package designpattern;

public class BridgePattern {

public static void main(String[] args) {

// 어댑터 패턴과 브릿지 패턴을 연결하여 이해 기는부분과 구현 부분을 분리 (어댑터패턴과흡사)

//모스부호 를 구현

PrintMorseCode code = new PrintMorseCode(new DefaultMSF());

code.g().a().r().a().m();

}

}

//필요할대마다 기능만넣어주는 클래스 생성 기계에 넣어주는 코드 등

class DefaultMSF implements MorseCodeFunction{

@Override

public void dot() {

// TODO Auto-generated method stub

System.out.print(".");

}

@Override

public void dash() {

// TODO Auto-generated method stub

System.out.print("-");

}

@Override

public void space() {

// TODO Auto-generated method stub

System.out.println(" ");

}

}

interface MorseCodeFunction{

public void dot();

public void dash();

public void space();

}

class MorseCode{ //수정시마다 이코드로와서 구현을 바꿔줘야함

private MorseCodeFunction function;

public void dot(){

function.dot();

// System.out.print(".");

// System.call.Bip(); 구현을 바꿀대마다 바꿔야하기때문에 힘듬 그래서 인터페이스필요

}

public void dash(){

//델리게이트패턴 다른대로 위임함

function.dash();

//System.out.print("-");

// System.call.longBip();

}

public void space(){

function.space();

// System.out.print(" ");

// System.call.longTip();

}

public MorseCodeFunction getFunction() {

return function;

}

public void setFunction(MorseCodeFunction function) {

this.function = function;

}

}

class PrintMorseCode extends MorseCode{

public PrintMorseCode g(){

dash();dash();dot();space();

return this;

}

public PrintMorseCode a(){

dot();dash();space();

return this;

}

public PrintMorseCode r(){

dot();dash();dot();space();

return this;

}

public PrintMorseCode m(){

dash();dash();space();

return this;

}

}