객체 지향 설계 기말 과제

Animation Diary

20151012 최현준

목차

1. 프로그램을 만들게 된 동기

나는 예전부터 그림 일기나 내 생각을 표현해야 하는 그림 같은 경우에서 한 장의 그림으로 나의 생각을 표현해야 한다는 사실이 너무 아쉬웠었다. 이러한 나의 아쉬운 점을 해결하기 위한 방식으로 애니메이션을 그리고, 보일 수 있는 그림 일기를 만들게 되었다.

또한 애니메이션을 만들 수 있는 프로그램을 찾아 보다 보니 포토샵 같은 크고 무거운 프로그램에서만 만들 수 있었었다. 나는 사용하기 쉽고 간편하게 애니메이션을 만들 수 있는 프로그램을 만들고 싶어서 이러한 프로그램을 계획하게 되었다.

2. 프로그램의 목적

2.1 그림 일기의 목적

이 프로그램은 기본적으로 그림 일기를 그리는데 도움을 주기 위해서 애니메이션을 지원하는 프로그램이다. 따라서 프로그램은 그림을 그릴 수 있는 Tool 들을 제공 해야 하며, 동시에 애니메이션을 만드는데 도움을 주어야 한다. 이때 일기라는 점에서 글자도 적을 수 있어야한다. 그림 일기로써 여러 가지 템플릿을 제공하여도 괜찮을 것 같다.

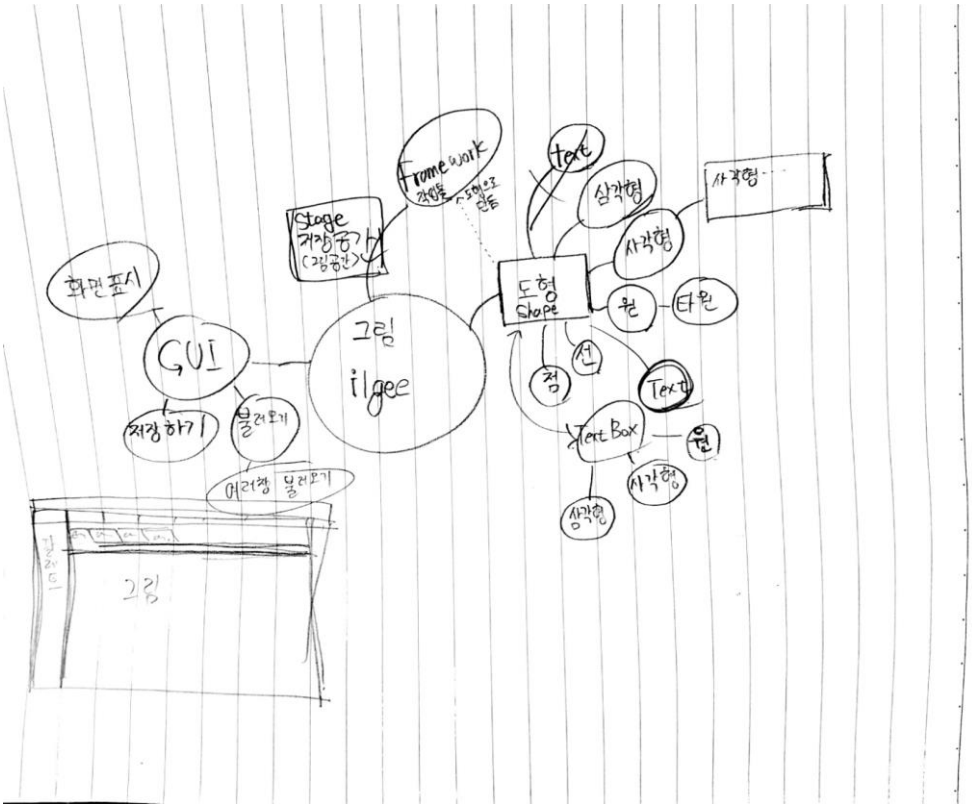
2.2 애니메이션 제작의 목적

위에서도 이야기했듯, 애니메이션을 자신이 직접 제작하는 프로그램은 기껏해야 포토샵 정도이다. 그러나 포토샵에서 지원하는 애니메이션은 복잡한 단계를 걸치고, 세세한 부분까지 설정하고, 어려운 조작을 지나서야 비로서 애니메이션이 만들어지게 된다.

나는 쉽고 빠르고 간편한 애니메이션 제작 도구로써 이 프로그램이 사용되길 원한다. 또한 자신의 생각을 자유롭게 표현 할 수 있도록 다양한 그림 도구들을 제공 할 계획이다.

