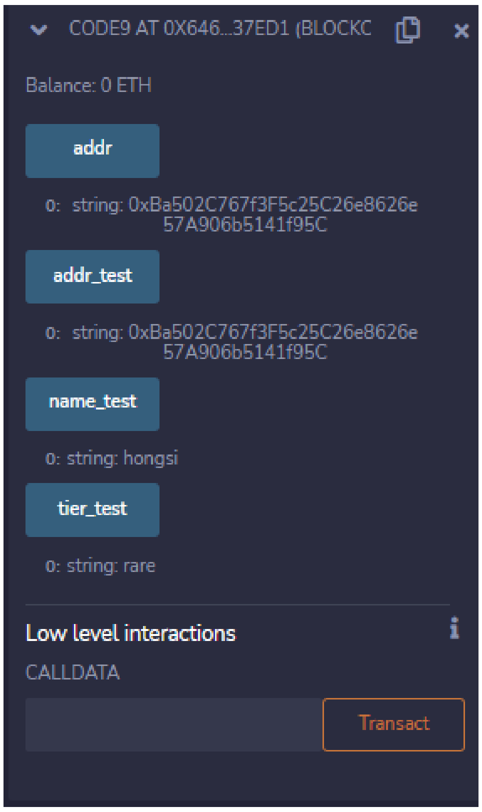
접근지정자 사용해보기

remix 코드



```solidity
// SPDX-License-Identifier: GPL-3.0
pragma solidity >= 0.7.0 < 0.9.0;

contract code9{

    string public addr = "0xBa502C767f3F5c25C26e8626e57A906b5141f95C";
    string private name = "hongsi";
    string private tier = "rare";

    function addr_test() public view returns(string memory) {
        return addr;
    }

    function name_test() public view returns(string memory){
        return name;
    }

    function tier_test() public view returns(string memory){
        return tier;
    }

}
```

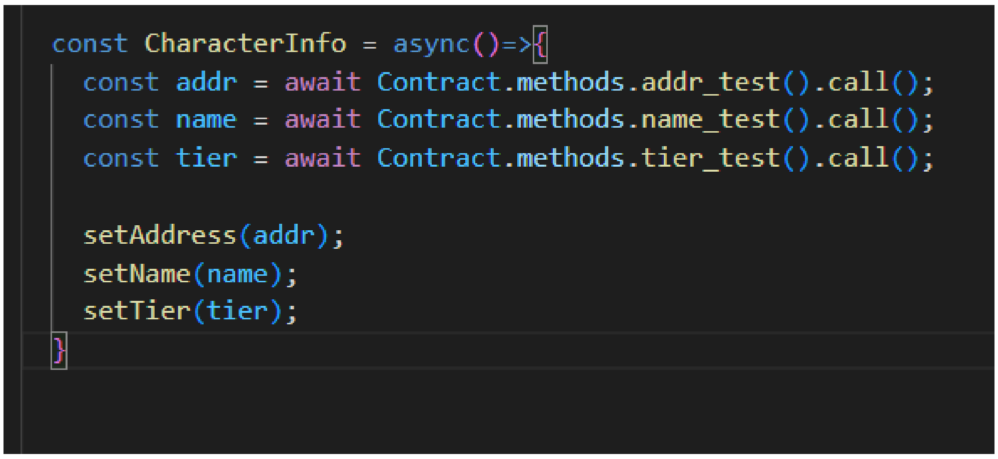
위 처럼 작성 시



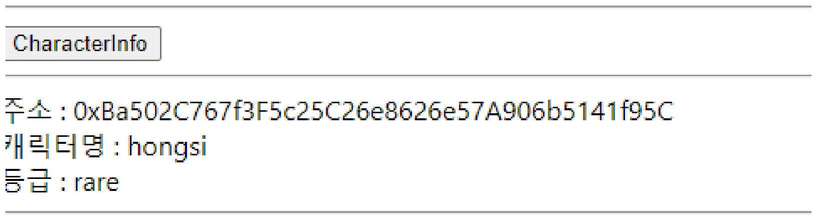
```
0: string: 0xBa502C767f3F5c25C26e8626e57A906b5141f95C
0: string: 0xBa502C767f3F5c25C26e8626e57A906b5141f95C
0: string: hongsi
0: string: rare
```

addr은 public으로 선언했으므로 addr은 보여지고

name과 tier는 private으로 선언했으므로 보여지지 않는다.



```
const CharacterInfo = async()=>{
  const addr = await Contract.methods.addr_test().call();
  const name = await Contract.methods.name_test().call();
  const tier = await Contract.methods.tier_test().call();

  setAddress(addr);
  setName(name);
  setTier(tier);
}
```



