KB금융그룹 | 국민의 평생 금융파트너

2024년 상반기 K-디지털 트레이닝

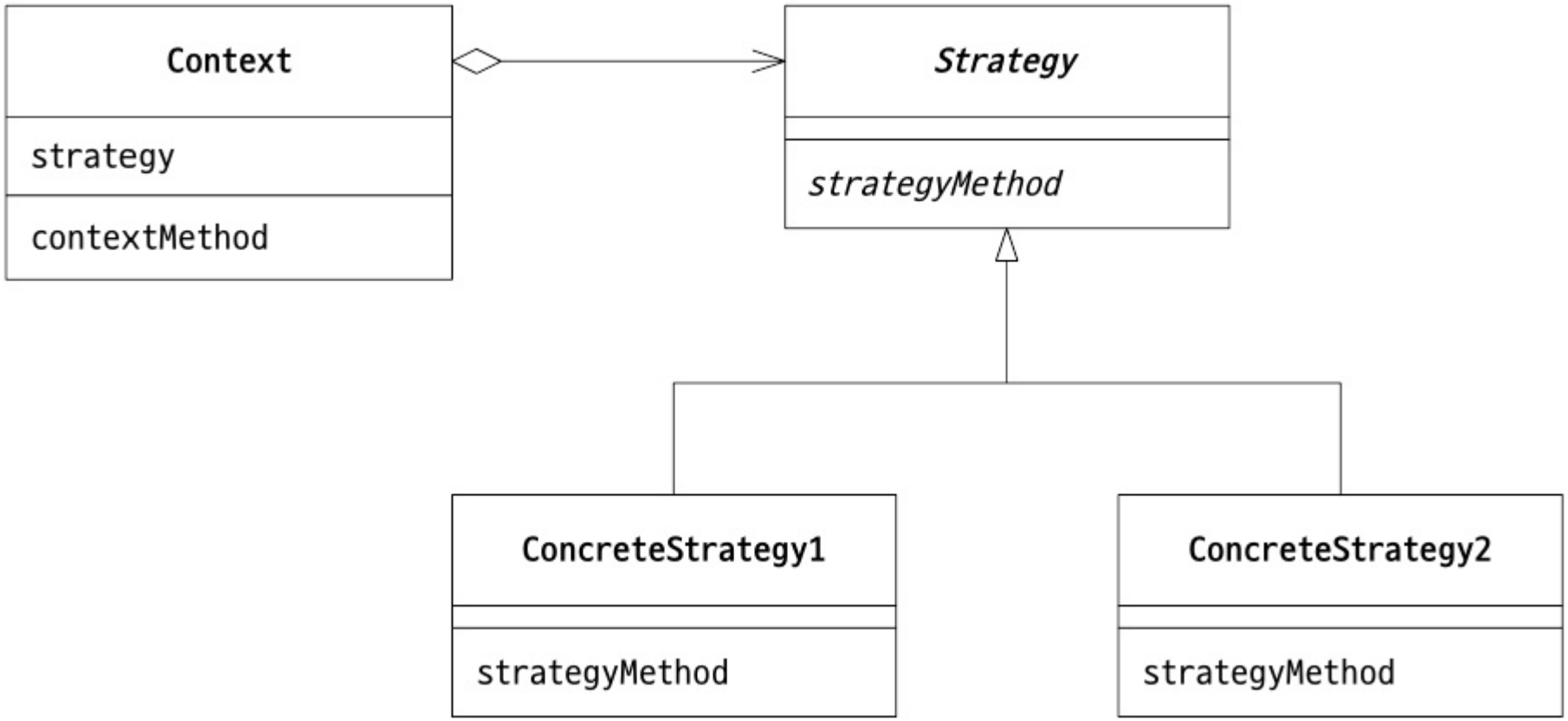
Strategy - 알고리즘을 모두 바꾼다

[KB] IT's Your Life

KB 국민은행

Strategy 패턴

Strategy 패턴의 클래스 다이어그램



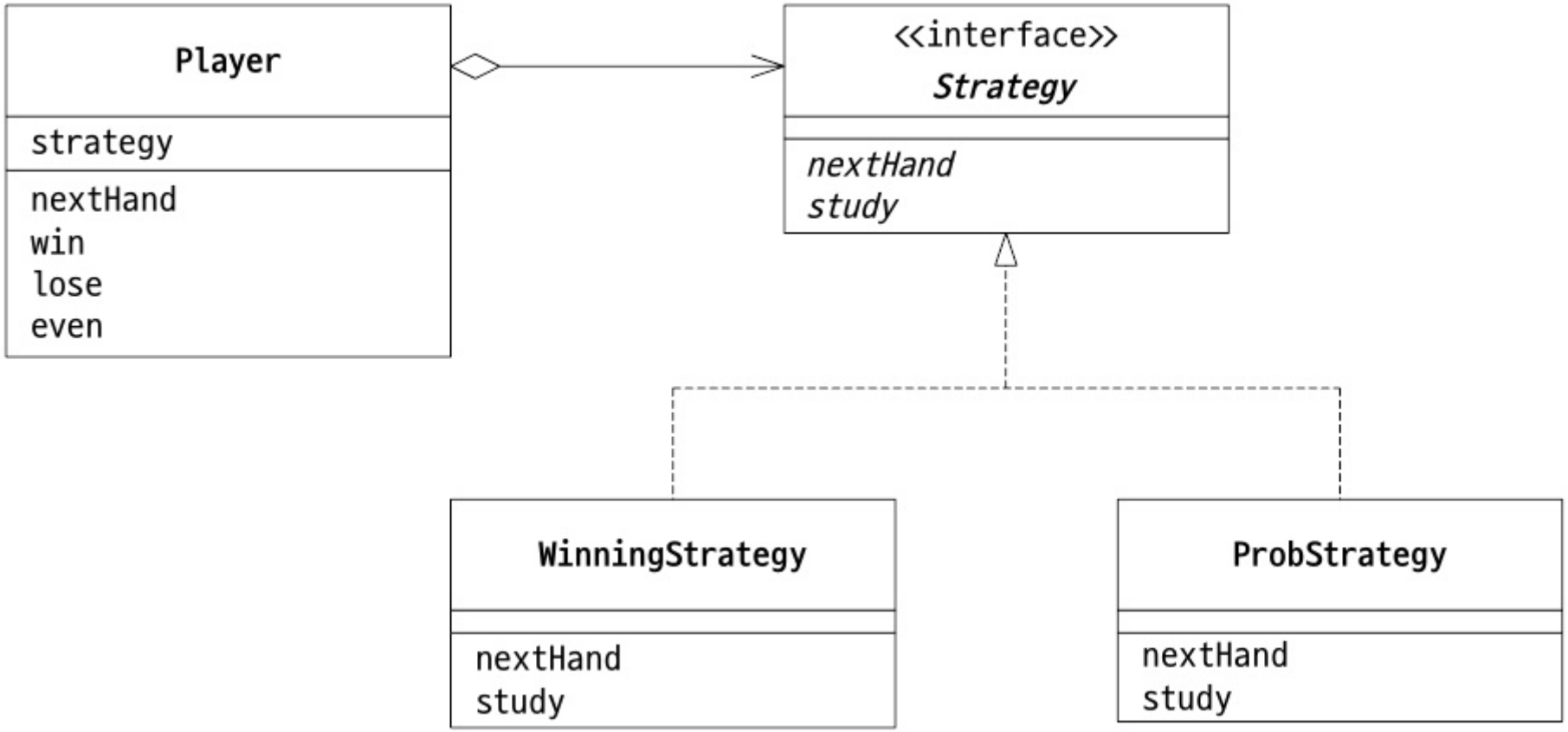
예제 프로그램

- 컴퓨터로 '가위바위보'를 하는 프로그램
- 전략
  - 이기면 다음에도 같은 손을 내는 전략 (WinningStrategy)
  - 직전 손에서 다음 손을 확률적으로 계산하는 전략(ProbStrategy)

이름	설명
Hand	가위바위보의 '손'을 나타내는 클래스
Strategy	가위바위보의 '전략'을 나타내는 인터페이스
WinningStrategy	이기면 다음에도 같은 손을 내는 전략을 나타내는 클래스
ProbStrategy	직전 손에서 다음 손을 확률적으로 계산하는 전략을 나타내는 클래스
Player	가위바위보를 하는 플레이어를 나타내는 클래스
Main	동작 테스트용 클래스

예제 프로그램

예제 프로그램의 클래스 다이어그램



Hand.java

```
package part10.exam01;

public enum Hand {
    ROCK("바위", 0),
    SCISSORS("가위", 1),
    PAPER("보", 2);

    private String name;    // 가위 바위 보 손의 이름
    private int handvalue;  // 가위 바위 보 손의 값

    private Hand(String name, int handvalue) {
        this.name = name;
        this.handvalue = handvalue;
    }

    private static Hand[] hands = {
            ROCK, SCISSORS, PAPER
    };

    public static Hand getHand(int handvalue) {
        return hands[handvalue];
    }
```

