**Report**

**Java Term Project Final Report\_6조**



|  |  |
| --- | --- |
| **과목명** | **자바와 객체지향프로그래밍** |
| **교수명** | **이긍해 교수님** |
| **학과** | **소프트웨어학과** |
| **학번** | **2018125018 / 2017125053**  **2016125088/ 2016126050** |
| **이름** | **김지원 / 성은비 / 심수현 / 이덕희** |
| **제출일** | **2019. 06.03** |



**<목차>**

**제 1장 서론**

* 1. **업무분담**
  2. **개발 목표**
  3. **일정**

**제 2장 본문 구성**

**2.1 요구사항 명세서**

2.1.1 개발할 내용의 정리된 표현

2.1.2 필수 기능

2.1.3 시스템 시나리오

2.1.4 설계 명세서

**2.2 Class Diagram**

**2.3 Class Interface**

2.3.1 Package Figure

2.3.2 Package Frame

2.3.3 Package Information

2.3.4 Package SubFrame

2.3.5 Package SubPanel

**제 3장 결론**

**3.1 필드 구현**

**3.2 Term Project 개인별 기여내용**



**<Java term project>**

**업무 분담(개인기여)**

|  |  |
| --- | --- |
| **김지원** | **성은비** |
| -텍스트 필드구현  -선 굵기 조절  -펜 기능, 굵기 조절  -보고서 작성 | -프로젝트 발표  - 지우개 기능, 굵기 조절  -스택 레이어  -색상 팔레트 구현 |
| **심수현** | **이덕희** |
| -객체 크기 조절, 객체 이동하기  -파일 입출력  -전체적인 UI 구성 | -사각형, 원, 삼각형, 직선 드로잉  - 도형 색 채우기  -객체 색상 바꾸기 |

**개발 목표**

그림판을 JAVA 언어로 programming 하고, 구현하여 누구나 쉽게 사용할 수 있도록 한다. JAVA Swing, awt 등 다양한 개발 package를 이용하여 team project로 협업하여 하나의 그림판 프로그램을 완성한다.

**일정**

3/19: 각자 맡은 기능 자료조사, 프로젝트 계획서 작성

4/9: 기능설계, 클래스설계 및 기본기능 역할분담

4/16, 23: 중간고사 대비 공부, 시험 기간

5/11: 프레임 구현, 기본기능 병합 및 세부기능 분담

5/21: 세부기능 중 어려운 부분 재설계

5 /28: 세부기능 병합 및 발표자료 분담

6/3: 프로젝트 발표



**<요구사항 명세서>**

**개발할 내용의 정리된 표현**

버튼과 리스트 combo box등의 자바의 기본적인 기능을 이용해서 도형 그리기와 텍스트입력, 지우개 등의 기능을 구현한 가진 기본적인 그림판.



**필수 기능**

1) 도형이나 텍스트를 선택가능.

2) 화면에 선택한 도형을 최대 10개 그릴 수 있음

3) 도형의 크기 조절, 위치이동, 지우기를 할 수 있음.

4) 그린 것을 파일로 저장 가능하고 저장한 파일은 불러 올 수 있음.

5) 선의 색, 채우기 색, 선의 굵기를 결정할 수 있음.

**시스템 시나리오(Use Case)**

실행을 시작하면 중간 부분에 흰 배경의 도화지에 그림판 작업을 시작 할 수 있다.

먼저 왼쪽의 버튼들 중에서 원하는 기능을 선택한다. 기본 기능은 ‘펜’으로 설정 되어있다.

-도형 그리기

‘원’, ‘사각형’, ‘삼각형’, ‘선’ 그리기 버튼을 클릭하면 각각 모양에 맞는 도형의 그림이 그려진다.

각각의 도형이 그려지는 순간, 오른쪽에 위치한 Figure Stack부분에 도형의 정보를 담은 버퍼가 리스트 형식으로 push된다.

그리기 버튼 클릭 후, 팔레트의 색을 선택한 후 사용할 수 있다.

-펜으로 그리기

‘연필’모양 버튼은 펜으로 그리는 기능이다. 자유 드로잉이 가능한데, 이것도 마찬가지로 그린 후에 마우스를 떼는 순간 stack에 리스트 형식으로 쌓인다.

‘연필’ 버튼을 클릭 한 후에 line size, 팔레트를 통해 원하는 굵기, 색을 지정한 후 사용할 수 있다.

-개체 이동, resize

버튼 패널의 맨 위에 위치한 ‘화살표’ 버튼을 클릭하면 개체를 이동시킬 수 있다.

그 아래의 또 다른 ‘검은색 화살표’ 버튼은 Resize 버튼으로 개체의 크기를 확대, 축소가 가능하다.

이 두 기능은 버튼을 누른 후 Figure stack에서 정보를 변경하고 싶은 개체를 선택한 다음에 사용이 가능하다. (선택 하지 않은 경우 오류 메시지를 발생한다.)

-색 선택, 채우기(팔레트)

‘펜 그리기’의 경우에는 ‘연필’ 아이콘을 선택한 다음 팔레트의 색을 선택하면 바로 색이 적용되어 그려진다.

도형의 색 선택 기능은 개체 이동/resize와 마찬가지로 stack부분을 사용한다.

먼저 버튼 중에서 ‘페인트’ 버튼을 클릭 한 후, stack에서 그려진 개체 중 하나를 선택한다. 그리고 팔레트의 색을 선택하면 선택한 색으로 도형/선의 색이 바뀐다. (stack에서 개체를 클릭하지 않으면 오류 메시지를 생성한다.)

-지우개

‘지우개’ 버튼을 클릭한 후, 지우면 그렸던 그림이 지워진다.

‘Eraser size’ 버튼을 통해 지우개의 굵기를 조절할 수 있다.

-텍스트 삽입

‘텍스트’ 버튼을 클릭하고, 도화지의 삽입을 원하는 위치에 클릭하면 텍스트를 입력 할 수 있는 다이얼로그가 생성되고, 입력하면 화면에 출력된다.

텍스트 삽입 버튼을 누른 후, Text size/Font style을 사용해서 텍스트의 사이즈와 폰트를 바꾸어 사용할 수 있다.

-Figure Stack(레이어)

스택(레이어)부분은 그림판의 오른쪽에 위치하였고, 개체의 정보를 저장하고, 선택하여 바꿀 수 있는 부분이다. 스택의 아래쪽에 4개의 버튼이 있다.

스택에서 개체를 선택하고, 버튼을 클릭하면 각각 버튼의 기능이 구현된다.

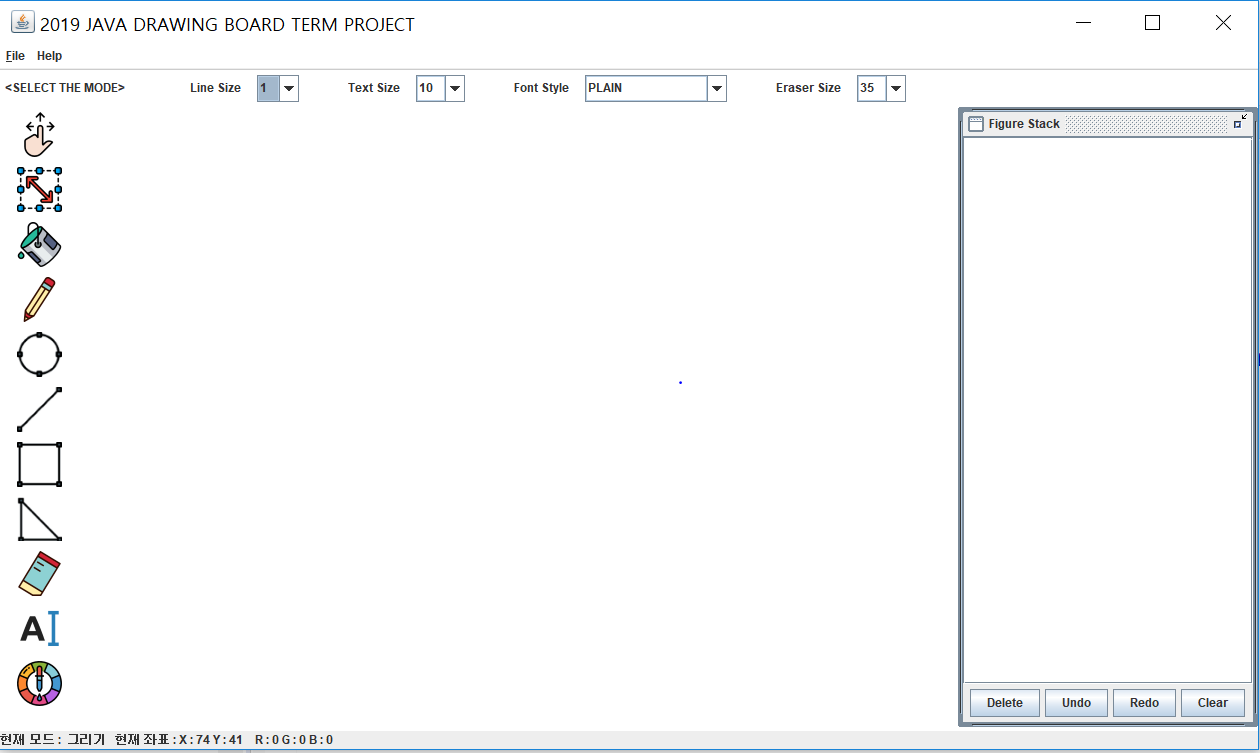
Delete버튼은 개체를 선택된 개체를 지운다. Undo 버튼은 뒤로 되돌리기 기능, Redo 버튼은 앞으로 되돌리기 이고, 마지막으로 clear버튼은 스택에 있는 모든 개체를 지운다.

