KB금융그룹 | 국민의 평생 금융파트너

2025년 상반기 K-디지털 트레이닝

Composite - 그릇과 내용물을 동일시한다

[KB] IT's Your Life

KB 국민은행

Composite 패턴

디렉토리/파일

- 디렉토리와 파일은 서로 다르지만, 둘 다 디렉토리 안에 넣을 수 있음
- 디렉토리와 파일을 디렉토리 엔트리라고 부름
- 디렉토리 엔트리는 디렉토리와 파일을 같은 종류로 간주(동일시 함)

→ 그릇과 내용물을 같은 종류로 취급
그릇은 또다른 그릇을 포함할 수 있음 – 중첩/재귀적인 구조를 가짐

같은 타입이어야 담을수있죠
부모타입이 있어야하고 부모타입으로 참조하면 ㄱㄴ

Composite 패턴

- 그릇과 내용물을 동일시하여 재귀적인 구조를 만드는 디자인 패턴

예시
메뉴

메뉴 아이템으로 서브메뉴

서브메뉴 아이템으로 서브메뉴...

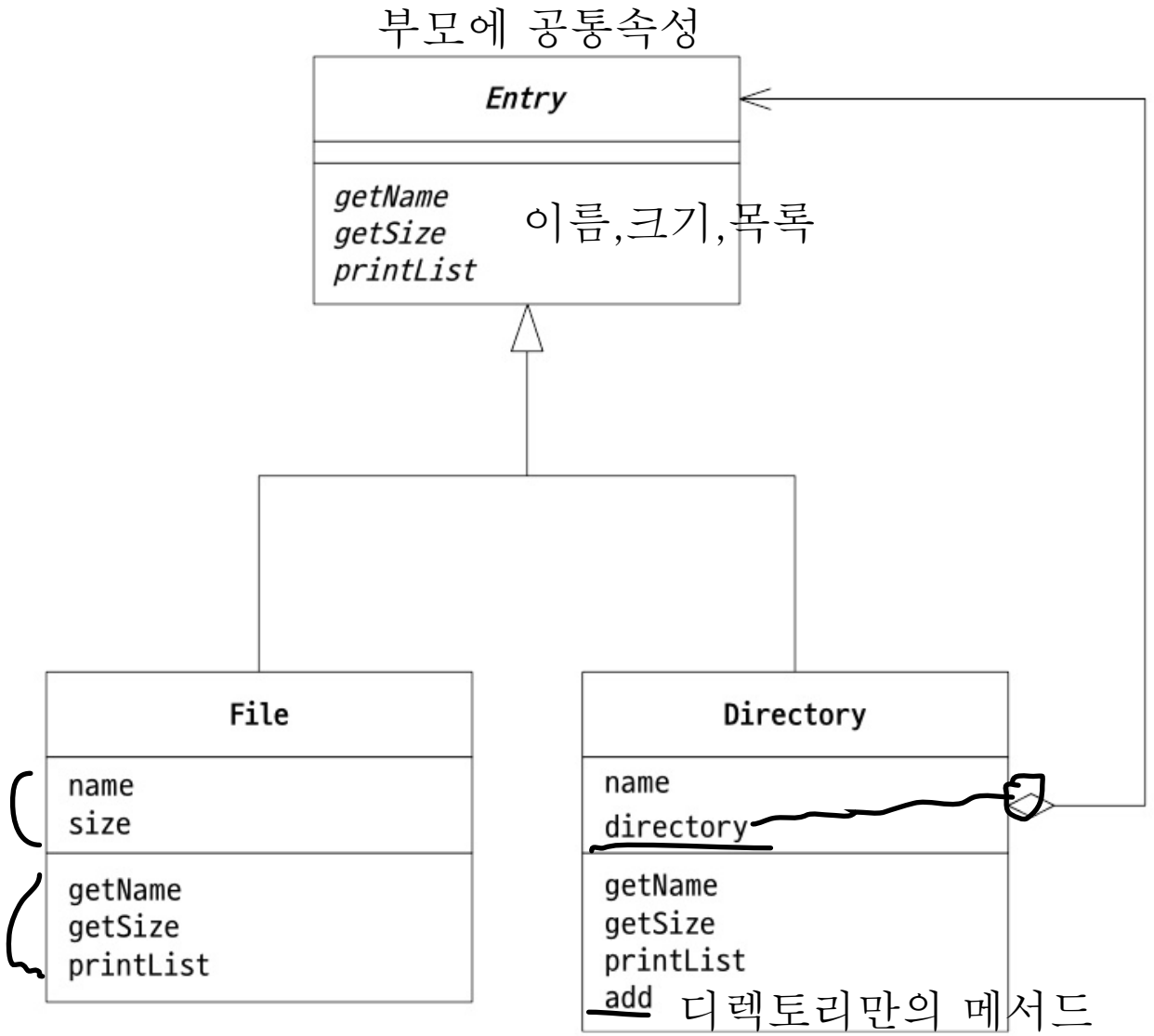
예제 프로그램

- 파일과 디렉토리를 도식적으로 표현하는 프로그램

이름	설명
Entry 부모 타입	File과 Directory를 동일시하는 추상 클래스
File	파일을 나타내는 클래스
Directory	디렉터리를 나타내는 클래스
Main	동작 테스트용 클래스

예제 프로그램

예제 프로그램의 클래스 다이어그램



Entry.java

일단 먼저 부모 클래스

추상클래스

```
public abstract class Entry {
    public abstract String getName();

    public abstract int getSize();

    public void printList() {
        printList("");
    }

    // prefix를 앞에 붙여서 목록을 표시
    protected abstract void printList(String prefix);

    // 문자열 표시
    @Override
    public String toString() {
        return getName() + "(" + getSize() + ")";
    }

}
```

예제 프로그램

File.java

파일부터.

```
public class File extends Entry {
    private String name;
    private int size;

    public File(String name, int size) {
        this.name = name;
        this.size = size;
    }

    @Override
    public String getName() {
        return name;
    }

    @Override
    public int getSize() {
        return size;
    }

    @Override
    public void printList(String prefix) {
        System.out.println(prefix + "/" + this);
    }
}
```

자신의 경로 출력

Directory.java

```
public class Directory extends Entry {
    private String name;
    private List<Entry> directory = new ArrayList<>();

    public Directory(String name) {
        this.name = name;
    }

    @Override
    public String getName() {
        return name;
    }
```

파일,디렉토리 담을수있어요

Directory.java

```
    @Override
    public int getSize() {
        int size = 0;
        for (Entry entry : directory) {
            size += entry.getSize();
        }
        return size;
    }

    @Override
    protected void printList(String prefix) {
        System.out.println(prefix + "/" + this);
        for(Entry entry: directory) {
            entry.printList(prefix + "/" + name);
        }
    }

    // 디렉터리 엔트리를 디렉터리에 추가한다.
    public Entry add(Entry entry) {
        directory.add(entry);
        return this;
    }
}
```

재귀적인 호출이죠??