Hand.java

```
    // 무승부는 0, this가 이기면 1, h가 이기면 -1
    private int fight(Hand h) {
        if(this == h) {
            return 0;
        } else if((this.handvalue + 1) % 3 == h.handvalue) {
            return 1;
        } else {
            return -1;
        }
    }

    // this가 h보다 강할 때 true
    public boolean isStrongerThan(Hand h) {
        return fight(h) == 1;
    }

    // this가 h보다 약할 때 true
    public boolean isWeakerThan(Hand h) {
        return fight(h) == -1;
    }

    // 가위 바위 보의 문자열 표현
    @Override
    public String toString() {
        return name;
    }
}
```

Strategy.java

```
package part10.exam01;

public interface Strategy {
    Hand nextHand();
    void study(boolean win);
}
```

WinningStrategy.java

```
package part10.exam01;

import java.util.Random;

public class WinningStrategy implements Strategy{
    private Random random;
    private boolean won = false;
    private Hand prevHand;

    public WinningStrategy(int seed) {
        random = new Random(seed);
    }

    @Override
    public Hand nextHand() {
        if(!won) {
            prevHand = Hand.getHand(random.nextInt(3));
        }
        return prevHand;
    }

    @Override
    public void study(boolean win) {
        won = win;
    }
}
```

ProbStrategy.java

```
package part10.exam01;

import java.util.Random;

public class ProbStrategy implements Strategy {
    private Random random;
    private int prevHandValue = 0;
    private int currentHandValue = 0;
    private int [][] history = {
            { 1, 1, 1, },
            { 1, 1, 1, },
            { 1, 1, 1, },
    };

    public ProbStrategy(int seed) {
        random = new Random(seed);
    }
```

ProbStrategy.java

```
@Override
public Hand nextHand() {
    int bet = random.nextInt(getSum(currentHandValue));
    int handvalue = 0;
    if(bet < history[currentHandValue][0]) {
        handvalue = 0;
    } else if(bet < history[currentHandValue][0] + history[currentHandValue][1]) {
        handvalue = 1;
    } else {
        handvalue = 2;
    }

    prevHandValue = currentHandValue;
    currentHandValue = handvalue;
    return Hand.getHand(handvalue);
}

private int getSum(int handvalue) {
    int sum = 0;
    for(int i = 0; i < 3; i++) {
        sum += history[handvalue][i];
    }
    return sum;
}
```

ProbStrategy.java

```
    @Override
    public void study(boolean win) {
        if(win) {
            history[prevHandValue][currentHandValue]++;
        } else {
            history[prevHandValue][(currentHandValue + 1) % 3]++;
            history[prevHandValue][(currentHandValue + 2) % 3]++;
        }
    }
}
```

Player.java

```
package part10.exam01;

public class Player {
    private String name;
    private Strategy strategy;
    private int wincount;
    private int losecount;
    private int gamecount;

    public Player(String name, Strategy strategy) {
        this.name = name;
        this.strategy = strategy;
    }

    // 전략에 따라 다음 손을 결정한다.
    public Hand nextHand() {
        return strategy.nextHand();
    }

    // 승리
    public void win() {
        strategy.study(true);
        wincount++;
        gamecount++;
    }
```

Player.java

```
    // 패배
    public void lose() {
        strategy.study(false);
        losecount++;
        gamecount++;
    }

    // 무승부
    public void even() {
        gamecount++;
    }

    @Override
    public String toString() {
        return "[" +
                name + ":" +
                gamecount + " games, " +
                wincount + " win, " +
                losecount + " lose" +
                "]";
    }
}
```

예제 프로그램

Main.java

```
package part10.exam01;

public class Main {
    public static void main(String[] args) {
        if(args.length != 2) {
            System.out.println("Usage: java Main randomseed1 randomseed2");
            System.out.println("Example: java Main 314 15");
            System.exit(0);
        }

        int seed1 = Integer.parseInt(args[0]);
        int seed2 = Integer.parseInt(args[1]);

        Player player1 = new Player("KIM", new WinningStrategy(seed1));
        Player player2 = new Player("LEE", new ProbStrategy(seed2));
```

Main.java

```
        for(int i = 0; i < 10000; i++) {
            Hand nextHand1 = player1.nextHand();
            Hand nextHand2 = player2.nextHand();

            if(nextHand1.isStrongerThan(nextHand2)) {
                System.out.println("Winner: " + player1);
                player1.win();
                player2.lose();
            } else if(nextHand2.isStrongerThan(nextHand1)) {
                System.out.println("Winner: " + player2);
                player1.lose();
                player2.win();
            } else {
                System.out.println("Even...");
                player1.even();
                player2.even();
            }
        }

        System.out.println("Total result");
        System.out.println(player1);
        System.out.println(player2);
    }
}
```