**설계 명세서**

**Editor의 개념도**

스크린샷이(가) 표시된 사진

자동 생성된 설명





**<Class Diagram>**

1. **Package Figure**

텍스트, 지도이(가) 표시된 사진

매우 높은 신뢰도로 생성된 설명

1. **Package Frame**

스크린샷이(가) 표시된 사진

매우 높은 신뢰도로 생성된 설명



1. **Package Information**

스크린샷이(가) 표시된 사진

매우 높은 신뢰도로 생성된 설명

1. **Package SubFrame**

스크린샷이(가) 표시된 사진

매우 높은 신뢰도로 생성된 설명



1. **Package SubPanel**

스크린샷이(가) 표시된 사진

매우 높은 신뢰도로 생성된 설명



**<Class Interface>**

**1. Package Figure**

1. **Class Figure**

public Figure()

-> Serializable ,Cloneable 인터페이스를 상속한다.

-> 현재의 색을 가져온다.

public void setColor(Color color), public Color getColor()

-> 채움 색, 선 색을 설정하고, 리턴한다.

public abstract void moveTo(int curX,int curY);

-> 각각의 그리는 개체의 class에 override하여 구현

-> 개체의 현재 위치를 받아와 모든 정점을 이동시킨 후, repaint한다.

public abstract void setFigureSize (int width, int height), public abstract int getSize();-> 개체의 사이즈를 설정하고, 리턴한다.

public abstract void drawFigure(Graphics2D g);

-> 개체를 그린다.

public abstract void calcFigure(int currentX, int currentY)

-> 현재 Mode에 따라 사이즈를 재설정한다.

public abstract Figure clone();

-> 현재 객체의 정보를 리턴한다.



1. **Class RecType**

public RecType(int startX, int startY,int width,int height)

-> Figure 클래스를 상속받는다.

public void setFigureSize() , public int getSize(),

public abstract void drawFigure();

public void calcFigure(), public abstract Figure clone()

-> Figure클래스에서 override한다.

1. **Class Circle**

public Circle(int startX, int startY,int width,int height)

-> x,y 좌표와 width, height를 받아서 원 생성

-> Rectype클래스를 상속받는다.

public void drawFigure()

-> Figure 클래스에서 override 하여, 색을 설정하고 원을 그린다

public String toString()

->override하여 그려진 원의 정보를 print한다.

public Figure clone()

-> Figure클래스에서 Override한다.

1. **Class rectangle**

public Rectangle(int startX, int startY,int width,int height)

-> x, y 좌표와 width, height를 받아서 사각형 생성

-> Rectype 클래스를 상속받는다.



public void drawFigure()

-> Figure 클래스에서 override 하여, 색을 설정하고 사각형을 그린다

public String toString()

->override하여 그려진 사각형의 정보를 print한다.

public Figure clone()

-> Figure클래스에서 Override한다.

1. **Class Line**

public Line(int startX, int startY ,int endX,int endY)

-> 시작점과 끝점의 X, Y 좌표를 받아 라인을 생성한다.

-> Figure 클래스를 상속받는다.

public void setFigureSize(int endX, int endY), public int getSize()

-> Figure 클래스에서 Override해서 크기를 설정하고, 리턴한다.

public void drawFigure()

-> Figure클래스에서 Override해서 현재 굵기와, 색의 정보를 가져와서 선을 그린다.

public void calcFigure(int currentX, int currentY)

-> Figure 클래스에서 Override해서 사이즈를 설정한다.

public void moveTo(int curX,int curY)

-> Figure 클래스에서 Override하고, 현재의 좌표를 받아와서 repaint한다.

public String toString()

-> Override해서 그려진 라인의 정보를 print한다.

public Figure clone()

-> Figure 클래스에서 Override 한다.



1. **Class Triangle**

public Triangle(int startX, int startY)

-> Figure 클래스를 상속받는다.

-> x,y좌표를 가져와서 삼각형을 생성한다.

public void setFigureSize(), public int getSize()

-> Figure 클래스에서 Override해서 크기를 설정하고, 리턴한다.

public void drawFigure()

-> Figure클래스에서 Override해서 색을 설정하고 다각형을 그린다.

public void calcFigure()

-> Figure 클래스에서 Override해서 사이즈를 설정한다.

public void moveTo(int curX,int curY)

-> Figure 클래스에서 Override하고, 현재의 좌표를 받아와서 repaint한다.

public String toString()

-> Override해서 그려진 삼각형의 정보를 print한다

public Figure clone()

-> Figure 클래스에서 Override 한다.

1. **Class Eraser**

public Eraser(int firstX, int firstY)

-> Figure클래스를 상속받는다.

-> 좌표를 가져와서 지우개를 생성한다

public void drawFigure()

-> Figure클래스에서 Override해서 정해진 굵기와, 흰색을 가져와서 선을 그린다.

public String toString()

-> Override해서 지우개 상태를 print한다



public Figure clone()

-> Figure 클래스에서 Override 한다.

1. **Class PenType**

class PenTypeTracePoint

-> Figure 클래스를 상속받는다.

-> Serializable 인터페이스를 상속한다.

-> 이전 좌표와, 나중 좌표를 받아 그리는 penType 클래스를 만든다.

public PenType(int firstX, int firstY)

-> 좌표를 인자로 받아오며, PenTypeTracePoint를 통해 좌표를 연속적으로 추가해서 그림을 그린다.

public void drawFigure(Graphics2D g)

-> Figure클래스에서 Override하고, 현재 좌표를 읽어서 좌표를 연속적으로 추가해 그림을 그린다.

public void calcFigure(int currentX, int currentY)

-> Figure 클래스에서 Override한다.

-> Resize Mode 이면, 개체의 사이즈를 설정한다.

public void moveTo(int curX,int curY)

-> Figure 클래스에서 Override 하며 현재의 좌표를 받아서 모든 정점을 이동하고 repaint한다.

public abstract Figure clone()

-> Figure 클래스에서 Override한다.

1. **Class Pen**

public Pen(int firstX, int firstY)

->PenType 클래스를 상속받는다.

-> 시작점의 좌표를 인자로 받으며, 현재 굵기의 정보를 가져온다.



public void drawFigure()

-> Figure 클래스에서 Override한다.

-> 선 굵기, 색을 설정하고, 마우스를 뗄때까지 화면에 연속적으로 좌표를 추가하여 그림을 그린다.

public String toString()

-> Override해서 펜의 정보를 print한다

public Figure clone()

-> Figure 클래스에서 Override 한다.

1. **Class Text**

public Text(int StartX, int StartY, String str)

-> Figure 클래스를 상속한다.

-> 현재의 텍스트의 사이즈와 폰트 스타일을 받아온다.

-> 클릭한 위치의 좌표를 받아온다.

-> 입력한 텍스트를 받아온다

public void drawFigure(Graphics2D g)

-> Figure클래스에서 Override하고, 색, 폰트를 설정하고, 클릭한 위치에 원하는 문자열을 그린다.

public void calcFigure(int currentX, int currentY)

-> Figure 클래스에서 Override한다.

-> Resize Mode 이면, 개체의 사이즈를 재설정한다.

public void moveTo(int curX,int curY)

-> Figure 클래스에서 Override하고, 현재의 좌표를 받아와서 repaint한다.

public String toString()

-> Override해서 펜의 정보를 print한다

public Figure clone()

-> Figure 클래스에서 Override 한다.



**2. Package Frame**

**Class Main**

Public static void main

=>그림이 그려지는 Frame 창을 열고 닫고, 지정된 사이즈로 information 창을 띄우는 함수이다.

**Class MainDesktopPane**

JDesktopPane을 상속받은 class이다.

