Java 20-2 Lab 7

32191826 박주희

<Command Pattern이란>

Command Pattern을 이용하면, 요구사항을 객체로 캡슐화 할 수 있으며, 매개변수를 써서 여러가지 다른 요구사항을 집어넣을 수도 있다. 또한 요청 내역을 큐에 저장하거나 로그로 기록할 수도 있으며, 작업 취소 기능도 지원이 가능하다.

< 전체적인 구조>

Shape 🡪 Circle, Rectangle, Triangle

Main클래스(Client)에서 JcomboBox를 이용하여 shape 변경

Command 🡪 MoveDownCommand, MoveUpCommand, MoveLeftCommand, MoveRightCommand

주문을 받아서 전달하여 결과물을 요청하는 execute 메소드

반대로 실행하는 undo 메소드, 즉 다시 되돌리는 메소드

ShapeMoveControl = Invoker

HashMap을 이용하여 execute, Stack을 이용하여 undo

Client=MainFrame. MainClass

<주요 구현 과정>

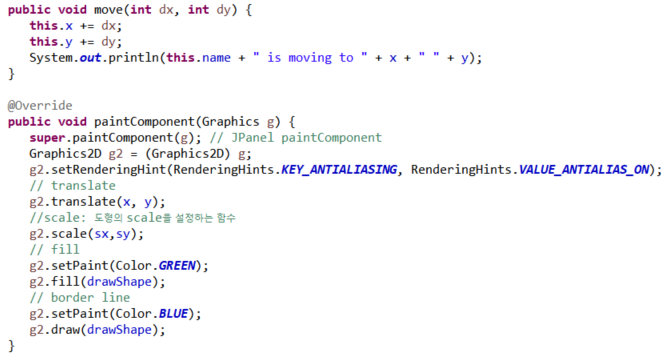
ShapeMoveControl 클래스에서 HashMap와 Stack을 이용하였다. HashMap의 key에는 명령Up/Down/Right/Left을 저장하고 value에는 이에 해당하는 Command 객체를 저장하였다.

Stack을 이용하여 가장 최근에 쌓인 Command를 찾고 해당 Command의 undo메소드를 호출하고 pop 하였다.



주석처리한 부분은 수정 전 코드이다. 이 방법으로 할 경우 setCommand 메소드를 이용할 필요가 없고 HashMap을 굳이 사용할 필요도 없다. 매개변수로 받은 key값을 이용하여 이를 key로 가진 value를 찾고 execute 메소드를 호출한다. 이 command를 stack에도 쌓아준다.

Shape 클래스에 move 메소드를 생성하고 MoveDownCommand, MoveUpCommand, MoveLeftCommand, MoveRightCommand의 execute, undo 메소드에서 이를 이용하여 도형을 움직인다.



translate라는 함수를 처음 배웠다. translate (double tx, double ty): 현재의 Graphics2D Transform를 이동 변환과 연결한다. 처음에는 shape instanceof Triangle/Circle/Rectangle을 하여 각자 다르게 move를 구현하여야 하는줄 알았으나 교수님의 도움으로 translate 함수를 알고 간편하게 구현할 수 있었다.

Client 클래스에서 평소에 자주 하는 실수를 하였다. Move는 모두 되는 undo만 작동하지 않아 디버그를 통해 stack을 검사해보니 stack에 쌓이지가 않고 초기화가 되어 저장되어서 계속 오류가 뜨는 것이었다. ShapeMoveControl shapecontrol=**new** ShapeMoveControl(); 를 actionPerformed 안에 넣어서 오류가 생기는 것이었다. 이를 메소드 밖에 클래스 변수로 만들었더니 제대로 돌아가는 것을 알 수 있었다. 변수 선언 위치의 중요성을 다시 한 번 알 수 있었다.

<My code>

translate에 대해 찾아보면서 scale함수도 알게되어 이를 이용하였다.

BiggerCommand 클래스: shape를 크게하는 명령

SmallerCommand 클래스: shape를 작게하는 명령



Shape 클래스의 changesize 메소드: 메소드 밖에서 double sx,sy를 모두 선언 및 1로 초기화하고 changesize 메소드를 이용하여 sx,sy를 바꿔준다.

paintComponent에서 g2.scale(sx,sy)를 추가한다.

<실행 결과>