CharacterInfo

주소 : 0xBa502C767f3F5c25C26e8626e57A906b5141f95C
캐릭터명 : hongsi
등급 : rare

---

string

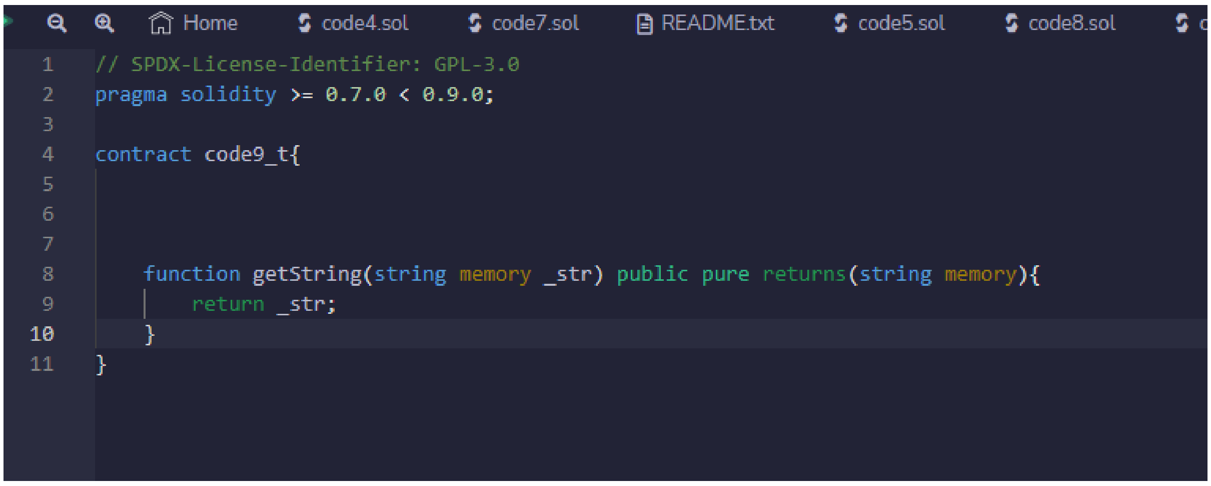
storage : 대부분의 변수 함수들을 저장, 영속적으로 저장되어 가스비가 비싸다.

memory : 함수의 파라미터, 리턴값, 레퍼런스 타입이 주로 저장

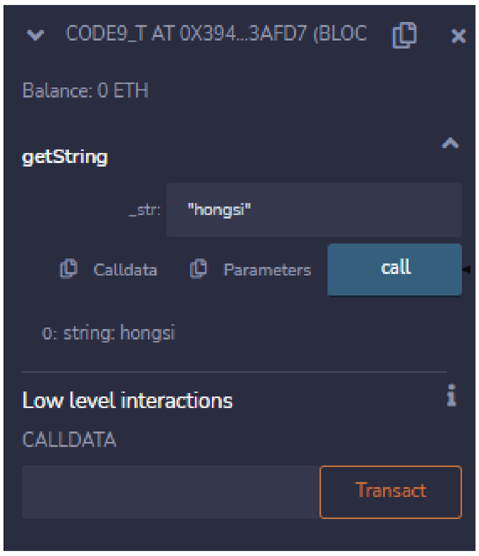
storage처럼 영속적이지 않고, 함수내에서만 유효하기 때문에 가스비용이 싸다.