Public static MainDesktopPane getInstance

=>MainDesktopPane의 객체를 반환한다.

Public MainDesktopPane

=>changeCursor 메소드를 반환받고, background color를 정하고, 우측 하단의 panel과 중앙의 panel을 형성하였다.

Public void paintComponent

=>우측 하단의 panel과 중앙의 panel의 위치를 정하였다.

Public void changeCursor

=>각 그리기 모드에 따라 마우스 커서를 바꾸었다.

**Class MainFrame**

JFrame을 상속받는 class이다.

Public static MainFrame getInstance

=>MainFrame의 객체를 반환한다.

Public MainFrame

=>왼쪽, 상단, 하단, 메뉴바의 panel을 생성하였다.

Public void setBottonLabel

=>color와 mode의 정보를 불러와서 창의 아래에 현재 모드와 좌표를 나타낸다.

Public void setBottomLabel

=>mode의 정보를 불러오고, 마우스가 창을 벗어날 경우 out of frame을 나타낸다.

**3. Package information**

**Class Information**

Public static final 변수들이 존재

Public static Toolkit getToolkit()

=>화면에서 마우스 커서 모양을 바꾸기 위한 toolkit객체 반환

Private static int textStyle

=>textstyle은 변수에 선택된 폰트를 넣는다.

Public static int gettextStyle

=>선택된 폰트 값을 반환한다.

Private static int textSize=10; public static void setTextSize(int size)

=>초기값을 10으로 정하고 변수에 선택된 사이즈 값을 넣고 폰트사이즈값을 반환한다.

Private static int LineSize=1; public static void setLineSize(int size); public static int getLineSize()

=>선 두께의 초기값을 1로 정하고 변수에 선택된 사이즈 값을 넣고 선두께의 값을 반환한다.

Private static void setCurrentFigure(Figure cur)

=>현재의 도형 객체를 설정한다.

Public static Figure getCurrentFigure()

=>현재 도형객체를 반환한다.

Public static void setCurrentpanel(DrawPanel current)

=>현재 drawpanel 객체를 설정한다.

Public static DrawPanel getCurrentJPanel()

=>현재 drawpanel 객체를 반환한다.

Public static MainDesktopPane setCurrentMainDesktopPane()

=>현재 MainDesktopPane 객체를 설정한다.

Public static MainDesktopPane getCurrentMainDesktopPane()

=>현재 MainDesktopPane 객체를 반환한다.

Public static void addDrawFrame\_Count(){*DrawFrame\_Count*++;}

=>그리기 창을 띄울 때마다 변수를 1씩 증가시키는 기능을 한다.

public static int getDrawFrame\_Count(){return DrawFrame\_Count;}

=>카운트한 변수를 반환하는 기능을 한다.

public static String getCurrentModeToString()

=> 현재 그리기 모드가 무엇인지 각 케이스에 따라 문자열로 반환한다.

public static int getCurrentMode()

=>현재 그리기 모드값(int)를 반환한다.

public static void setCurrentMode(int mode)

=>현재 그리기 모드값(int)를 설정한다.

Public static Color getCurrentColor()

=>현재 그리기 색깔을 반환한다.

public static void setCurrentColor(int R, int G, int B)

=>현재 그리기 색깔을 설정한다.

**4. Package SubFrame**

**Class BottomLabel**

public BottomLabel(String name)

=>인자로 전달받은 String 값을 라벨에 저장하여 현재 좌표 값이나 색상 값이 표시 되게 한다.

**Class CenterPanel**

public static CenterPanel getInstance()

=>CenterPanel의 객체를 반환한다.



public CenterPanel()

=>addMouseListener를 이용해 mouseEntered 이벤트의 경우에 changeCursor를 사용하여 현재 그리기 모드에 알맞은 커서 모양이 나오게 한다.

public void replacePanel(DrawPanel panel)

=>DrawPanel 객체인 draw의 figureSet을 changeVector를 이용하여 인자로 전달받은 panel의 figureSet Vector로 바꾼다.

public void changeCursor()

=>각 그리기 모드에 따라 커서의 모양이 바뀌게 한다.

**Class LeftPanel**

public LeftPanel()

=>왼쪽 도구상자의 버튼을 생성하고 아이콘 이미지를 불러와 삽입한다. setName으로 각 그림으로 되어있는 버튼에 name을 부여하고 addListener를 이용하여 이벤트를 받을 수 있게 끔 해 놓았다.

8개의 색깔을 배열에 저장한 후 8개의 버튼을 add하면서 각 색상을 버튼에 입힌다.

public void actionPerformed(ActionEvent e)

=>각 버튼을 눌렀을 때 버튼에 부여되었던 name값을 가져와 Information.setCurrentMode로 현재 그리기 모드의 값을 바꾸는 이벤트를 정의한다.

**Class RightInternalFrame**

public RightInternalFrame()

=>화면에 들어갈 list와 하단의 4개의 버튼을 추가한 후 list는 addListSelectionListener로 이벤트를 받을 수 있게 하였고 4개의 버튼은 각 버튼을 만들 때 마다 이벤트를 정의한다.

Delete버튼의 이벤트는 DrawPanel의 deleteFigure를 사용하여 리스트의 선택된 도형을 삭제한다.

Undo버튼의 이벤트는 DrawPanel의 popStackTrace를 이용하여 이전상태로 돌아가게 한다.

Redo버튼의 이벤트는 DrawPanel의 popStackCaneStack을 이용하여 취소한(이전상태로 돌아간) 상태를 다시 되돌리게 한다.

Clear버튼의 이벤트는 DrawPanel의 clearFigure를 사용하여 도형을 모두 지우도록 한다.

public void valueChanged (ListSelectionEvent e)

=>Information 클래스에 현재 도형을 현재 Draw Panel의 figureSet의 선택된 인덱스 번째의 값을 셋 한다.

public void setListItems(Vector<Figure> dataSet)

=>인자로 받은 Vector를 현재 list를 모두 비운 후 새로 넣음으로써 도형이 추가되거나 삭제될 때 리스트를 업데이트 할 수 있다.

public int getCurrentFigureIndex()

=>선택된 인덱스 값을 반환한다.

public static RightInternalFrame getInstance()

=>RightInternalFrame 객체를 반환한다.

**Class TopMenu**

public TopMenu()

=> save메뉴의 이벤트는 jFileChooser를 이용하여 저장하는 창을 띄우고 원하는 경로에 파일을 저장하게 되면 ObjectOutputStream 객체를 이용하여 현재 DrawPanel의 값을 직렬화 하여 저장한다.

exit메뉴의 이벤트는 System.exit(0)를 이용하여 프로그램을 종료하게한다.



**Class TopPanel**

public TopPanel()

=>1~100까지의 크기를 갖는 라인사이즈, 텍스트 사이즈 콤보박스를 추가하고, 6가지의 텍스트 타입을 저장한 콤보박스를 추가한다. 각 콤보박스를 선택했을 때 일어나는 이벤트는 Information클래스의 setTextSize,setLineSize,setTextStyle을 사용하여 값을 변경한다.

**5. Package SubPanel**

**1) class DrawPanel**

public DrawPanel()

=>addMouseMotionListener와 addMouseListener를 호출하여 마우스 이벤트를 받을 수 있게 한다.

public void paintComponent(Grahpics g)

=>for each문을 통해 figureSet Vector에 있는 figure 객체들을 하나씩 가져와 그림을 그림

public void mouseDragged (MouseEvent e)

=> setBottomLabel을 이용하여 마우스가 움직일 때 마다 좌표 값을 표시할 수 있게 해준다.

=> 현재 모드를 받아와서 MOVE라면 dragStartX,Y를 이용하여 얼마만큼 움직였는지를 moveX,Y에 저장하고 이를 moveTo함수에 인자로 전달하여 도형이 이동한 X,Y좌표를 저장한다.

=>현재 모드가 RESIZE라면 Figure클래스의 calcFigure를 호출하여 마우스의 좌표를 통해 도형의 너비와 높이를 조정한다.

=>MOVE,PAINT,RESIZE 가 아닐때는 drawCurrentFigureFunc()를 호출하여 현재 그리고 있는 도형이 드래그 하고 있는 마우스 좌표에 따라 변하는 모습을 보여주게 한다.

public void mouseMoved(MouseEvent e)

=>마우스가 드래그가 아닌 화면에서 움직이는 순간에도 아래 BottomLabel에 표시되는 마우스의 좌표값을 표시할 수 있어야 한다.

=>setBottomLabel을 이용하여 재정의 한다.

public void mouseClicked(MouseEvent e)

=>마우스가 클릭되는 순간에 현재 figureSet Vector를 RightInternalFrame에 전달하여 리스트를 set한다.

=> 현재 모드가 MOVE일 때 dragStartX,Y에 움직이기 시작하는 좌표 값을 저장한다.