3. 프로젝트 설계 및 설계에서의 수정

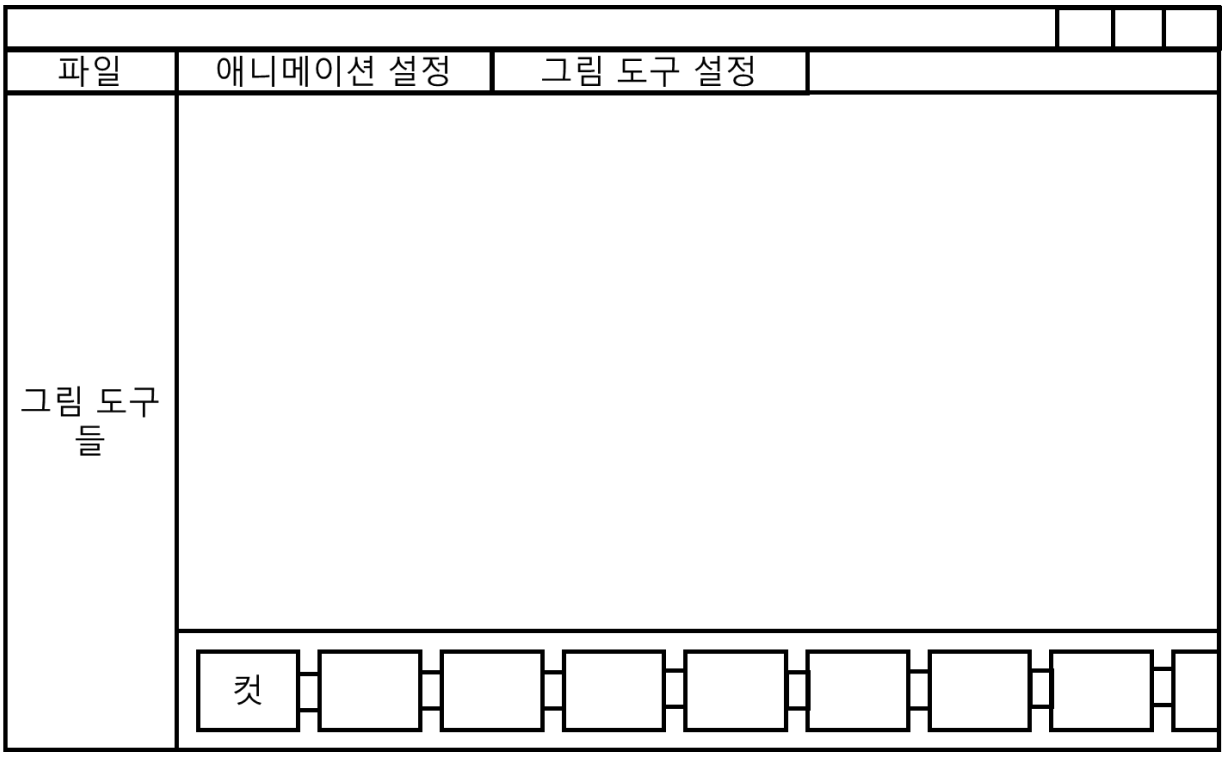
3.1 프로젝트 설계 : 프로그램 마인드맵



먼저 그림 일기 프로그램에서 중요한 것들이 무엇인가 생각 해 보았다. M(model) 과 VC (View, Control) 을 묶어서 각각 생각했다. 먼저 M (model) 들로 그려질 수 있는 도형들이 있다. 템플릿, 도형들, 펜 등이 이에 속한다. 그 외에 VC (View, Control) 로 GUI와 그 GUI 의 버튼들에 들어있는 Action Listener, Action Event 들이 있다

또한 TextBox를 따로 구현하여 글을 그릴 수 있는 도형과 그냥 도형을 구분하도록 생각했었다. 또한 TextBox 뿐만 아니라 그냥 Text 를 그릴 수 있도록 생각했다. 전체적인 조작 방식을 MS Power Point 에서 지원하는 것처럼 생각했다.

3.2 프로젝트 설계 : 프로그램 GUI



전체적인 GUI는 다음과 같이 표현 될 것이다. 그림을 그릴 수 있는 도구들은 왼쪽에, 컷을 표현하는 칸들은 아래에 표현되어 장면에 이동에 따라 보여지는 것이 달라진다. 파일 탭에서는 파일의 저장과 관련된 모든 기능들이 (저장, 불러오기, 다른 이름으로 저장, 실행 파일로 저장 등이 있다.) 있으며, 애니메이션 설정에서는 애니메이션의 빠르기, 애니메이션에서 진행 방식 등을 설정 할 수 있다. 또한 그림 도구 설정에서는 그림의 투명 배경 등을 설정 할 수 있도록 한다.

3.3 프로젝트 설계 : 프로그램 기능 생각

필수 기능

- 그림과 글이 동시에 나와야 한다.
- 저장 및 불러오기가 가능해야 한다.

생각한 추가 기능

- PPT 처럼 미리 템플릿을 만들어서 작성 가능하도록 한다.
- 그림판과는 다르게 각각의 요소를 선택 가능하도록 만든다.
- 슬라이드를 이용해서 여러 장의 그림을 하단에 보이게 한다.
- 드래그 앤 드랍 방식으로 이미지를 추가, 불러오기를 지원하도록 한다.
- 그림을 연속해서 시간 차를 두고 보여주어 애니메이션을 진행한다.
- 애니메이션에 대한 정보를 따로 클래스를 만들어서 담도록 한다.
- 애니메이션 제작의 간편화를 위해 UI를 최소화 한다.
- 빠른 프로그램 실행을 위해 실행 모드와 제작 모드를 나눈다.