colldata : 주로 external function의 파라미터에서 사용

stack : EVM stack data를 관리할때 쓰는 영역 1024mb

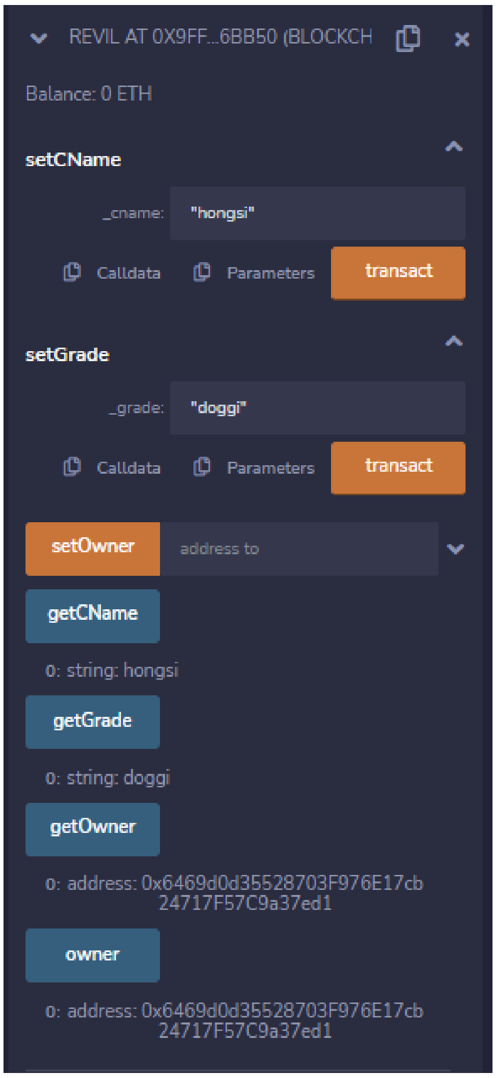


```
// SPDX-License-Identifier: GPL-3.0
pragma solidity >= 0.7.0 < 0.9.0;

contract code9_t{


    function getString(string memory _str) public pure returns(string memory){
        return _str;
    }
}
```



0: string: hongsi

```
// SPDX-License-Identifier: GPL-3.0
pragma solidity >= 0.7.0 < 0.9.0;

/*
    public : 주소
    private : 캐릭터이름, 등급
*/

contract revil{
    address public owner = 0x6469d0d35528703F976E17cb24717F57C9a37ed1;
    string private CharacterName = "?????";
    string private grade = "?????"; // C B A S SS

    function setOwner(address to) public {
        owner = to;
    }

    function getOwner() public view returns(address){
        return owner;
    }

    function setCName(string memory _cname) public{
        CharacterName = _cname;
    }

    function getCName() public view returns(string memory)
    {
        return CharacterName;
    }

    function setGrade(string memory _grade) public{
        grade = _grade;
    }

    function getGrade() public view returns(string memory)
    {
        return grade;
    }
}
```



"hongsi"

"doggi"

