solidity-기본-8

태그	
날짜	@2023년 6월 22일

```
//SPDX-License-Identifier : GPL-3.0
pragma solidity >= 0.6.0 < 0.9.0;


SPDX-License-Identifier 목적
1. 라이센스를 명시해줌으로써 스마트컨트랙에 대한 신뢰감을 높일 수 있음
2. 스마트 컨트랙 소스코드가 워낙 오픈되어 있으니, 저작권과 같은 관련된 문제를 해소
```

---

payable, msg.value

```
Payable은 이더/토큰과 상호작용시 필요한 키워드라고 생각하면 간단하다.
send, transfer, call을 이용하여, 이더를 보낼 때ㅔ payable이라는 키워드가 필요하다
이 payable은 주로 함수, 주소, 생성자에 붙여서 사용된다.

msg.value는 송금보낸 코인의 값이다.

이더를 보내는 3가지

1. send : 2300 gas를 소비, 성공여부를 true 또는 false로 리턴
2. transfer : 2300 gas를 소비, 실패시 에러를 발생
3. call : 가변적인 gas 소비(gas값 지정 가능), 성공여부를 true 또는 false로 리턴
          재진임(reentrancy) 공격 위험성 있음, 2019년 12월 이후 call 사용 추천
```

```
contract lec{
      event howMuch(uint256 _value);

//_to는 smart contract주소도 가능하다. 즉, smart contract도 이더를 받을 수 있다.
      function sendNow(address payable _to) public payable{
        bool sent = _to.send(msg.value); // return true or false
```

```
        require(send, "Failed to send ether");
        emit howMuch(msg.value);
      }

      function transferNow(address payable _to) public payable{
        _to.transfer(msg.value);
        emit howMuch(msg.value);
      }

      function callNow(address payable _to) public payable{
        // ~0.7
        // (bool sent, ) = _to.call.gass(1000).value(msg.value)("");
        // require(sent, "Failed to send ether");

        // 0.7~
        (bool sent, ) = _to.call{value : msg.value , gas:1000} ("");
        emit howMuch(msg.value);
      }

}
```

```
// payable을 생성자에 넣을 때
// 특정 주소에게만 권한 주기

contract MobileBanking{

    address owner;

    // constructor는 smart contract가 배포 될 때 가장 먼저 실행되는 것
    constructor() payable{
        owner = msg.sender;
    }
    event SendInfo(address _msgSender, uint256 _currentValue);
    event MyCurrentValue(address _msgSender, uint256 _value);
    event CurrentValueOfSomeone(address _msgSender, address _to, uint256 _value);

    function sendEther(address payable _to) public payable{
        require(msg.sender == owner, "Only Owner!") // owner만 이 함수를 사용할 수 있다.
        require(msg.sender.balance >= msg.value, "Your balance is not enough");
        _to.transfer(msg.value);
        emit SendInfo(msg.sender, (msg.sender).balance);
      }
}
```

balance와 msg.sender

```
주소.balance :
해당 특정 주소의 현재 가지고 있는 이더의 잔액을 나타낸다. (msg.value는 송금액)

msg.sender : 스마트 컨트랙을 사용하는 주체라고 볼 수 있다.

contract MobileBanking{

      event SendInfo(address _msgSender, uint256 _currentValue);
      event MyCurrentValue(address _msgSender, uint256 _value);
      event CurrentValueOfSomeone(address _msgSender, address _to, uint256 _value);

      function sendEther(address payable _to) public payable{
          require(msg.sender.balance >= msg.value, "Your balance is not enough");
          _to.transfer(msg.value);
          emit SendInfo(msg.sender, (msg.sender).balance);
      }

      function checkValueNow() public{
          emit MyCurrentValue(msg.sender, msg.sender.balance);
      }

      function checkUserMoney(address _to) public {
          emit CurrentValueOfSomeone(msg.sender, _to, _to.balance);
      }
}
```