3.4 프로그램 설계 : 실제 설계 방식

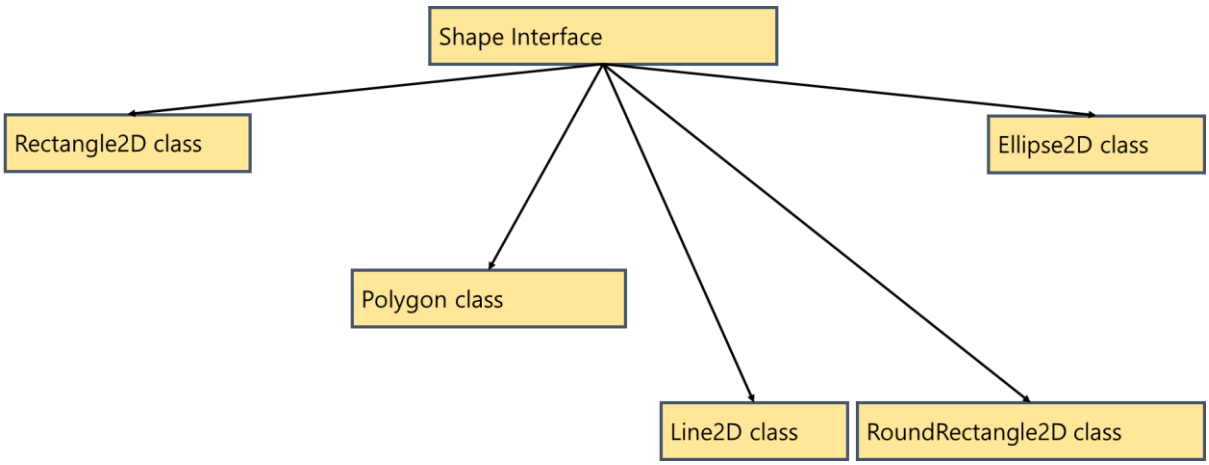
나는 원래 만들어져 있던 JAVA 에서의 여러 API 와 여러 Class들을 사용하고 싶었다. 그래서 Graphics2D 와 Graphics, 또한

geom 패키지를 참고하여 프로그램을 설계하였다.

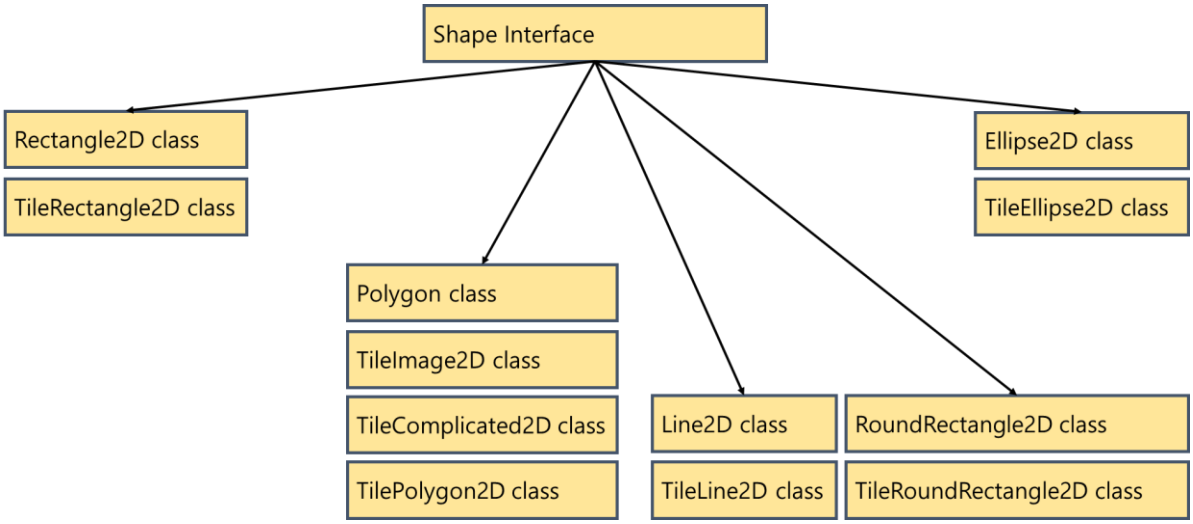
그러나 원래 Geom 패키지의 도형들은 실제 도형을 정의하기만 하는 클래스들 이어서, 나는 따로 이 도형들을 하나로 묶어주면서 동시에 Draw 메소드를 구현 할 수 있도록 해야했다.

1. 원래 Shape Interface 를 상속하면서
2. 원래 도형 (Rectangle2D class) 를 extends 하고
3. Serializable, clonable 을 implement 하면서
4. Draw 메소드를 지원하고, 또한 하나로 묶어낼 수 있어야 한다.
   (공통된 하나의 클래스를 상속해야 한다.)

위의 조건을 만족하는 설계 방식을 그림으로 표현해보면 다음과 같다.