0: string: hongsi

0: string: doggi

0: address: 0x6469d0d35528703F976E17cb24717F57C9a37ed1

0: address: 0x6469d0d35528703F976E17cb24717F57C9a37ed1

이미지 바꾸기

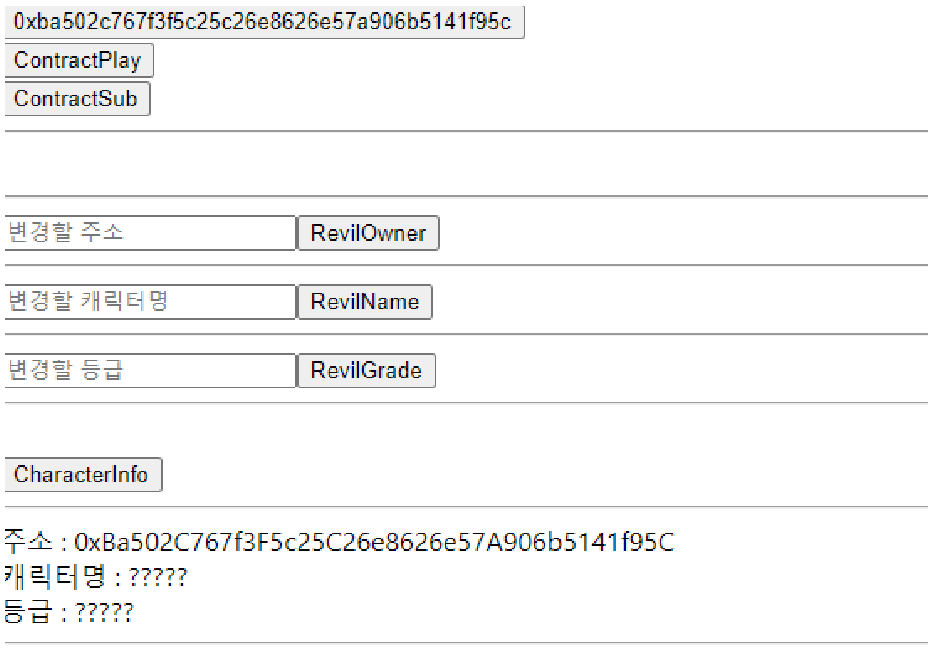
bool card

string image = “~”

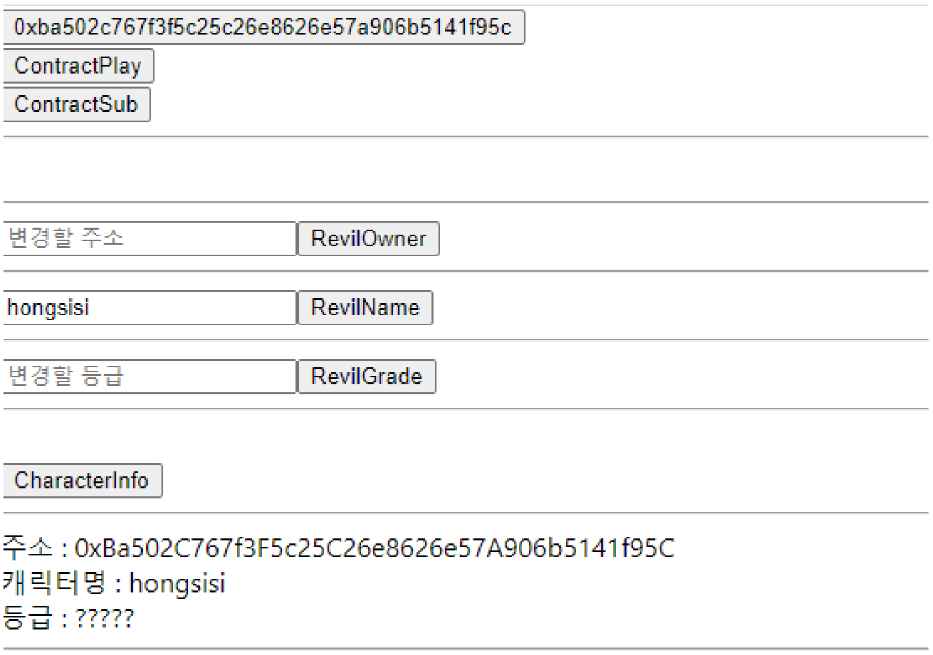
실습 code9 리액트로 변환

```
//remix code4
import Web3 from 'web3';
import React,{useState, useEffect} from 'react';
import './App.css';
import Abi from './Abi';

//컨트렉트 주소
const ContractAddress = '0x255c5d91F1E6417Ca92b6810cFC8073ae6a8F7D9';
```

```
function App() {
  const [web3, setWeb3] = useState();
  const [account, setAccount] = useState();
  const [Contract, setContract] = useState();
  const [pressStart, setPressStart] = useState();
  const [submessage, setSubmessage] = useState();
  const [inOwner, setInOwner] = useState();
  const [inName, setInName] = useState();
  const [inGrade, setInGrade] = useState();




  useEffect(()=> {
    if(typeof window.ethereum != "undefined") {
      try{
        const web = new Web3(window.ethereum);
        setWeb3(web);
      }catch(err){
        console.log(err);
      }
    }
  },[]);

  //메타마스크로부터 계정을 연결, 계정 주소를 저장
  const connectWallet = async()=>{
    const accounts = await window.ethereum.request({
      method : "eth_requestAccounts",
    })
    setAccount(accounts[0]);
    //컨트렉트
    const contract = await new web3.eth.Contract(Abi, ContractAddress);
    setContract(contract);
  }

  // 컨트렉트 실행 call get / set
  const ContractPlay = async()=>{
    // //컨트렉트
    // const Contract = await new web3.eth.Contract(Abi, ContractAddress);

    // 빨강 함수 가스비가 나가는 함수는 send로 호출
    await Contract.methods.setCName().send({from : account});
  }
  const ContractSub = async()=>{


    const message = await Contract.methods.Submessage().call();

    setSubmessage(message);
  }

  const RevilOwner = async()=> {
    await Contract.methods.setOwner(inOwner).send({from : account});

  }

  const RevilName = async()=> {
    await Contract.methods.setCName(inName).send({from : account});
  }

  const RevilGrade = async()=> {
    await Contract.methods.setGrade(inGrade).send({from : account});
  }





  const CharacterInfo = async()=>{
    const addr = await Contract.methods.getOwner().call();
    const name = await Contract.methods.getCName().call();
    const tier = await Contract.methods.getGrade().call();

    setInOwner(addr);
    setInName(name);
    setInGrade(tier);
  }
  return (
    <div>
      <button
        onClick={()=>{
          connectWallet();
          setPressStart(true);
        }}
      >
        {pressStart ? account : "Connect Wallet"}
      </button>
      <br/>
      <button onClick = {ContractPlay}>ContractPlay</button>
      <br/>
      <button onClick = {ContractSub}>ContractSub</button>
      <br/><hr/>
      {submessage}
      <br/><hr/>

      <input name="Owner" placeholder='변경할 주소' onChange={(e)=>setInOwner(e.target.value)}/>
      <button onClick={RevilOwner}>RevilOwner</button>
      <br/><hr/>

      <input name="Name" placeholder='변경할 캐릭터명' onChange={(e)=>setInName(e.target.value)}/>
      <button onClick={RevilName}>RevilName</button>
      <br/><hr/>

      <input name="Grade" placeholder='변경할 등급' onChange={(e)=>setInGrade(e.target.value)}/>
      <button onClick={RevilGrade}>RevilGrade</button>
      <br/><hr/>
      <br/>
      <button onClick = {CharacterInfo}>CharacterInfo</button>
      <br/><hr/>
      주소 : {inOwner}
      <br/>
      캐릭터명 : {inName}
      <br/>
      등급 : {inGrade}
      <hr/>
    </div>
  );
}

export default App;
```