=> 현재 모드가 PAINT일 때 Information클래스를 통해 현재 색깔을 불러와서 색깔을 설정 한 후, repaint 한다.

public void mouseExited(MouseEvent e)

=>마우스가 화면 위에 없다는 것을 BottomLabel에 표시될 수 있게 한다.

public void mousePressed(MouseEvent e)

=> mouseClicked와 마우스의 동작이 같으므로 같은 기능을 가지고 있다.

=>그리기 모드가 MOVE와 PAINT가 아닐 때 drawFigureFunc를 호출한다.

public void mouseReleased(MouseEvent e)

=>RightInternalFrame에 새로 도형이 추가된 figureSet으로 업데이트 하여 리스트에 도형이 추가되게 한다.

=> figureStack에 현재 figureSet을 쌓는다.

private void drawFigureFunc(MouseEvent e)

=>현재 그리기 모드 상태에 따라 각 도형의 객체를 생성하여 figureSet에 추가한다.

public void drawCurrentFigureFunc(MouseEvent e, FIgure temp)

=>인자로 전달받은 객체를 calcFigure를 이용하여 마우스 좌표에 따라 값을 바꾼 후 repaint한다.

=>mouseDragged에서 이 매써드를 사용하여 마우스의 좌표에 따라 도형의 모습이 변하는 모습을 표현할 수 있게 한다.

public void changeVector(Vector<Figure> vector)

=>인자로 전달 받은 vector를 현재 figureSet에 덮어 씌운다

public void addVector(Vector<Figure> addData)

=>figureSet에 인자로 전달받은 vector를 붙여 넣고 RightInternalFrame에 리스트를 업데이트 한다.

=>figureStack에 figureSet을 통해 현재 도형

public void addVector(Figure addData)

=>figureSet에 인자로 전달받은 객체 하나를 붙여넣고 RightInternalFrame에 마찬가지로 업데이트 한다.

=> FigureStack에 figureSet을 통해 현재 도형 상태를 저장

public void clearFigure()

=>figureSet, figureStack, figureCancelStack을 clear메써드를 통해 비우고 RightInternalFrame에 모두 없어진 상태를 업데이트 한다.

public void deleteFigure(int idx)

=>figureStack에 지우기 전의 상태를 저장한 후 인자로 전달받은 인덱스의 figureSet값을 지운다.

public void popStackTrace()

=>RightInternalFrame에서 undo 할 때 figureCancelStack에 현재 figureSet의 정보를 저장한다.

=> figureSet과 figureStack의 맨위의 값이 같다면 Stack을 pop한다.

=>Stack 이 비어있지 않은 경우에 figureStack의 맨 위의 값을 figureSet에 저장한후 figureStack을 pop한다.

=> repaint를 호출하여 undo한 상태를 바로 표현하게 한다.

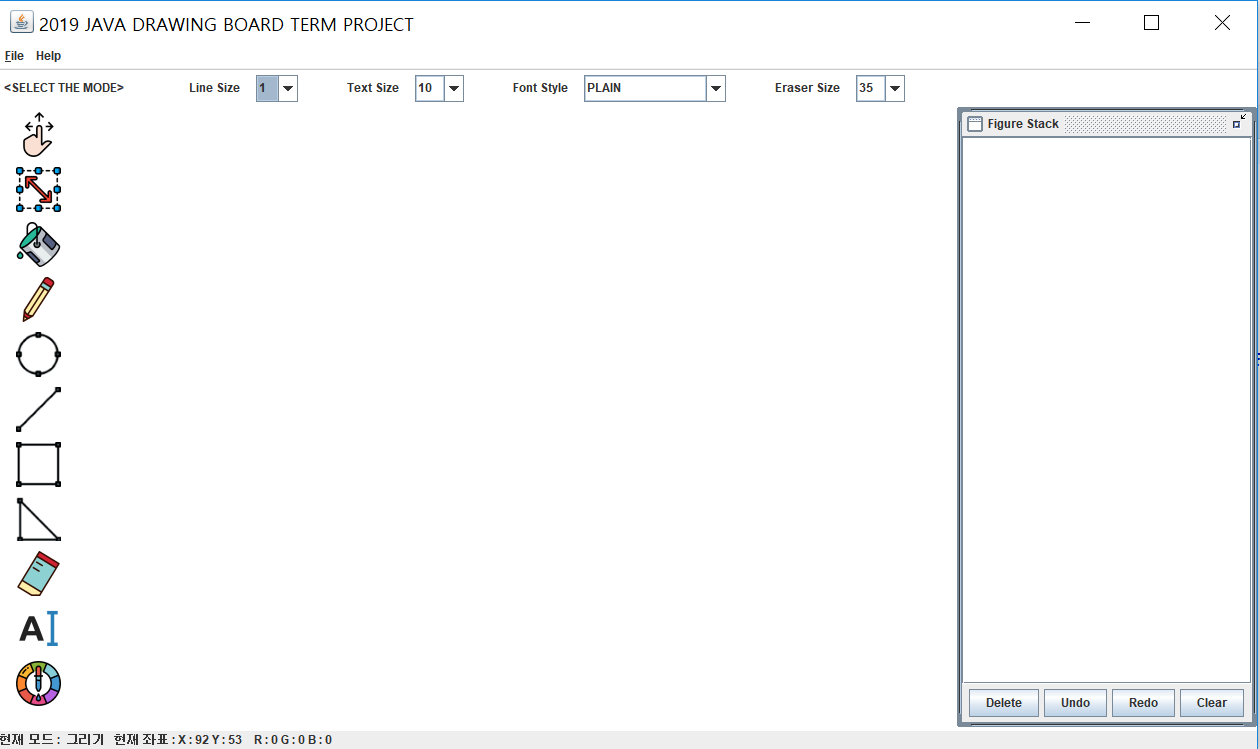
public void popStackCaneStack()

=>figureCancelStack이 비어있지 않을 때 figureStack에 현재 figureSet을 저장하고 figureSet에는 figureCancelStack을 pop하면서 반환 값을 저장한다.

=>repaint를 호출하여 redo한 상태를 바로 표현하게 한다.