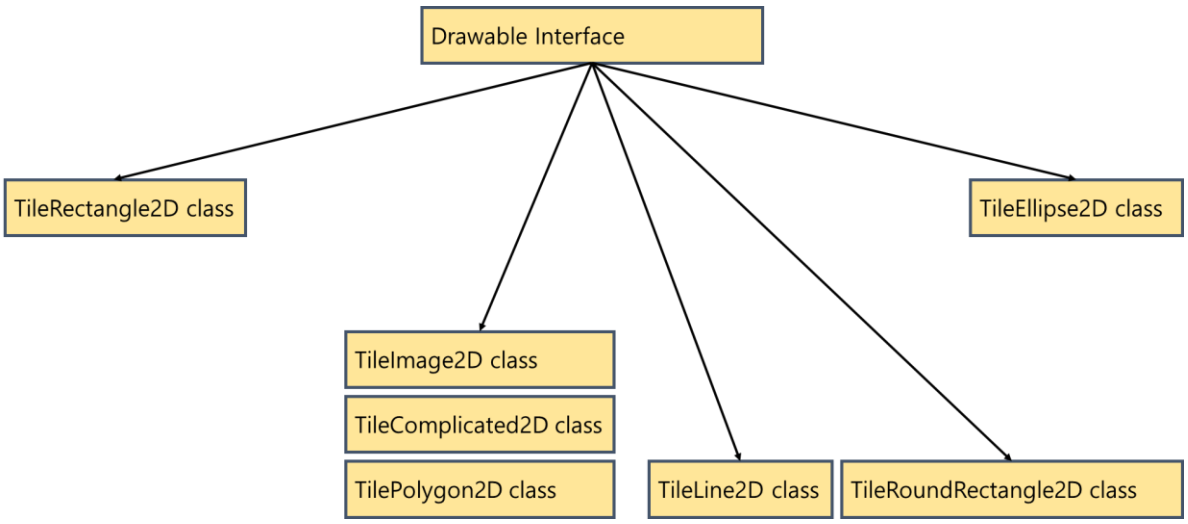


< Shape Interface를 상속받는 원래의 Class들 >



< Shape Interface를 상속받고 원래의 Class들을 상속 받는 Class >

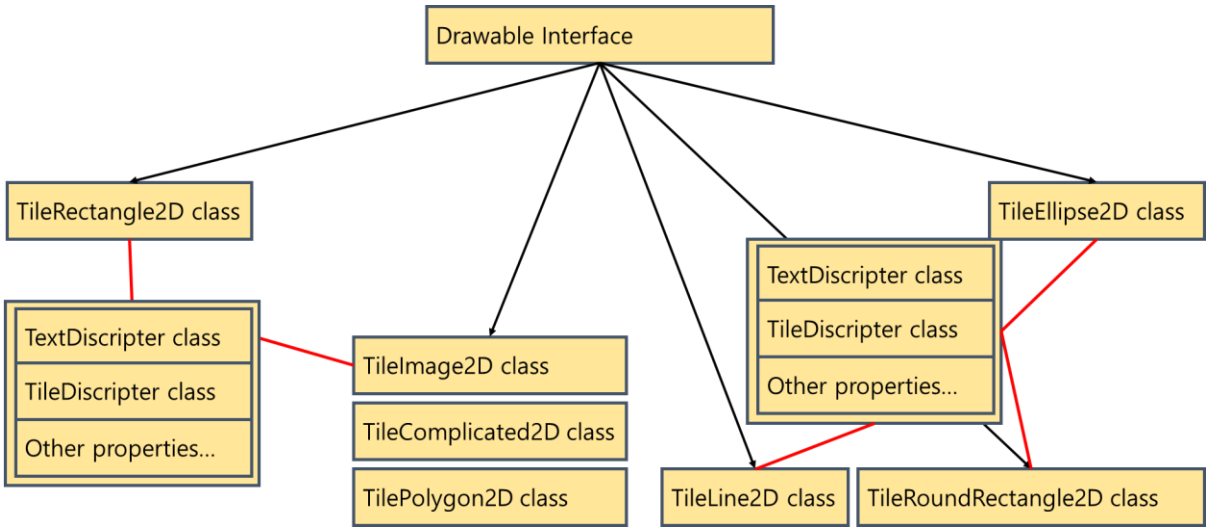
따라서 나는 여기서 4번 조건을 위해서 Drawable Interface 를 정의하고 이에 따라서 Drawable Interface 에 draw(Graphics g) 메소드와 그 외 등등 메소드를 정의해 주었다.



< Drawable Interface를 상속받고 원래의 Class들을 상속 받는 Class >

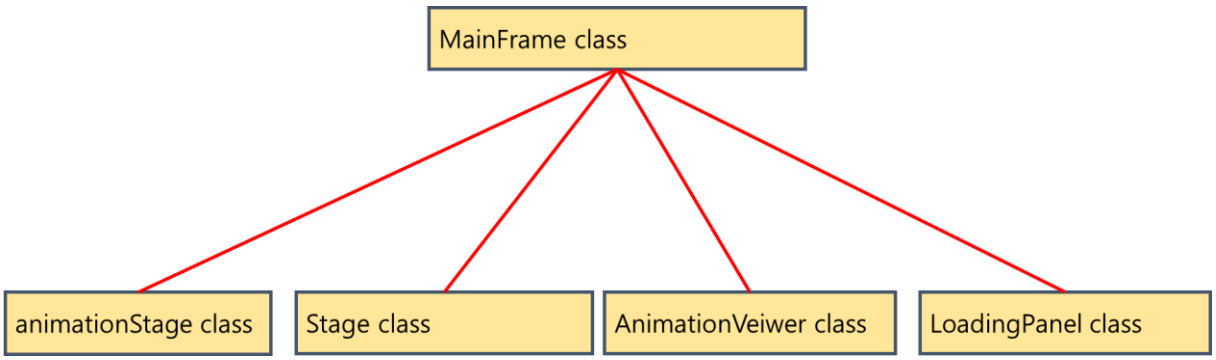
또한 그림에 관련된 정보들 (각도, 변형된 요소 등...) 은 TileDiscripter 클래스에서 관리하고, 글자에 관련된 정보들 (글자 색,

글자의 내부 위치 등...) 은 TextDiscripter 클래스에서 관리하게 설계 했다.



< Discripter 들이 추가된 클래스 설계 >

- 검은 선은 상속 관계를 의미한다.
- 빨간 선은 포함 관계를 의미한다.



< Veiw 에 관련된 설계 >

MainFrame 에서는 그림을 그리고 애니메이션을 만드는 역할을 진행하며, 이를 AnimationViewer 클래스에서 보여주도록 한다.

LoadingPanel 은 AnimationViewer 클래스의 아래에서 얼마나 진행되

었는가 진행 된 상태를 보여주는 Panel이다.

Stage 클래스는 현재 그리고 있는, 혹은 그려진 여러 상태들을 기억하는 클래스이다.

4. 프로그램 구현 및 시행착오

4.1 해결한 어려운 점

1. 이미 만들어져 있던 Polygon2D 등의 클래스를 상속 받으면서도 Tile이라는 이름으로 하나로 묶고 싶었던 설계상 문제

- Drawable interface로 묶어 줌으로써 해결.

