(디지털컨버전스)스마트 콘텐츠와 웹 융합응용SW개발자 양성과정

강사 - Innova Lee(이상훈)
gcccompil3r@gmail.com
학생 –Joongyeon Kim(김중연)
jjjr69@naver.com

2021년 5월 27일 질문노트

[김중연]

```java
import java.util.Arrays;

// 최종 결론 +(많이 헷갈린다)
// 게터고 나발이고 다 떠나서
// 객체는 메모리 자체를 전달하며
// 값은 메모리가 아닌 값을 전달한다.
class CloneMemory {
    int[] arr;
    int num;

    public CloneMemory () {
        arr = new int[3]; //배열 객체를 생성한다
        num = 3;

        for (int i = 0; i < 3; i++) {
            arr[i] = (int)(Math.random() * 6 + 1);
        }
    }

    public void reRandArr () {
        for (int i = 0; i < 3; i++) {
            arr[i] = (int)(Math.random() * 6 + 1);
        }
    }

    public void reNum () { num = 7; }

    // 게터는 모두 클론임을 어떻게 증명할까 ?
    // 일단 게터로 배열값을 리턴받아 어딘가에 저장한다.
    // 다음에 매서드를 사용해서 객체의 배열의 값을 변경한다.
    // 리턴받아 저장했던 배열을 출력했을때 결과가 같으면 객체가 전달된 것이고
    // 만약 다른 결과가 도출된다면 복제되었음을 알 수 있다.
    // 결론: 결국 객체에 대한 리턴이므로 메모리 정보가 전달됨
    public int[] getCloneArr () { return arr; }

    public int getCloneVariable () { return num; }

    public String toString () {   // 배열의 인덱스마다 랜덤값이 들어간다
        return "arr[0] = " + arr[0] +
                ", arr[1] = " + arr[1] +
                ", arr[2] = " + arr[2];
    }
}

public class MemoryCloneTest {
    public static void main(String[] args) {
        CloneMemory cm = new CloneMemory();

        System.out.println(cm);  //객체의 들어있는 메소드들의 결과를 출력한다

        int[] save = cm.getCloneArr();   //객체의 값(복제)을 리턴받아서 save에 대입한다

        System.out.printf("save[0] = %d, save[1] = %d, save[2] = %d\n",
                save[0], save[1], save[2]);  //기존 arr과 같은 결과가 나온다 (배열 전달시 인덱스 지정하면 값이 전달된다)

        cm.reRandArr();  //cm.reRandArr()를 호출함

        System.out.println("객체에 접근해 출력");
        System.out.println(cm);  //메모리에 올라가있는 객체 자체를 불러온다

        System.out.println("사전 저장 정보 출력");
        System.out.printf("save[0] = %d, save[1] = %d, save[2] = %d\n",
```

```java
// 결론: 자바에서 객체에 대한 접근은 모두 메모리를 제어하는 방식이 된다.

int num = cm.getCloneVariable();  //게터에서 3을 불러온다

System.out.println("객체내 변수값 획득: " + num); //게터에 값을 받아 3이출력된다

cm.reNum();  //객체안에 있는 객체를 불러온다 (객체내에 들어있는 객체를 요청하면 원본을 줌)

System.out.println("변경 후 사전 획득한 정보 재출력: " + num); //3이 출력된다(값)

// 결론: 앞서서도 확인했지만 값에 대해서는 복제가 이루어짐을 확인할 수 있다.
System.out.println("변경 정보 파악: " + cm.getCloneVariable()); // reNum객체가 전달되었다(처음 것이 아닌 복제된 것을 불러옴)
```