엔트리가 디렉토리라면 재귀호출
아니면 그냥 호출됨

이것도 대상에 따라 재귀호출이 될 수 있음.
디렉토리면 재귀호출이죠

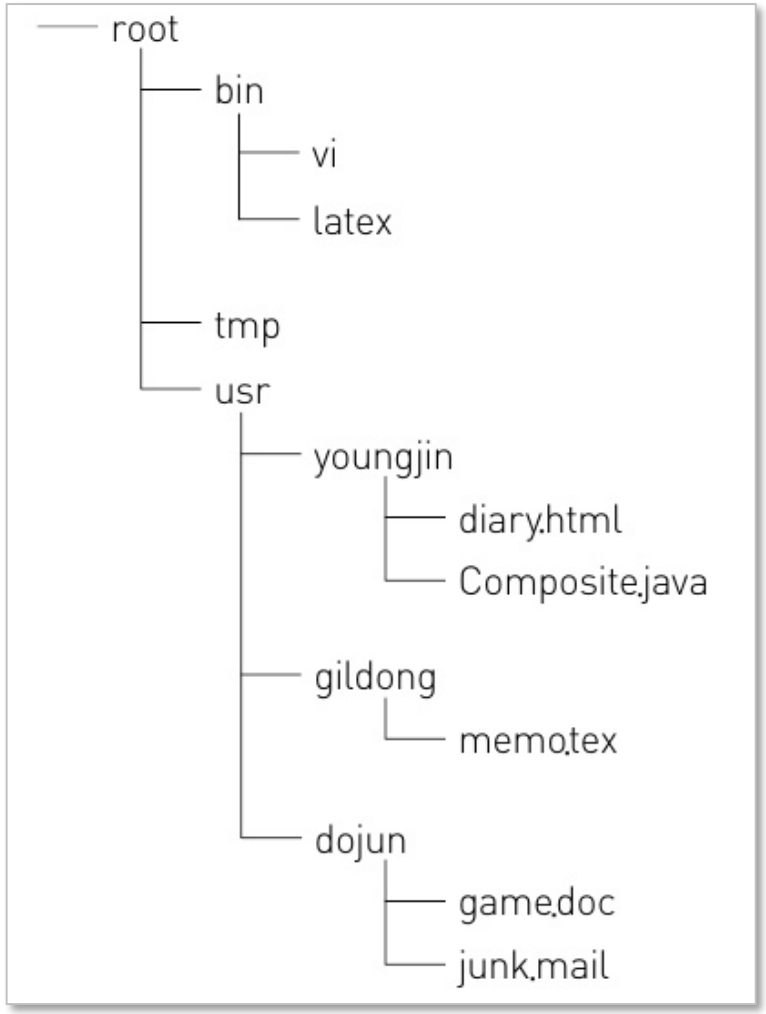
그릇 기능 엮할 메소드

예제 프로그램

Main 클래스

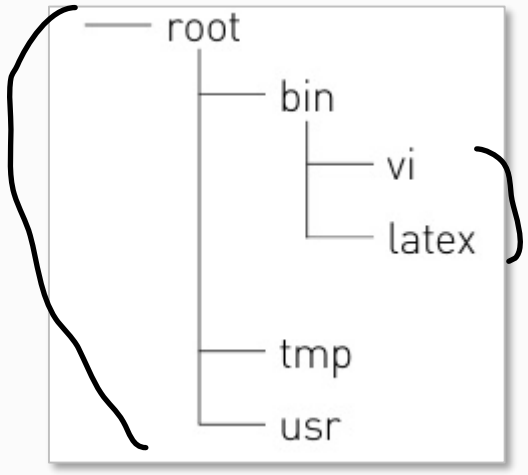
- 테스트 디렉토리 계층

가정사항



Main.java

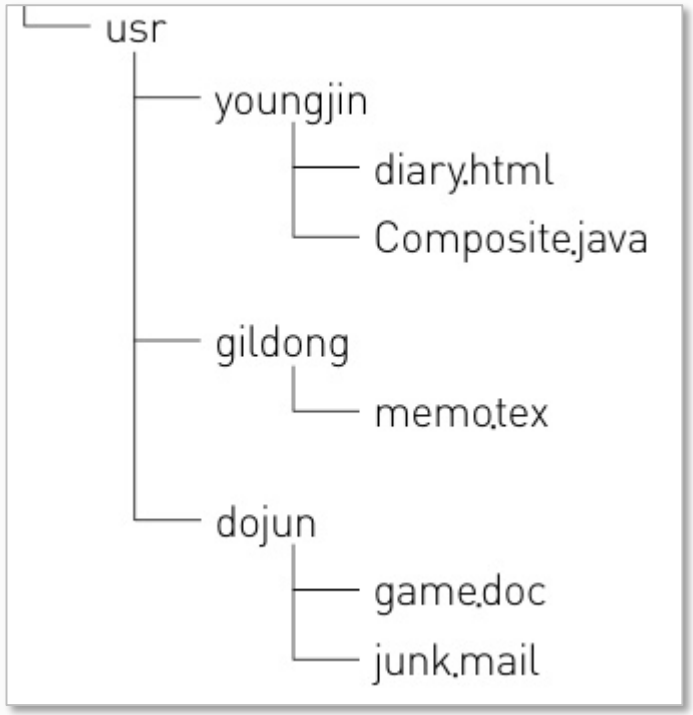
```
public class Main {
    public static void main(String[] args) {

        System.out.println("Making root entries...");
        Directory rootdir = new Directory("root");

        Directory bindir = new Directory("bin");
        Directory tmpdir = new Directory("tmp");
        Directory usrdir = new Directory("usr");

        rootdir.add(bindir);
        rootdir.add(tmpdir);
        rootdir.add(usrdir);

        bindir.add(new File("vi", 10000));
        bindir.add(new File("latex", 20000));

        rootdir.printList();
        System.out.println();
```



예제 프로그램

Main.java

```
        System.out.println("Making user entries...");
        Directory youngjin = new Directory("youngjin");
        Directory gildong = new Directory("gildong");
        Directory dojun = new Directory("dojun");

        usrdir.add(youngjin);
        usrdir.add(gildong);
        usrdir.add(dojun);

        youngjin.add(new File("diary.html", 100));
        youngjin.add(new File("Composite.java", 200));

        gildong.add(new File("memo.tex", 300));

        dojun.add(new File("game.doc", 400));
        dojun.add(new File("junk.mail", 500));

        rootdir.printList();
    }
}
```