2. Heap 공간 부족 문제 해결

- BufferedImage 를 여러 장 만들어서 화면을 구성하기 보다는 draw 를 여러 번 하는 것으로 화면을 구성하여 BufferedImage를 줄여서 Ram 용량을 확보 함으로써 해결.

3. BufferedImage로 저장한 이미지 애니메이션으로 출력 시 화면 깜빡임 문제 해결

- BufferedImage를 저장하면서 화면이 깨지므로 BufferedImage를 포

기하고 Drawable ArrayList를 이용해서 저장함으로써 문제 해결.

4. Clonable Implements 에서 clone() 메소드의 return 값이 Object에서 바뀌지 않던 문제 해결

- clone() 재정의로 문제 해결

5. 클릭 한 곳에 어떤 타일이 있는가 문제 해결

- 클릭 한 곳에 무엇인가 있는가? 라는 메소드를 위해 contain(x, y) 메소드를 Drawable Interface에 정의.

- 모든 Tile Class들에서 재정의로 Polygon2D.contain(x, y) 혹은 타원의 공식 등을 사용했다.

- 클릭 하 곳의 타일의 종류를 나타내는 int getTileShape() 라는 클래스도 Drawable 인터페이스에 정의해서 형 변환에 도움을 주었다.

6. 회전 시 자신 Tile을 중심으로 한 것이 아니라 화면의 왼쪽 위인 0, 0 을 기준으로 회전하는 문제

- 이때 0, 0으로 회전 시키고 중심이 이동 한 만큼 각 점을 원래의 곳으로 이동 시킴.

7. ArrayList.size() 까지 for문을 돌리면서 i 번째 것을 지우는 것에서의 오류

ArrayList al 이 있다고 할 때,

```
For( int i = some_number; i < al.size(); i++) {
        al.remove( i );
}
```

- 이렇게 되면 remove 되면서 size가 변하기 때문에 모두 삭제되지 않는다.

8. ArrayList 값 복사와 레퍼런스 복사 문제 해결

- ArrayList 끼리 set 하고 get 하다 보니 변하지 않길 바라던 원래의 값이 바뀌던 경우가 있었다. Clone() 함수를 이용해서 해결했다.

9. Paint 가 재정의 되어야하는 Panel (AnimationViewer, LoadingPanel 등..) 문제

- 각각의 Panel 을 상속 받은 Class에서 paint(Graphics g) 를 재정의 하여 해결했다.

10. 저장 후 불러오기에서 이미지가 저장 된 곳의 위치가 변하면 불러오지 못하는 문제 해결

- 파일의 위치를 가지고 불러오면 불러오지 못하던 문제를 처음 연 파일을 불러와서 BufferedImage로 파일을 가지고 있도록 하여 해결하였다.

4.2 해결 못한 어려운 점

1. 이미지 저장 시 BufferedImage 가 직렬화 되지 않던 문제 발생

- BufferedImage 는 serializable implements 하지 못해서 직렬화 되지 못해서 여전히 사진을 저장하지 못한다.

2. 네트워크 서버 IP 가 0.0.0.0 : 8000 으로 뜨던 문제

- 0.0.0.0 이 wild card 로써 자신을 나타낸다는 의미를 이해하지 못함.

3. 다른 프로젝트와 결합하려 했엇으나 시간이 부족

- http://codegiraffe.iptime.org₩Savefile 에 파일을 올리고 보고 저장할 수 있도록 하려 했으나 시간이 부족하여 진행하지 못했습니다..

4. 기능이 구현되지 않은 버튼 (Stroke Size)

- 간단한 기능이지만 아직 구현 못한 버튼들이 존재합니다.

Ex) GIF 로 저장, 서버에 저장, 애니메이션 상세 설정 같은 버튼들..

5. MainFrame 클래스에서 View 와 Control의 혼재.

- MainFrame에 직접 MouseListener 를 구현하다 보니 코드가 너무 길어지는 문제 발생. 후에 Control을 따로 분리해 낼 계획입니다.

- Data 접근성에서 문제가 있어서 그랬었습니다.

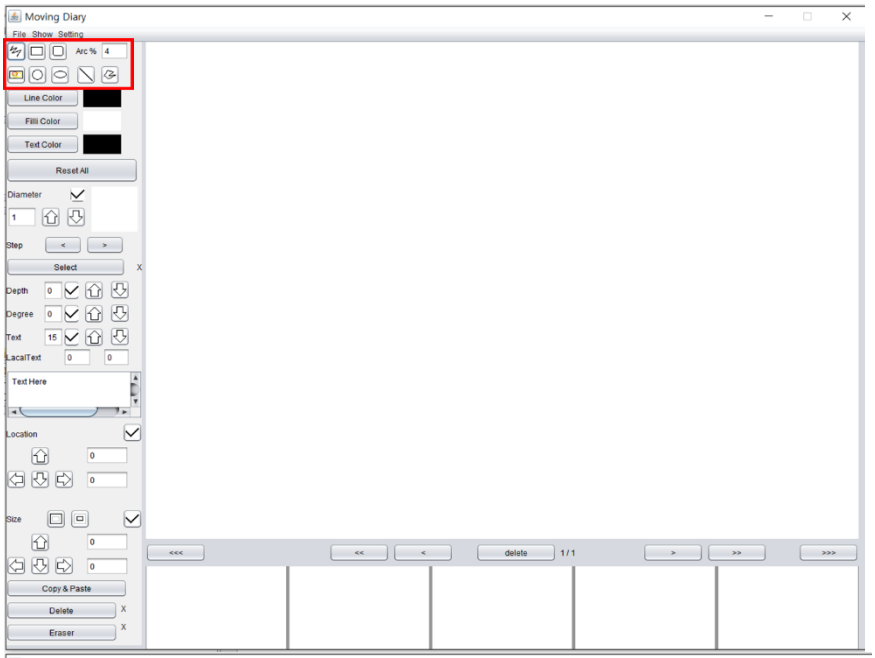
4.3 그 외의 구현에서 특이 사항.

