Template Method Pattern : 일정한 프로세스를 가진 요구사항을 묶어주는 패턴

* 구현하려는 알고리즘이 일정한 프로세스가 있다. (여러 단계로 나눌 수 있다)
* 구현하려는 알고리즘의 변경 가능성이 있따

사용 단계

1. 알고리즘을 여러 단계로 나눈다
2. 나눠진 알고리즘 단계를 메소드를 선언
3. 알고리즘을 수행할 템플릿 메소드를 만든다
4. 하위 클래스에서 나눠진 메소드들을 구현한다.

아래 예시에서 추가사항 있을시 하위 클래스에서 알고리즘 구조의 변경없이 알고리즘을 변경한다 .

예시

package designpattern;

public class Template\_Method\_Pattern { // 원래 다른 패키지에서 가저와야 함

public static void main(String[] args) {

// 접속시 보안 인증 권한 접속 과정으로 나눠서 접근해야함 ==> 요청사항

AbstGameConnectHelper helper = new DefaultGameconnectHelpler();

helper.requestConnection("idandpassword");

//추가요청 보안강화 + 10시 셧다운

}

}

abstract class AbstGameConnectHelper {

protected abstract String doSecurity(String string);

protected abstract boolean authentication(String id, String password);

protected abstract int authorization(String userName);

protected abstract String connection(String info);

// 템플릿 메소드

public String requestConnection(String encodedInfo) {

// 보안작업 -> 암호화된 문자열을 디코드

String decodedInfo = doSecurity(encodedInfo);

String id = "aa";

String password = "bbb"; // 디코딩 한걸로 수신했다고 가정

// 반환된 것을 가지고 id,password 를 받음

if (!authentication(id, password)) {

System.out.println("암호 불일치");

}

String userName = "userName";

// 인증 성공시

//권한획득

int i = authorization(userName);

switch(i){

case -1:

throw new Error("10시 셧다운");

case 0: //게임메니저

System.out.println();

break;

case 1: //유료 회원

break;

case 2: //무료회원

break;

case 3: //권한 없음

break;

default://기타사항

break;

}

return connection(userName);

}

}

class DefaultGameconnectHelpler extends AbstGameConnectHelper{

@Override

protected String doSecurity(String string) {

// 추가요청사항

System.out.println("강화된 알고리즘 ");

System.out.println("디코드");

return string;

}

@Override

protected boolean authentication(String id, String password) {

System.out.println("아이디 암호 확인과정");

return true;

}

@Override

protected int authorization(String userName) {

// 추가요청 10시이후

System.out.println("서버에서 나이를 불러와 확인후 시간과 비교 ");

System.out.println("권한 확인");

return 0;

}

@Override

protected String connection(String info) {

System.out.println("마지막 접속 단계");

return info;

}

}