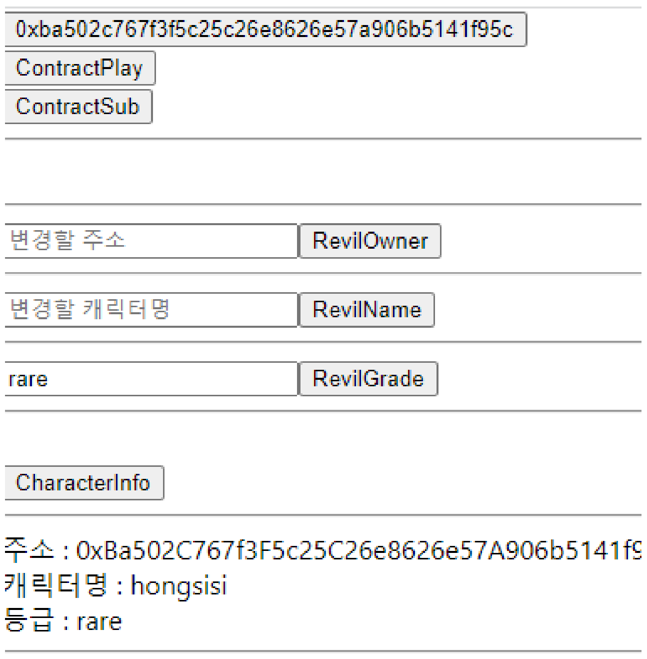


주소 : 0xBa502C767f3F5c25C26e8626e57A906b5141f95C
캐릭터명 : ?????
등급 : ?????