PPT에서 발표한 자료 참고.

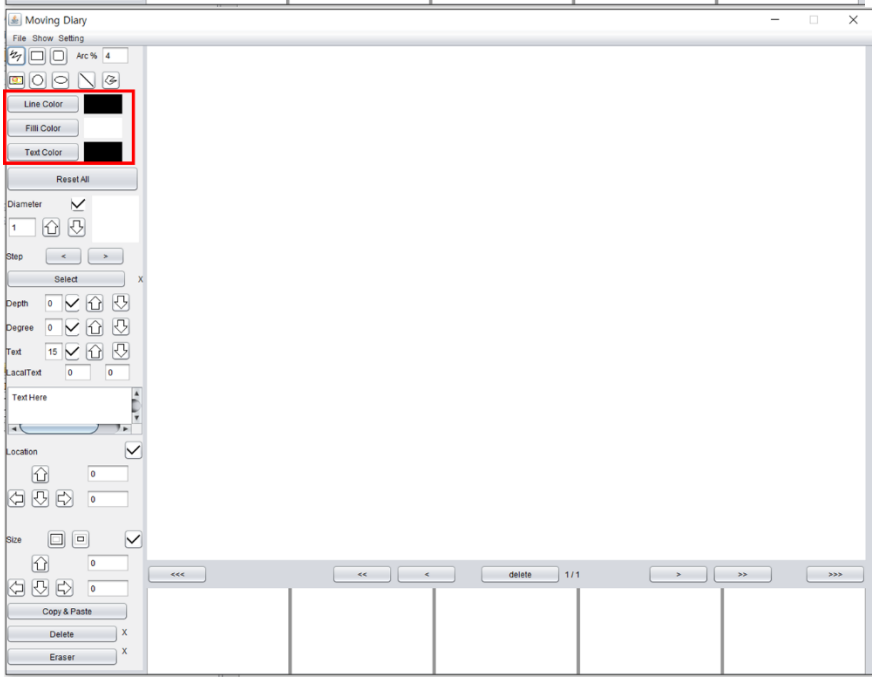
- Drawable Interface에 있는 모든 함수들은 하나하나 벽에 막힐 때마다 한 개씩 추가한 메소드들 이라서 설계를 유지하는 것에 어려움이 있었지만, 잘 해결해나갔다.