**<구현>**

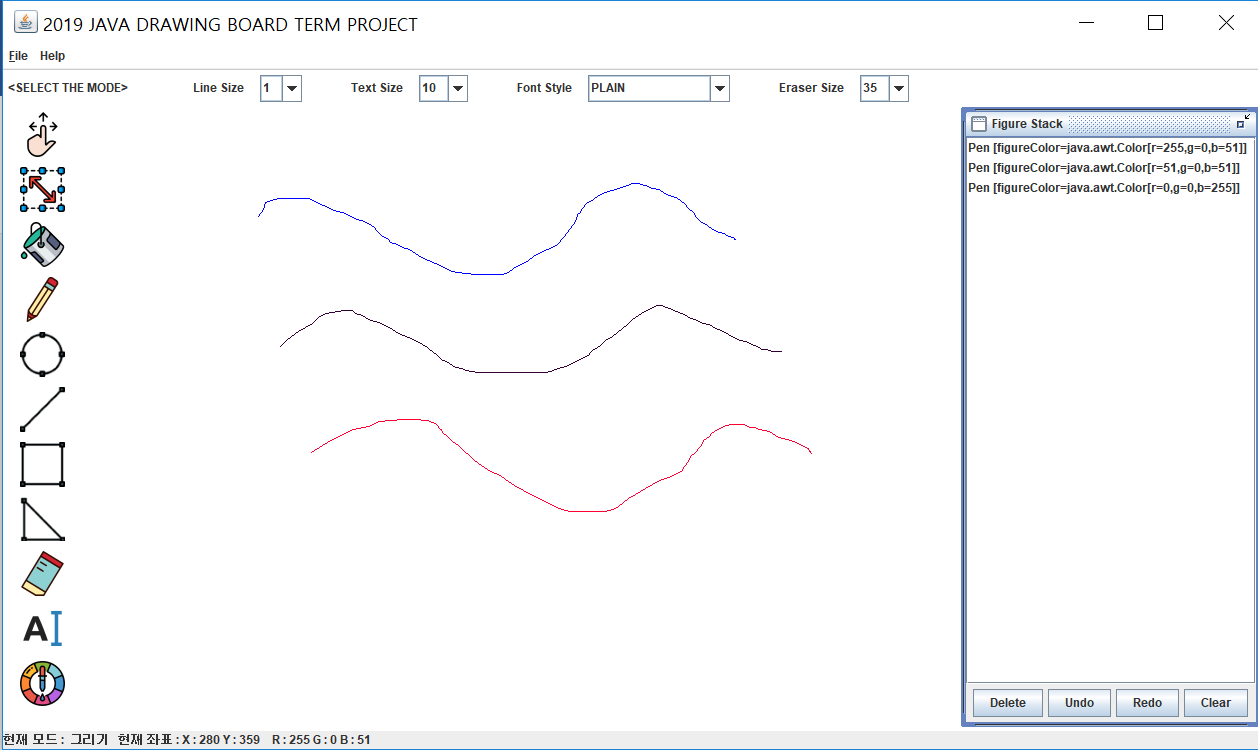
첫 실행 화면



**1)펜으로 그리기**

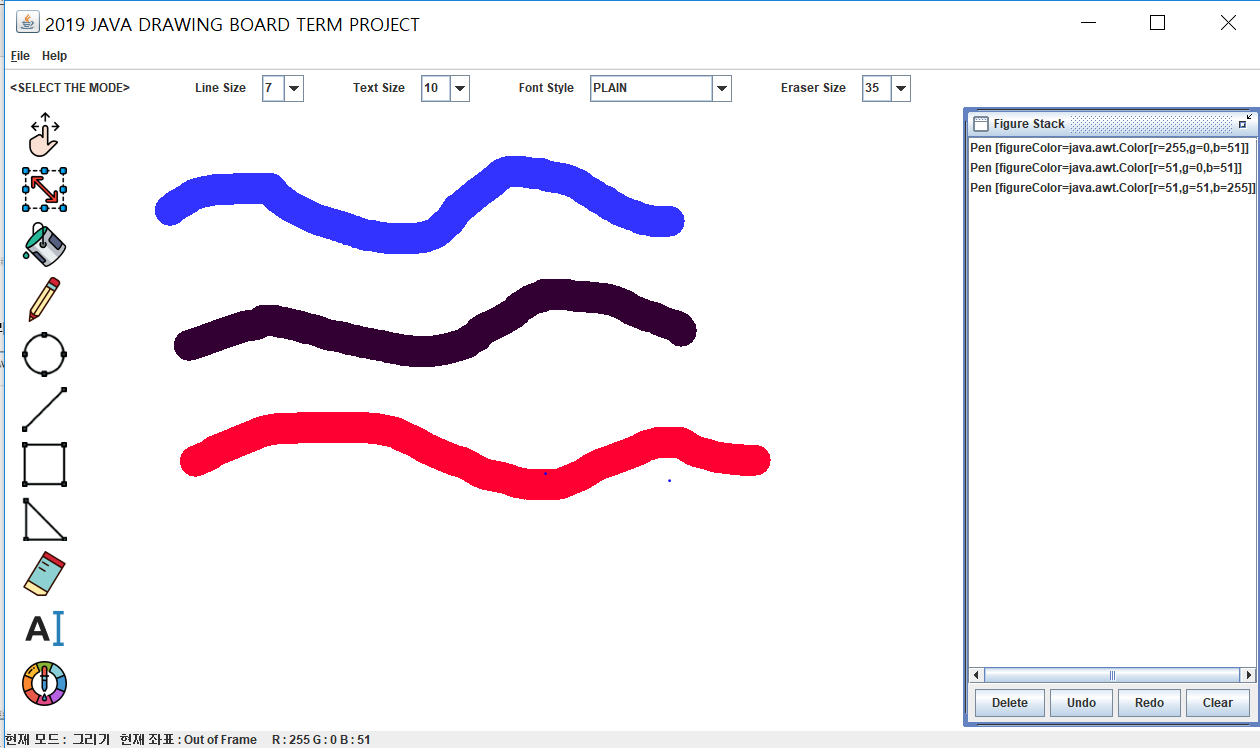
펜을 선택 후 색을 선택하여 그린 화면으로 그린 후에는 figure stack에 각각 기록이 남는다.

Line size는 1로 초기화 되어있다. 초기 색상은 흰색으로 정해 두었다.





Line size를 7로 바꾸어 그린 결과화면이다.



**2)도형 그리기**

처음 도형을 그리면 색이 채워지지 않은 검은 선의 도형이 그려진다.

스크린샷이(가) 표시된 사진

자동 생성된 설명



색을 선택한 뒤 도형을 그리면 색이 채워진 채로 그려진다.

스크린샷이(가) 표시된 사진

자동 생성된 설명

도형의 색을 바꾸고 싶다면, 우선 Figure stack에서 찾아 클릭하여 색을 바꿀 도형을 정한다.

그리고 페인트 버튼을 클릭하고, 색상 팔레트에서 색을 선택한 뒤 해당 도형에 클릭하면 도형의 색이 바뀐다. 도형의 테두리 색 역시 페인트 버튼을 이용해서 바꿀 수 있다.

스크린샷이(가) 표시된 사진

자동 생성된 설명



**3)텍스트 삽입**

텍스트 버튼을 클릭하고 빈화면을 클릭하면 텍스트 입력 내용을 받는 입력창이 나타난다.

스크린샷이(가) 표시된 사진

자동 생성된 설명

처음 입력된 텍스트는 기본 크기인 10size로 출력되며, size를 60으로 바꾼 텍스트도 출력한 결과이다.