hongsisi

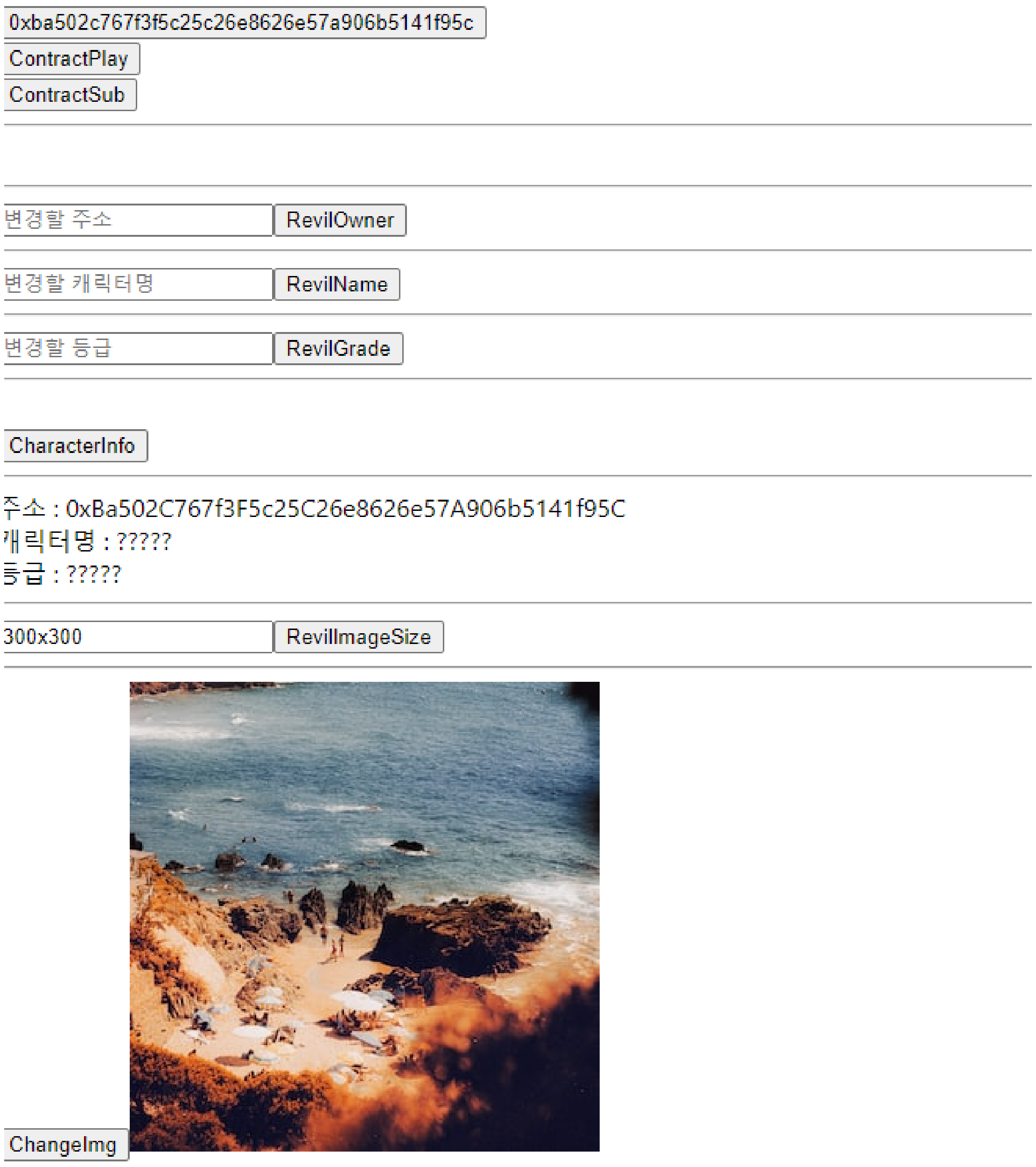
주소 : 0xBa502C767f3F5c25C26e8626e57A906b5141f95C
캐릭터명 : hongsisi
등급 : ?????



