solidity-기본-4

태그	
날짜	@2023년 6월 20일

mapping

```
contract lec17{
  mapping(uint256 => uint256) private ageList;
  mapping(string => uint256) private priceList;
  mapping(uint256 => string) private nameList;

  function setAgeList(uint256 _index, uint256 _age) public{
    ageList[_index] = _age;
  }

  function getAge(uint256 _index) public view returns(uint256){
    return ageList[_index];
  }

  function setNameList(uint256 _index, string memory _name) public{
    ageList[_index] = _name;
  }

  function getName(uint256 _index) public view returns(string memory){
    return nameList[_index];
  }

  function setPriceList(string memory _itemName, uint256 _price) public{
    priceList[_itemName] = _price;
  }

  function getPriceList(string memory _index) public view returns(uint256){
    return priceList[_index];
  }
}
```

배열 array

```
contract lec18{
    uint256[] public ageArray; // 사이즈가 제한되어있지 않음
    uint256[10] public ageFixedSizeArray; // 사이즈가 10으로 제한된 배열
    string[] public nameArray = ["kal","john","Kerri"];

    function AgeLength() public view returns(uint256){
      return ageArray.length;
    }

    function AgePush(uint256 _age) public{
      ageArray.push(_age);
    }

    function AgeGet(uint256 _index) public view returns(uint256){
      return ageArray[_index];
    }

    function AgePop(uint256 _index) public {
      ageArray.pop(); // 제일 최신의 값을 삭제하게 된다.
    }

    function AgeDelete(uint256 _index) public{
      delete ageArray[_index];
    // 완전히 지워지지 않는게 아니라 삭제된 index자리를 0으로 채운다.
    // length의 길이는 그대로.
    }

    function AgeChange(uint256 _index, uint256 _age) public {
        ageArray[_index] = _age;
    // 만약 delete로 값이 0인 index의 값을 변경할 때 사용
    }

}
```

---

array와 mapping의 주의사항

```
contract lec19{
  uint256 num = 89;
  mapping(uint256 => uint256) numMap;
  uint256[] numArray;

  function changeNum(uint256 _num) public{
    num = _num;
  }
```

```
  function showNum() public view returns(uint256) {
    return num;
  }

  function numMapAdd() public{
    numMap[0] = num;
  }

  function showNumMap() public view returns(uint256) {
    return numMap[0];
  }

  function numArrayAdd() public{
      numArray.push(num);
  }

  function showNumArray() public view returns(uint256) {
    return numArray[0];
  }

  function updataArray() public{
    numArray[0] = num;
  }

}
```

---

struct (구조체)

: 나만의 타입을 만드는것

```
contract lec20{
  struct Character{
    uint256 age;
    string name;
    string job;
  }

  mapping(uint256 => Character) public CharacterMapping;
  Character[] public CharacterArray;

  function createCharacter(uint256 _age, string memory _name, string memory _job)
           pure public returns(Character memory) {
    return Character(_age, _name, _job);
  }

  function getCharacterMapping(uint256 _key) public view returns(Character memory){
```

```
    return CharacterMapping[_key];
  }

  function createCharacterArray(uint256 _age, string memory _name, string memory _job) public{
    CharacterArray.push(Character(_age,_name,_job));
  }

  function getCharacterArray(uint256 _index) public view returns(Character memory){
    return CharacterMapping[_index];
  }
}
```