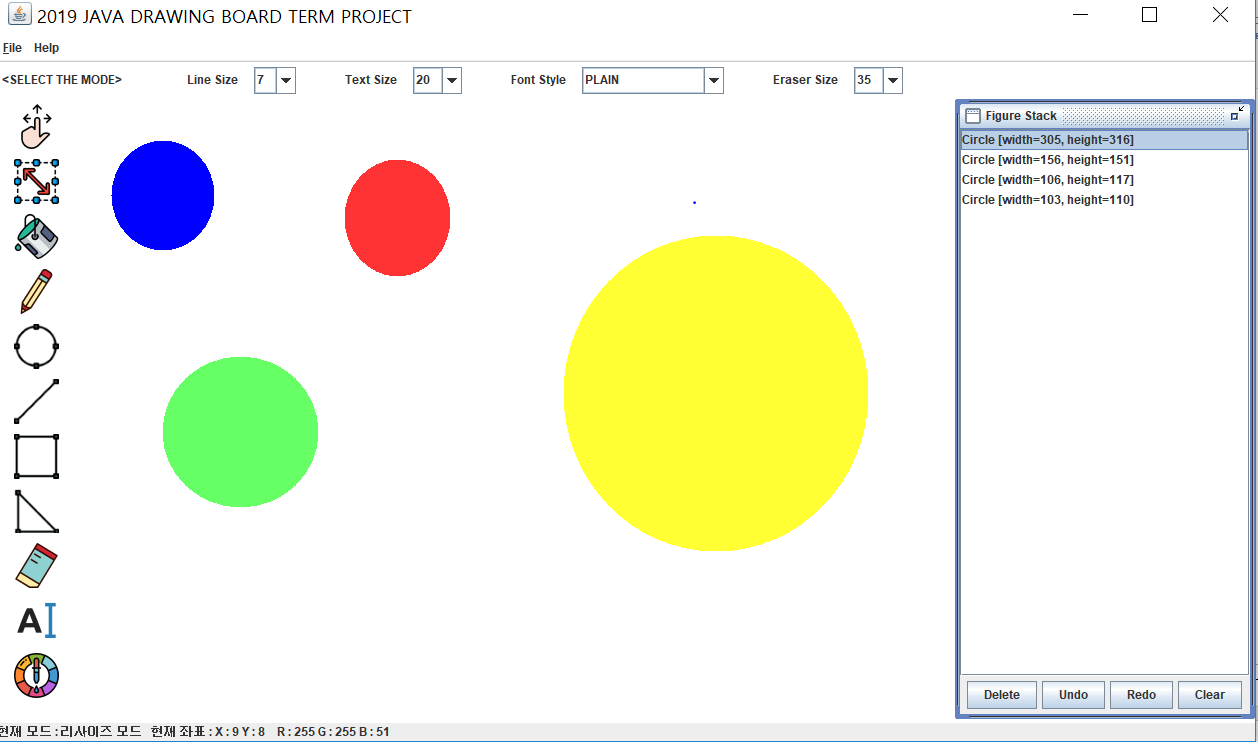
스크린샷이(가) 표시된 사진

자동 생성된 설명

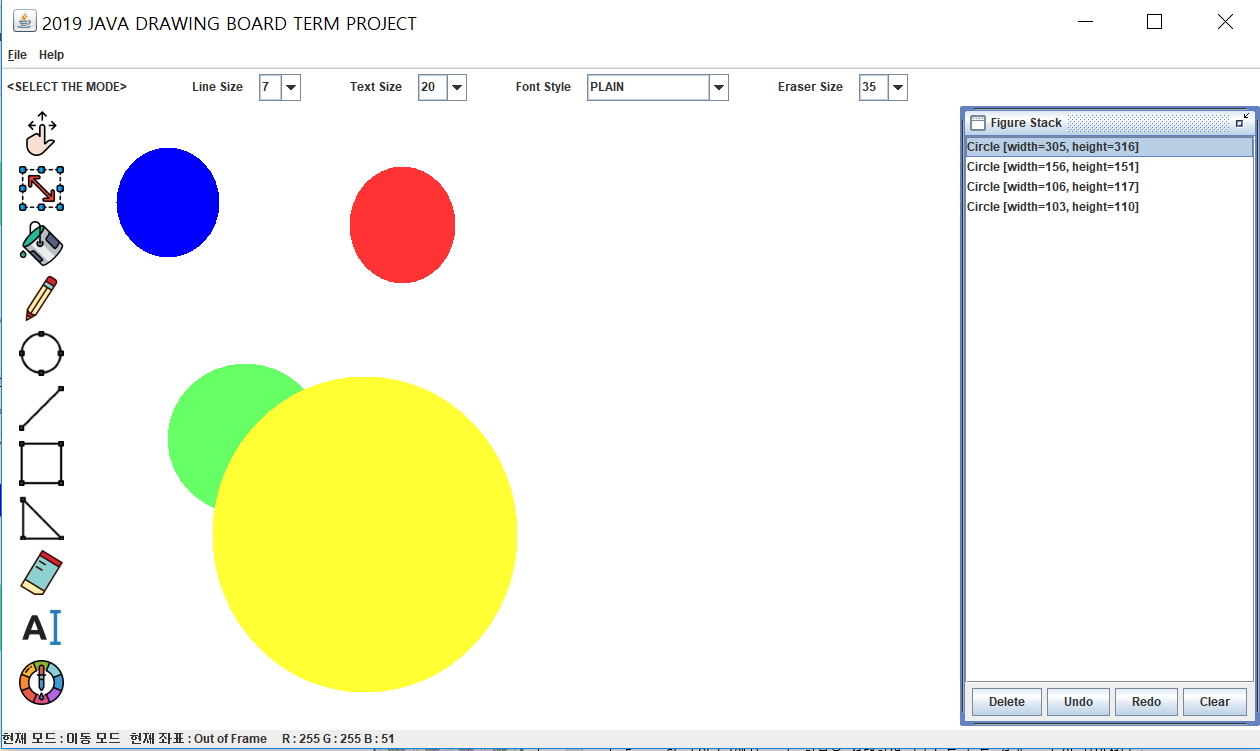


**4)Resize, 도형 옮기기**

스택의 항목을 선택 후 resize 버튼을 누르고 도형을 선택하면 도형의 크기를 바꿀 수 있다.



Stack의 항목을 선택한 후 도형 옮기기 버튼을 누르면 도형을 이동시킬 수 있다.





**5)Stack (delete/ undo/ redo/ clear)**

delete

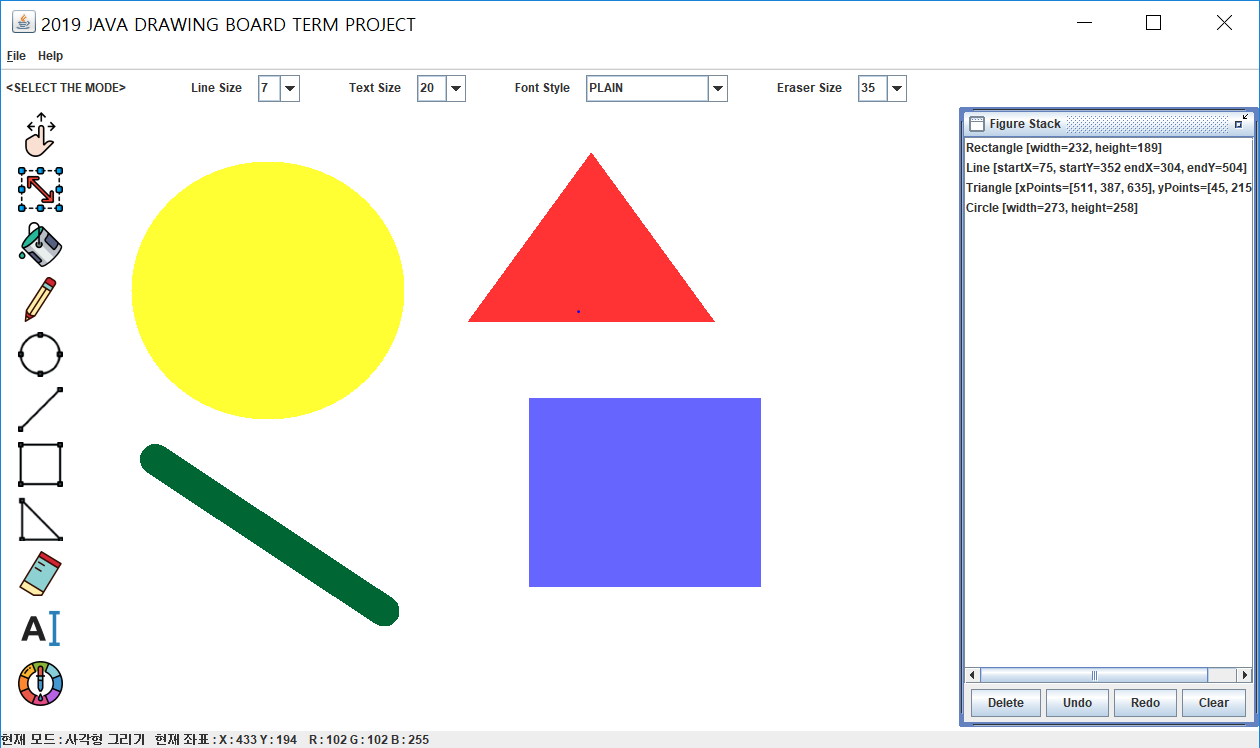
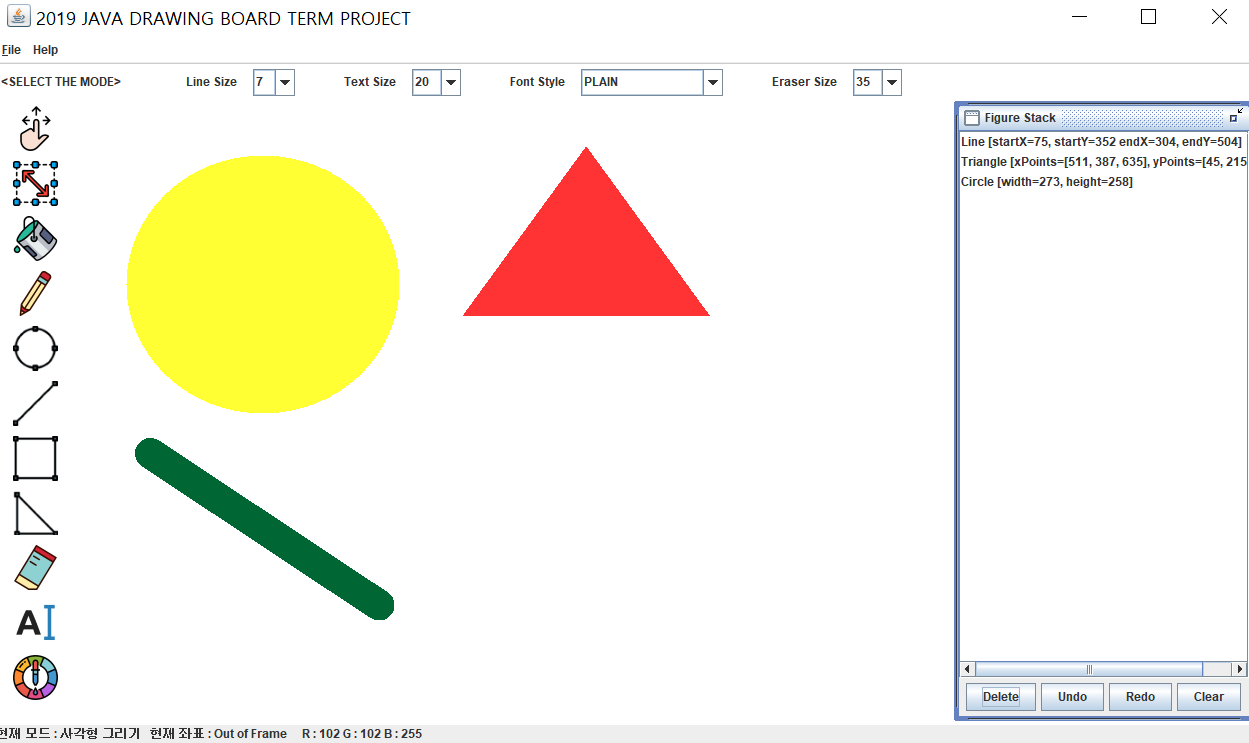
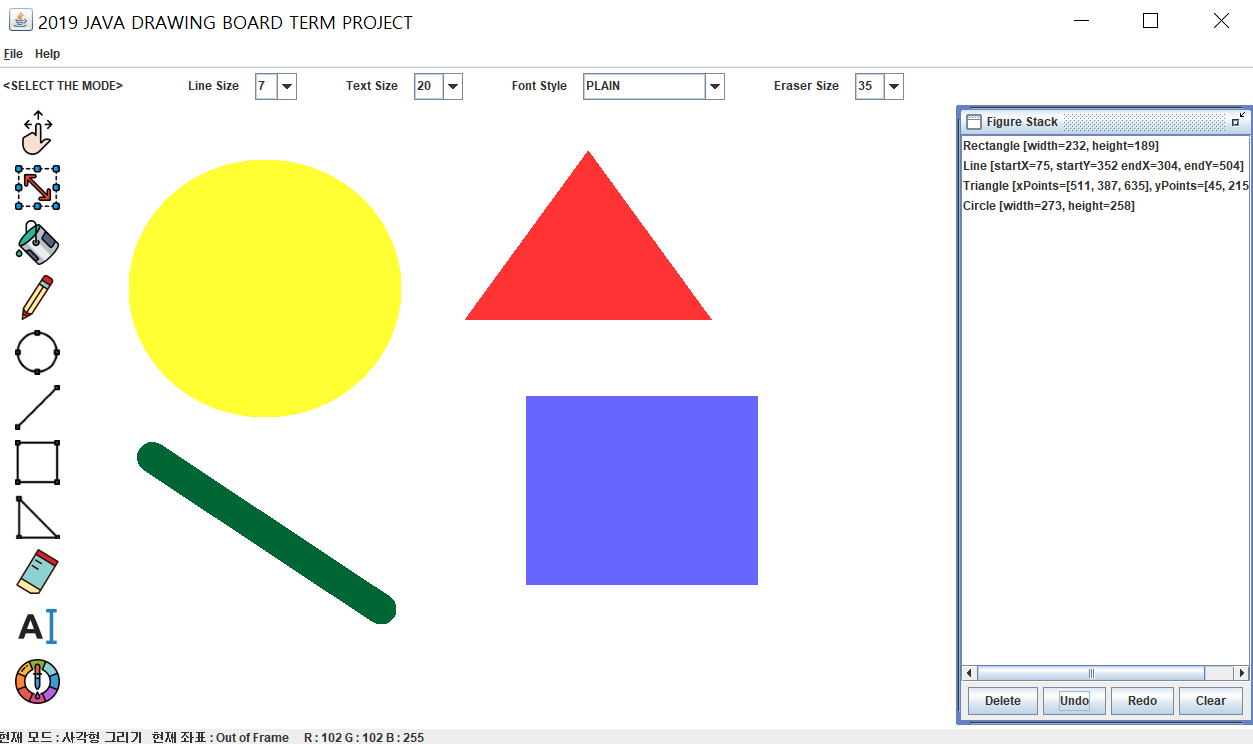