5. 프로그램 결과, 사용법 및 정리

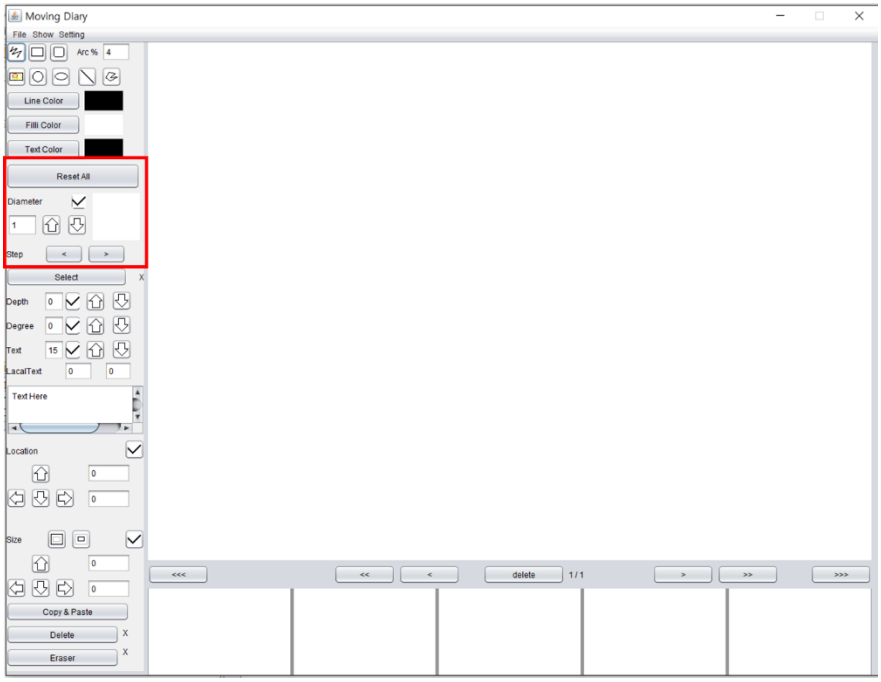
5.1 프로그램 결과 화면 및 버튼 정리



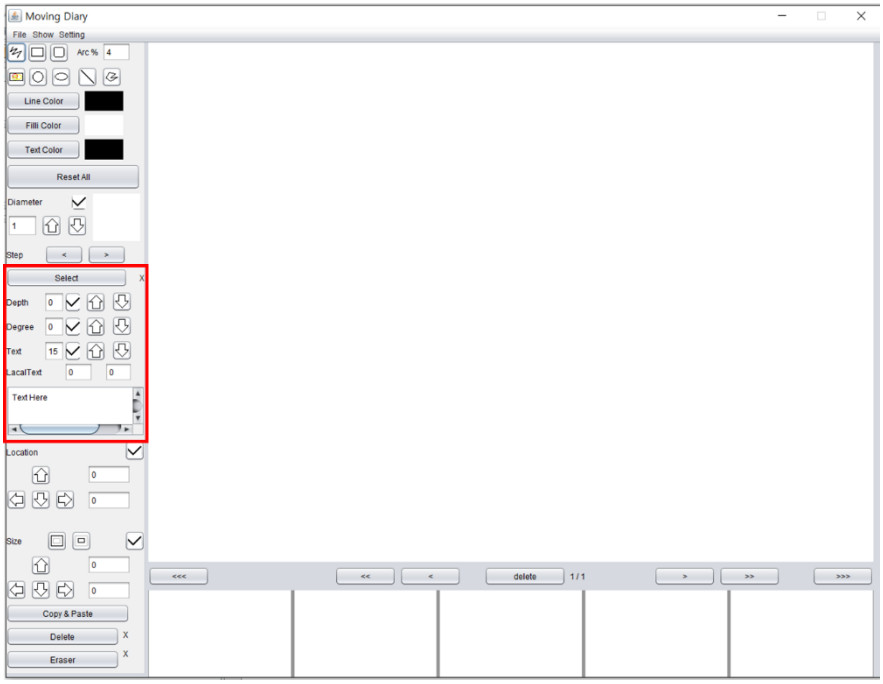
그리기 버튼, 각각 자유 곡선, 사각형, 둥근 사각형, 그림 추가하기, 원, 타원선, 다각형 버튼이다. %는 둥근 사각형의 곡률이다.



각각 외곽선 색, 내부 색, 글자 색을 나타내는 버튼과 색들이다.



모두 지우기 버튼과 팬의 크기를 조절하는 버튼, Step < >은 한 단계 전 후 이동이다.

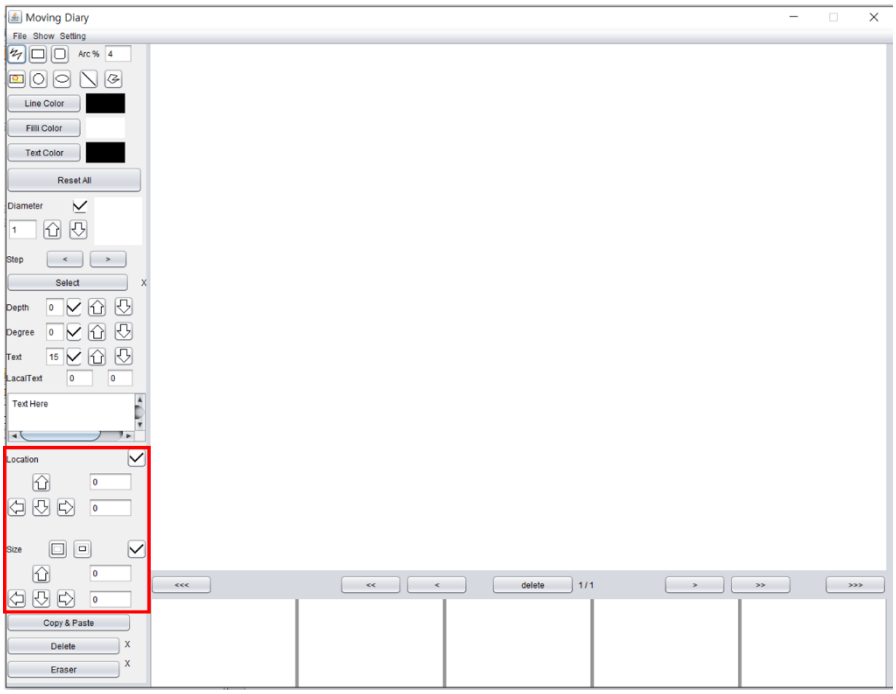


Select 버튼은 이미 그려져 있는 객체를 조절할 때 쓰인다.

Select 버튼을 누르면 선택 상태로 들어가게 되어 객체를 클릭 할 수 있게 된다. 또한 Depth 는 그려지는 순서이며, Degree는 Tile을 기울일 때 쓰는 각도, Text 는 글자의 크기 이다. Local Text 는 각각 Text 가 그려지는 X, Y 값이다. 아래에 Text Here 에는 집어넣고 싶은 글자를 입력하면 된다.

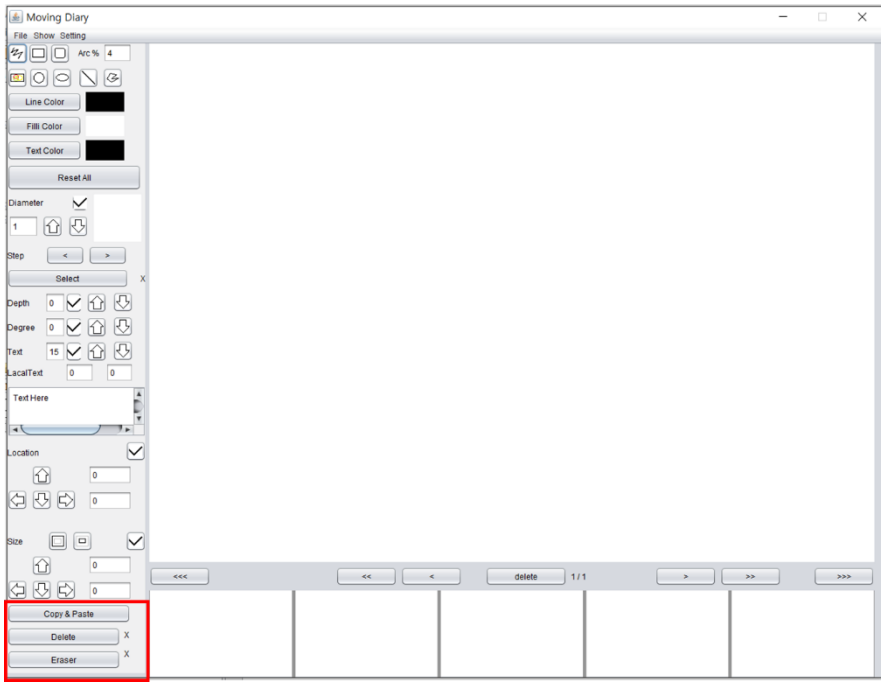
각각의 요소에는 V 모양 버튼이 있는데, 요소를 변경하고 V 를 눌러

서 저장할 수 있다.