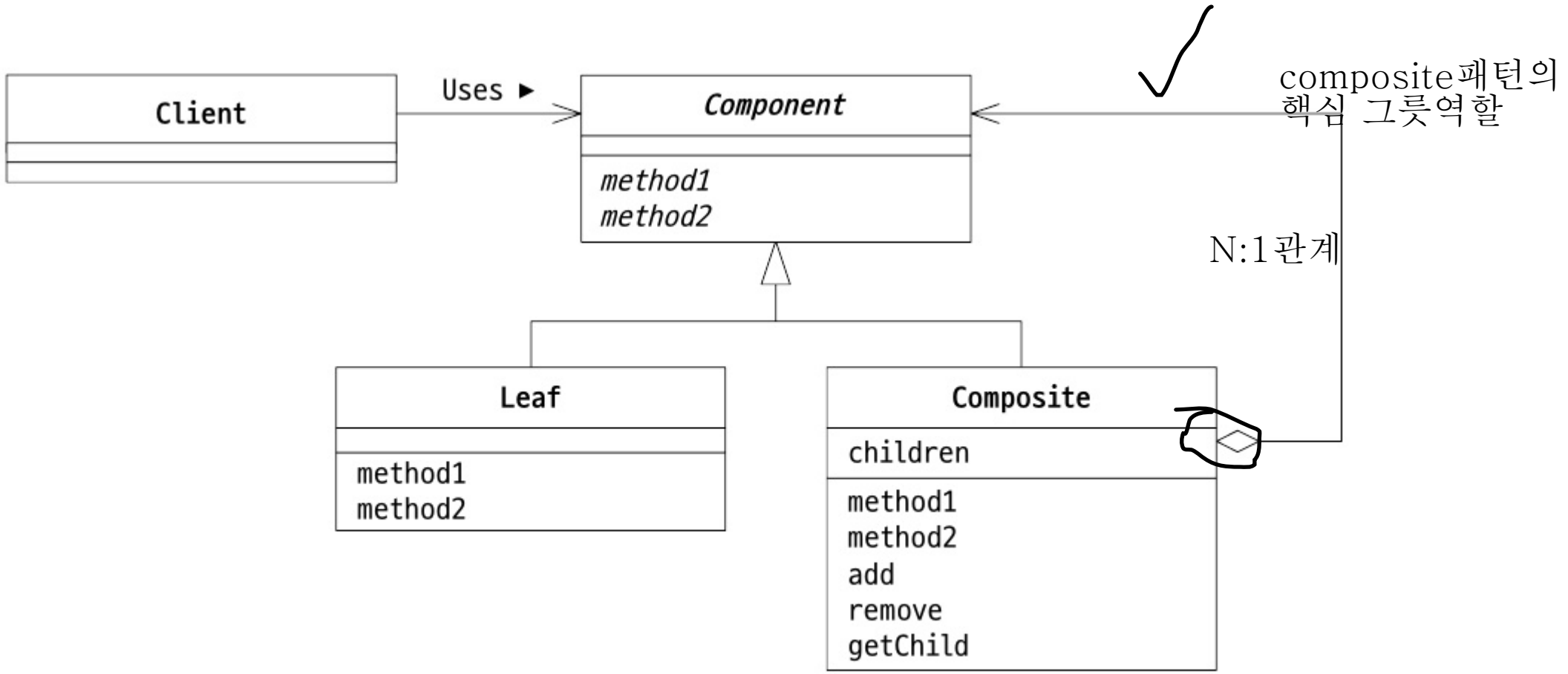


예제 프로그램

```
Making root entries...
/root(30000)
/root/bin(30000)
/root/bin/vi(10000)
/root/bin/latex(20000)
/root/tmp(0)
/root/usr(0)

Making user entries...
/root(31500)
/root/bin(30000)
/root/bin/vi(10000)
/root/bin/latex(20000)
/root/tmp(0)
/root/usr(1500)
/root/usr/youngjin(300)
/root/usr/youngjin/diary.html(100)
/root/usr/youngjin/Composite.java(200)
/root/usr/gildong(300)
/root/usr/gildong/memo.tex(300)
/root/usr/dojun(900)
/root/usr/dojun/game.doc(400)
/root/usr/dojun/junk.mail(500)
```

Composite 패턴

Composite 패턴 클래스 다이어그램



Composite 패턴

- 복수와 단수를 동일시 하기
- add는 복수 측에
- 재귀적 구조를 가지며, 탐색 시 재귀 호출 사용

보통 그릇에서 재귀적 구조가 나타남