Figure Stack의 list에서 Rectangle 항목을 선택하여 delete를 누른 결과 사각형이 지워졌다.

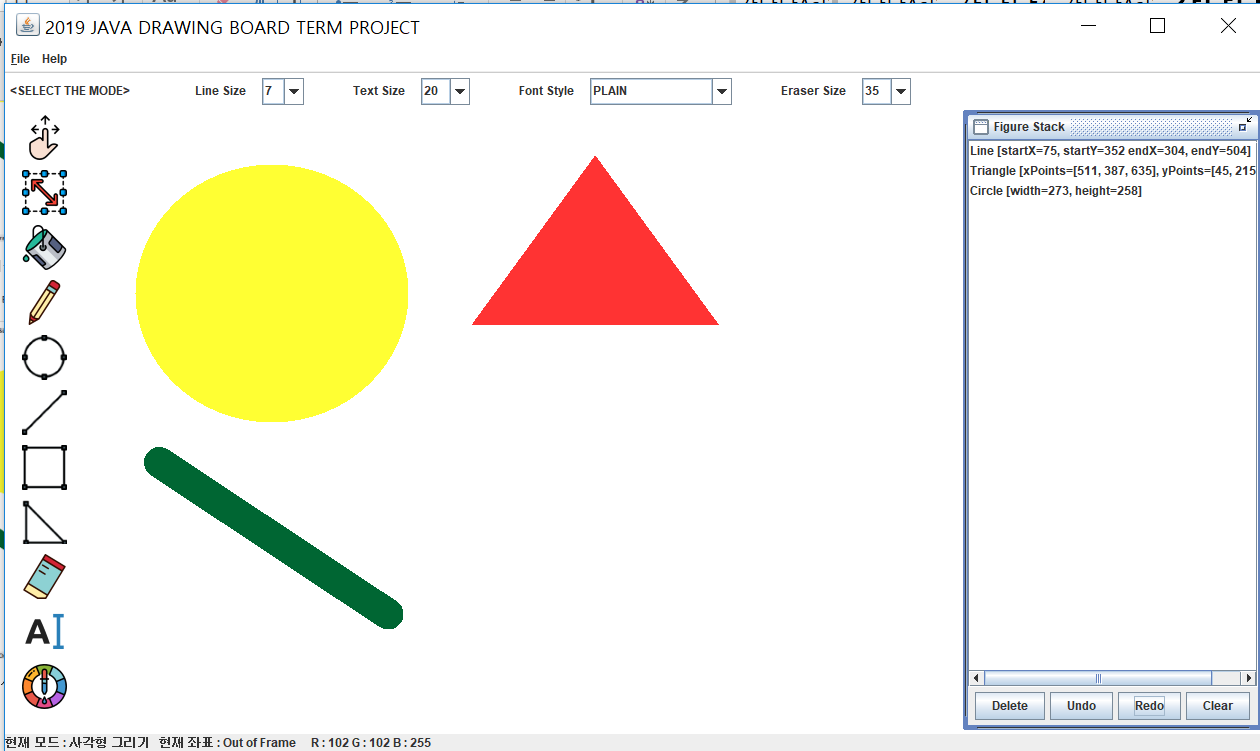




Undo를 누르자 지웠던 사각형을 다시 복구하였다.

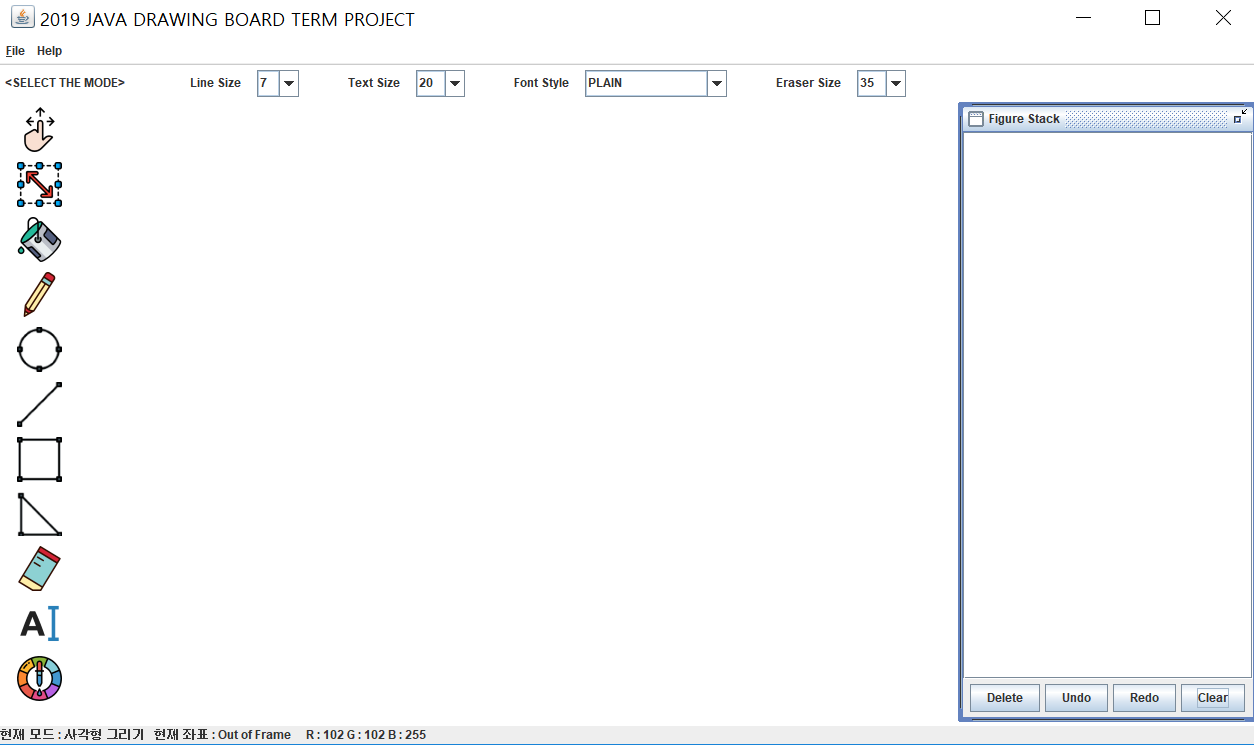


Redo를 누르면 다시 사각형이 지워진다.