1. 객체는 메모리 자체를 전달
2. 값은 메모리가 아닌 값을 전달한다
3. 게터는 모두 객체를 복제한 것 (값)

```java
public class ArrayListTest {
    public static void main(String[] args) {
        // 용도: 일종의 배열임
        //       배열의 사이즈를 지정하고 사용해야 하지만
        //       이 녀석은 넣고 싶은대로 아무때나 막 넣어도 된다.
        //       (참고로 이 녀석도 Heap을 이용한 동적할당을 수행함)

        // 사용법: ArrayList<내부에저장할데이터타입> 변수명 = new ArrayList<내부에저장할데이터타입>();

        // 일반 배열과의 차이점은 ?
        // 배열은 메모리가 연속적으로 배치된다.
        // 이 녀석은 불연속 배치다.
        // 어떻게 ?
        // | 데이터1 | 다음링크 | ---> | 데이터2 | 다음링크 | ---> | 데이터3 | 다음링크 | ---> ....
        // 배열은 ?
        // | 데이터1 | 데이터2 | 데이터3 | 데이터4 | 데이터5 | 데이터6 | 데이터7 | ...
        ArrayList<String> lists = new ArrayList<String>();

        lists.add("빵");
        lists.add("버터");
        lists.add("우유");
        lists.add("계란");
        lists.add("쥬스");
        lists.add("베이컨");
        lists.add("파스타");
        lists.add("비프샐러드");
        lists.add("피자");

        for (String list : lists) {  // for : each 문을 사용하여 ArrayList문에 있는 변수값들을 순서대로 전부 대입함
            System.out.println("현재 항목은 = " + list);
        }
```

```java
import java.util.ArrayList;
import java.util.Scanner;

class Shop {
    ArrayList<String> lists;
    Scanner scan;

    public Shop () {
        lists = new ArrayList<String>();
        scan = new Scanner(System.in);
    }

    public void deliveryCome () {
        System.out.print("필요한 물품을 말씀하세요: ");
        lists.add(scan.nextLine());
    }

    public void cancelOrder () {
        System.out.print("취소할 물품을 말씀하세요: ");
        lists.remove(scan.nextLine());
    }

    // toString 으로 자동 완성 가능
    // 객체 정보 출력에 사용합니다.
    // 아직 인터페이스 배우지 않았으므로 설명은 향후 진행
    @Override
    public String toString() {
        return "Shop{" +
                "lists=" + lists +
                '}';
    }
}

public class ShopTest {
    public static void main(String[] args) {
        Shop s = new Shop();

        for(int i = 0; i < 3; i++) {
            s.deliveryCome();
        }

        s.cancelOrder();

        // 아래와 같이 객체를 전달하면 toString이 호출됨
        System.out.println(s);
    }
}

// ArrayList는 Queue 혹은 Stack 역할을 할 수 있는데 기본이 Queue(큐) 역할
// 맨 처음 넣은 정보가 가장 앞에 배치되고
// 두 번째 넣은 정보가 두 번째에 배치되고 ...
```

```java
import java.util.ArrayList;
import java.util.Arrays;

class Roulette {                            //변수들을 설정해준다
    ArrayList<String> nameLists;
    String[] tmpArr;
    int[] tmpIdx;

    int[] success;

    int nameLength;
    Boolean isRedundant;

    public Roulette (String[] names) {   //변수값을 초기화 해준다
        nameLength = names.length;
        isRedundant = true;

        nameLists = new ArrayList<String>();
        tmpArr = new String[nameLength];  //이름의 문자열을 tmpArr에 대입한다
        tmpIdx = new int[nameLength];   //이름의 갯수를 tmpIdx에 대입한다

        success = new int[3];

        int i = 0;

        for (String name : names) { // names에 들어있는 이름들을 String name에 넣는다
            tmpArr[i++] = name;
        }
    }

    private Boolean checkDuplicate (int idx) {  //shuffle메소드와 checkSuccess메소드
        for (int i = 0; i < idx; i++) {   // 값의 중복을 배제하기 위해만든 함수
            if (tmpIdx[i] == tmpIdx[idx]) {  //tmpIdx[i] == tmpIdx[idx] 값이 같으면 다시 반복
                return true;
            }
        }

        return false;
    }

    public void shuffle () {  // i에 27가지의 무작위 수가 대입된다, 나오는 숫자를 섞기위한 함수
        int i = 0;

        isRedundant = true;

        do {
            tmpIdx[i] = (int)(Math.random() * nameLength);

            if (checkDuplicate(i)) {  //이 checkDuplicate(i)의 0~26의 무작위 수가 들어간다??
                continue;  // 해당 반복부분 탈출 후 반복실행(do로 돌아감)
            }

            i++;

            if (i == nameLength) {
                isRedundant = false;
            }
        } while (isRedundant);
    }
```

```java
public void checkSuccess () {  //당첨번호를 확인하기위한 메소드
    int i = 0;

    isRedundant = true;

    do {
        success[i] = (int)(Math.random() * nameLength);  //0~26룰렛을 돌린다

        if (checkDuplicate(i)) {
            continue;
        }

        i++;

        if (i == 3) {
            isRedundant = false;
        }
    } while (isRedundant);
}

public void printSuccessArr () {  // 0~26중 무작위 3가지 수 가 출력된다
    for (int i = 0; i < 3; i++) {
        System.out.printf("success[%d] = %d\n", i, success[i]);
    }
}
```

```java
    @Override
    public String toString() {  //Roulette객체의 tmpIdx를 출력한다???
        return "Roulette{" +
                "tmpIdx=" + Arrays.toString(tmpIdx) +
                '}';
    }
}
```

```java
ublic class Prob48 {
    public static void main(String[] args) {
        String[] names = {
                "박세진", "김창욱", "김민규", "김중연", "문성호",
                "강병화", "최승현", "유종현", "한상우", "전승리",
                "이경환", "최준환", "김원석", "여인준", "이태양",
                "김윤영", "정도영", "황정아", "임초롱", "김남교",
                "이주형", "김도연", "최혜주", "김도혜", "고재권",
                "임익환", "안보미", "이상훈"
        };

        Roulette r = new Roulette(names);

        System.out.println(r);

        r.shuffle();  //

        System.out.println(r);

        r.checkSuccess();
        r.printSuccessArr();
```