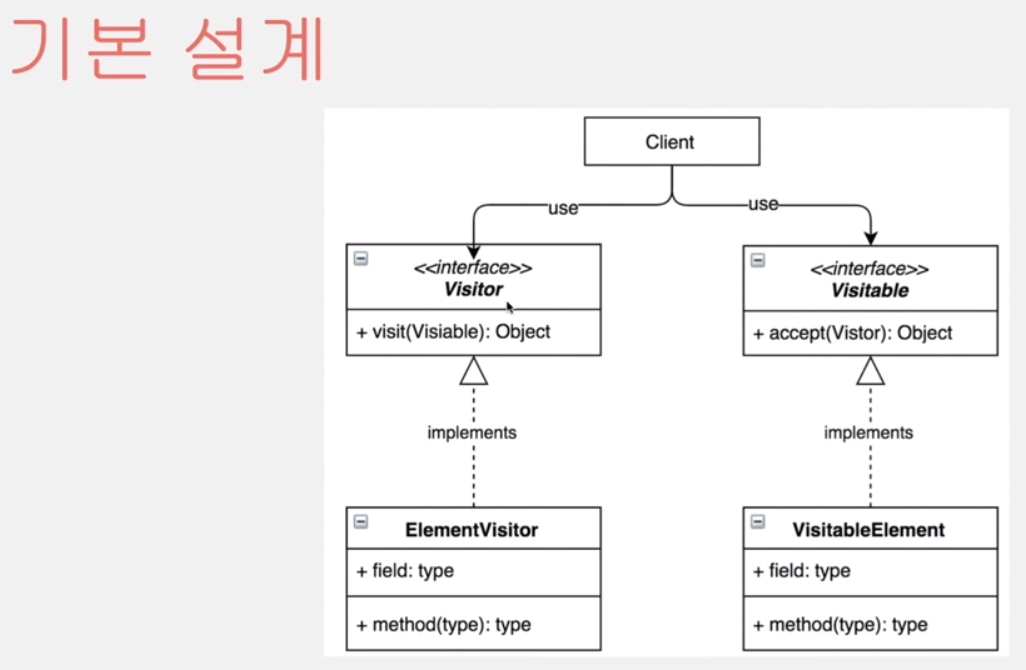
VisitorPattern

객체에서 처리를 분리해서 사용할 수 있다.

객체class), 처리(method) ,분리



소스코드

Visitable 이라는 소스코드를 수정할수 없을대

Visitor 라는 클래스를만들어서 응용 가능

package designpattern;

import java.util.ArrayList;

public class VisitorPattern {

public static void main(String[] args) {

VisitorA v1 = new VisitorA();

ArrayList<Visitoable> visitables = new ArrayList<Visitoable>();

visitables.add(new VisitoableA(1));

visitables.add(new VisitoableA(2));

visitables.add(new VisitoableA(3));

visitables.add(new VisitoableA(4));

visitables.add(new VisitoableA(5));

Visitor visitor = new VisitorA();

int ageSum=0;

for (Visitoable visitable : visitables) {

visitable.accept(visitor);

ageSum += ((VisitoableA) visitable).getAge();

}

System.out.println(ageSum);

System.out.println(((VisitorA) visitor).getAgeSum());

}

}

interface Visitor {

public void visit(Visitoable visitoable);

}

interface Visitoable {

public void accept(Visitor visitor);

}

class VisitoableA implements Visitoable {

private int age;

@Override

public void accept(Visitor visitor) {

visitor.visit(this);

}

public VisitoableA(int age) {

super();

this.age = age;

}

public int getAge() {

return age;

}

public void setAge(int age) {

this.age = age;

}

}

class VisitorA implements Visitor {

private int ageSum;

public VisitorA() {

this.ageSum = 0;

}

@Override

public void visit(Visitoable visitoable) {

if (visitoable instanceof VisitoableA) {

ageSum += ((VisitoableA) visitoable).getAge();

} else {

// ...

}

}

public int getAgeSum() {

return ageSum;

}

}