Clear를 누르면 stack의 모든 항목들을 삭제하고 빈 화면으로 초기화한다.





**개인별 기여 내용**

**2018125018 김 지 원**

-그림판 메뉴에 ‘Text Size’라는 메뉴와 ‘Font Style’이라는 메뉴아이템을 만들었고, Text Size는 콤보박스를 사용하여 다양하게 조절할 수 있게 하였다.

Font Style은 PLAIN, BOLD, ITALIC, .ROMAN\_BASELINE, .CENTER\_BASELINE, HANGING\_BASELINE의 스타일 종류를 콤보박스를 이용해서 구현하였다.

콤보박스에 입력된 내용을 저장하여 Information에 전달하고, Text클래스에 전달하여 적용하였다.

-그림판 메뉴의 ‘Line Size’라는 메뉴를 만들었다.

콤보박스를 이용해서 다양한 크기의 사이즈로 조절할 수 있게 하였다.

이도 Text 메뉴와 마찬가지로 mouse listener를 이용해서 콤보박스의 내용을 저장하고 Information 클래스로 전달한다. 그리고, 이를 Pen 클래스에 전달해서 효과를 적용하여 그릴 수 있도록 구현하였다.

- PenType클래스를 생성하여 커서의 처음과 나중 위치를 저장할 수 있게 하였고, 커서정보를 필요로 하는 클래스에 상속해주는 역할을 할 수 있게 하였다.

pen클래스는 PenType을 상속받아 받아온 커서 정보에 Information클래스를 통해 펜 사이즈에 접근하고, setStroke 메소드를 통해 펜 사이즈를 조절하였다.

-DrawPanel 클래스를 통해 마우스의 드래그에 접근해서 자유 드로잉을 구현하였다.

-텍스트입력 부분에서 Pen타입을 상속받는 과정은 펜으로 그리기 과정과 같다.

PenType을 상속받은 Text 클래스를 만들어서, PenType에서 받아온 커서의 위치를, Information 클래스에서 사이즈, 폰트 정보에 접근해서, 효과를 적용한다. 그리고 setStroke 메소드를 이용해서 펜사이즈를 조절한다.

-DrawPanel 클래스에서 마우스로 패널을 클릭할 시에 텍스트 입력 다이얼로그를 출력하고,

입력 받은 문자열을 설정된 폰트, 사이즈에 맞게 출력하도록 구현하였다.

**2017126053 성 은 비**

**-** Figure package에 eraser class를 만들고 생성자로 현재 x값과 y값을 가져온다. 이때 PenType class를 상속받기 때문에 super를 사용한다. 기본값은 information class의 eraser size를 따르게 한다. 또한 크기를 설정하고 리턴할 때 필요한 get, set 메소드를 만든다. 현재의 x, y좌표를 받으면 하얀색으로 설정된 라인을 그려서 지우개 기능을 수행한다. 즉 setColor를 통해 255, 255, 255를 매개변수로 주어 하얀색으로 맞추고 drawLine을 통해 직선을 그리게 한다. 마지막으로 eraser 객체가 선택되었을 것을 명시하는 toStirng() 메소드로 eraser를 리턴한다.

- 지우개 굵기는 information class에서 기본값으로 설정되어 있는데 이때 TopPanel에서 JComboBox를 이용해 각 값을 채워 넣고 선택된 인덱스에 따라 값을 변경하게 만드는 ActionListner를 구현한다. 구현한 JComboBox는 TopPanel에 추가해서 마무리한다.

- color palette를 사용할 수 있게 import를 해주고 subFrame package의 LeftPanel에 해당 이미지를 선택하면 color palette 창이 뜨게 만들고 선택된 색상의 RGB값을 가져와서 이용한다. 즉 information class에 있는 current color를 color palette에서 선택한 색상의 RGB 각각의 값을 setColor의 매개변수로 가져와 변경한다.

-도형 상태 저장 부분은 DrawPanel 클래스를 생성하여 구현하였다. DrawPanel 클래스에는 각각의 도형객체를 figureSet 벡터를 이용해서 저장한 후, stack에 개체를 그릴 때마다 push하여 현재 그림상태를 기억할 수 있게 하였다. delete버튼을 선택했을 경우에는 최근에 그렸던 도형이 pop되도록 하였고, ‘undo’를 구성하기 위해 기존의 stack외에 따로 cancle stack을 구성하여 저장하였다.



**2016125088 심 수 현**

- 도형의 이동과 사이즈 조절 기능은 모두 마우스 callback 메소드를 사용하여 구현하였고 각각 함수에 Information에서 현재 모드를 받아와 각 모드 별로 이벤트를 정의하여 구현하였다.

도형이동은 추상 클래스인 Figure 클래스에서 moveTo를 작성한 후 mouseDragged 메소드에서 현재 좌표 값을 받아와 repaint하였다.

사이즈 조절 부분도 도형이동과 비슷하게 구현하였는데 Figure클래스에서 calcFigure 함수를 작성한 후, 각각 개체를 그리는 클래스에서 override하여 setSize 하도록 구현하였다.

- 그림 파일에 대한 입출력 코드를 작성하였다. JMenuBar를 상속받은 TopMenu class에 File메뉴를 만들었고, Actionevent를 통해서 버튼 형식으로 파일 입출력이 실행되도록 하였다. JFileChooser를 사용하여 파일을 가져오고 FileOutputStream 클래스로 파일을 생성한다.

파일을 저장하는 것은 ObjectOutputStream으로 저장할 데이터를 직렬화해서 저장하도록 하였다.

파일을 가져오는 것은 FileInputStream으로 입력 받은 파일을 OjectInputStream을 사용함으로써 직렬화 했던 것을 되돌리도록 했다. 그리고 readOject()으로 파일을 읽고 화면에 출력하게 하였다.

- addMouseListener를 사용하여 mouseEntered 이벤트의 경우에 changeCursor 메소드를 사용하여 현재 그리기 모드에 알맞은 커서 모양이 나오도록 하였다. 그리기 모드는 addListener를 사용하여 이미지로 된 버튼을 클릭하면 그 모드에 따른 이벤트를 받을 수 있게 하였다. 우측의 stackframe은 delete, undo, redo, clear의 4개의 버튼을 가지고 있는데, addListenSelectionListener로 각각의 이벤트를 받을 수 있게 하였다.

**2016125050 이 덕 희**

-추상클래스 figure package의 추상클래스 Figure Class의 추상 메서드moveTo(**int** curX,**int** curY), drawFigure(Graphics2D g)를 넣어서 도형을 드래그 해서 그리는 기본 틀을 만들었다.

-Pen을 구현하기 위해서 Figure Class를 상속받은 PenType에 ArrayList<PenTypeTracePoint> points라는 변수를 선언하여 기본적인 좌표를 저장할 공간을 만들고 PenType의정보를 Pen Class에 넘겨주어 펜을 가지고 그림을 그리는 것을 구현하였다.

-여러가지 도형을 그리는 것을 구현하기 위해 Figure Class를 상속받은 RecType Class를 만들어 도형을 그리기 위한 기본 객체를 생성해 놓은 후에 Circle Class와 Rectagle Class가 RecType Class 정보를 이용하여 drawRect, drawOval, fillRect, fillOval, setColor 등의 메서드를 이용해 원하는 객체를 그리고 색을 칠하고 색을 변경할 수 있게 구현했다.

여기서 setColor메서드에 들어가는 색깔은 information package에 저장된 값을 이용하는데 이는 사용자가 그림판에서 색을 고르면 그 색깔이 Figure Class에 넘어가게 되고 또 그 색깔이 각각의 그림을 그리는 객체에 currentcolor에 저장되에 setColor(currentcolor)가 사용자가 현재 선택한 색을 고를 수 있게 구현했다.

선을 그리는 Line Class와 삼각형을 그리는 Triangle Class는 Figure Class만 상속을 받아서 각자의 인스턴스 변수를 만들어 함수내부에서 직접 좌표를 구해서 그림을 그리는 기능을 했고 색을 채우고 선의 색을 바꾸는 것은 위와 동일하게 구현했다.