주소 : 0xBa502C767f3F5c25C26e8626e57A906b5141f9
캐릭터명 : hongsisi
등급 : rare

```
//remix code4
import Web3 from 'web3';
import React,{useState, useEffect} from 'react';
import './App.css';
import Abi from './Abi';

//컨트렉트 주소
const ContractAddress = '0x206EADCcE400b685E6954558D624F7bb36A4E203';

function App() {
  const [web3, setWeb3] = useState();
  const [account, setAccount] = useState();
  const [Contract, setContract] = useState();
  const [pressStart, setPressStart] = useState();
  const [submessage, setSubmessage] = useState();
  const [inOwner, setInOwner] = useState();
  const [inName, setInName] = useState();
  const [inGrade, setInGrade] = useState();
  const [inImg, setInImg] = useState();
  const [inImageSize, setInImageSize] = useState();




  useEffect(()=> {
    if(typeof window.ethereum != "undefined") {
      try{
        const web = new Web3(window.ethereum);
        setWeb3(web);
      }catch(err){
        console.log(err);
      }
    }
  },[]);

  //메타마스크로부터 계정을 연결, 계정 주소를 저장
  const connectWallet = async()=>{
    const accounts = await window.ethereum.request({
      method : "eth_requestAccounts",
    })
    setAccount(accounts[0]);
    //컨트렉트
    const contract = await new web3.eth.Contract(Abi, ContractAddress);
    setContract(contract);
  }

  // 컨트렉트 실행 call get / set
  const ContractPlay = async()=>{
    // //컨트렉트
    // const Contract = await new web3.eth.Contract(Abi, ContractAddress);

    // 빨강 함수 가스비가 나가는 함수는 send로 호출
    await Contract.methods.setCName().send({from : account});
  }
  const ContractSub = async()=>{


    const message = await Contract.methods.Submessage().call();

    setSubmessage(message);
  }

  const RevilOwner = async()=> {
    await Contract.methods.setOwner(inOwner).send({from : account});

  }

  const RevilName = async()=> {
    await Contract.methods.setCName(inName).send({from : account});
  }

  const RevilGrade = async()=> {
    await Contract.methods.setGrade(inGrade).send({from : account});
  }

  const RevilImageSize = async()=> {
    await Contract.methods.setImageSize(inImageSize).send({from : account});

  }
  const Image1 = "//source.unsplash.com/"+inImg;



  const ChangeImg =async()=>{
     const Img = await Contract.methods.getImageSize().call();

     setInImg(Img);
  }


  const CharacterInfo = async()=>{
    const addr = await Contract.methods.getOwner().call();
    const name = await Contract.methods.getCName().call();
    const tier = await Contract.methods.getGrade().call();

    setInOwner(addr);
    setInName(name);
    setInGrade(tier);
    console.log(Image1);
  }
  return (
    <div>
      <button
        onClick={()=>{
          connectWallet();
          setPressStart(true);
        }}
      >
```

```
        {pressStart ? account : "Connect Wallet"}
      </button>
      <br/>
      <button onClick = {ContractPlay}>ContractPlay</button>
      <br/>
      <button onClick = {ContractSub}>ContractSub</button>
      <br/><hr/>
      {submessage}
      <br/><hr/>

      <input name="Owner" placeholder='변경할 주소' onChange={(e)=>setInOwner(e.target.value)}/>
      <button onClick={RevilOwner}>RevilOwner</button>
      <br/><hr/>

      <input name="Name" placeholder='변경할 캐릭터명' onChange={(e)=>setInName(e.target.value)}/>
      <button onClick={RevilName}>RevilName</button>
      <br/><hr/>

      <input name="Grade" placeholder='변경할 등급' onChange={(e)=>setInGrade(e.target.value)}/>
      <button onClick={RevilGrade}>RevilGrade</button>
      <br/><hr/>
      <br/>
      <button onClick = {CharacterInfo}>CharacterInfo</button>
      <br/><hr/>
      주소 : {inOwner}
      <br/>
      캐릭터명 : {inName}
      <br/>
      등급 : {inGrade}
      <hr/>

      <input name="ImageSize" placeholder='변경할 이미지크기' onChange={(e)=>setInImageSize(e.target.value)}/>
      <button onClick={RevilImageSize}>RevilImageSize</button>
      <br/><hr/>
      <button onClick={ChangeImg}>ChangeImg</button>
      <img src={Image1} alt=""/>
    </div>
  );
}

export default App;
```



0xba502c767f3f5c25c26e8626e57a906b5141f95c
ContractPlay
ContractSub

변경할 주소 RevilOwner

변경할 캐릭터명 RevilName

변경할 등급 RevilGrade

CharacterInfo

주소 : 0xBa502C767f3F5c25C26e8626e57A906b5141f95C
캐릭터명 : ?????
등급 : ?????

300x300 RevilImageSize

ChangeImg