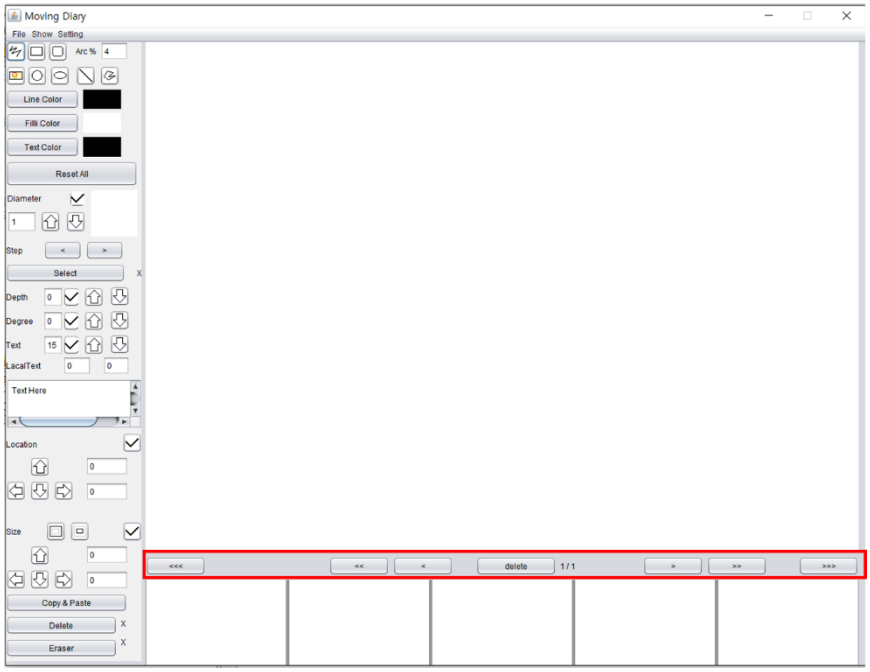


Location은 Tile의 왼쪽 위 모서리가 가리키고 있는 지점이다. 각각 화살표를 이용해 5 씩 조절 할 수 있으며, V 를 눌러 직접 바꾼 값을 저장할 수도 있다.

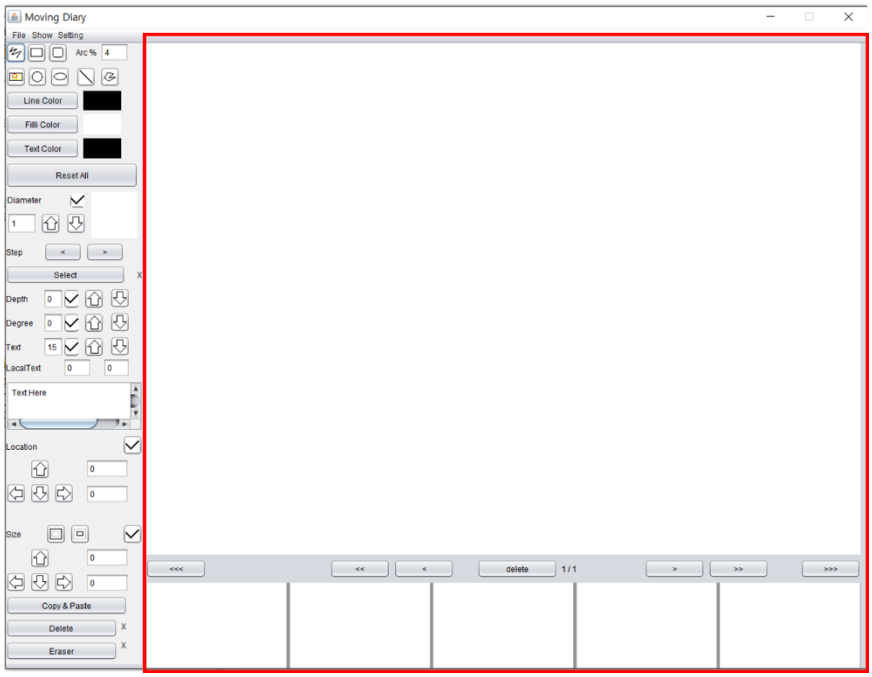
Size 는 Tile의 크기를 의미하며 각각 Width 와 Height 를 글자로 보여준다. 글자를 바꾸고 V 를 눌러 저장하거나 역시 화살표를 눌러서 크기를 바꿀 수 있다. 위에 버튼은 크게 할 것인가 작게 할 것인가를 나타내는 버튼이다.



복사 붙여넣기, 삭제 모드 진입, 지우개 모드 진입 관련 버튼들이다.



10장 뒤로, 5장 뒤로, 1장 뒤로, 현재 슬라이드 삭제, 1장 앞으로, 5장 앞으로, 10장 앞으로 이동하는 버튼들이다. 한 장씩 앞뒤로 이동 시 없는 칸이 나오게 되면 새롭게 칸을 만들고 이동한다. 5장이나 10장씩 옮길 때 없는 칸으로 가게 되면 오류 창이 뜬다.

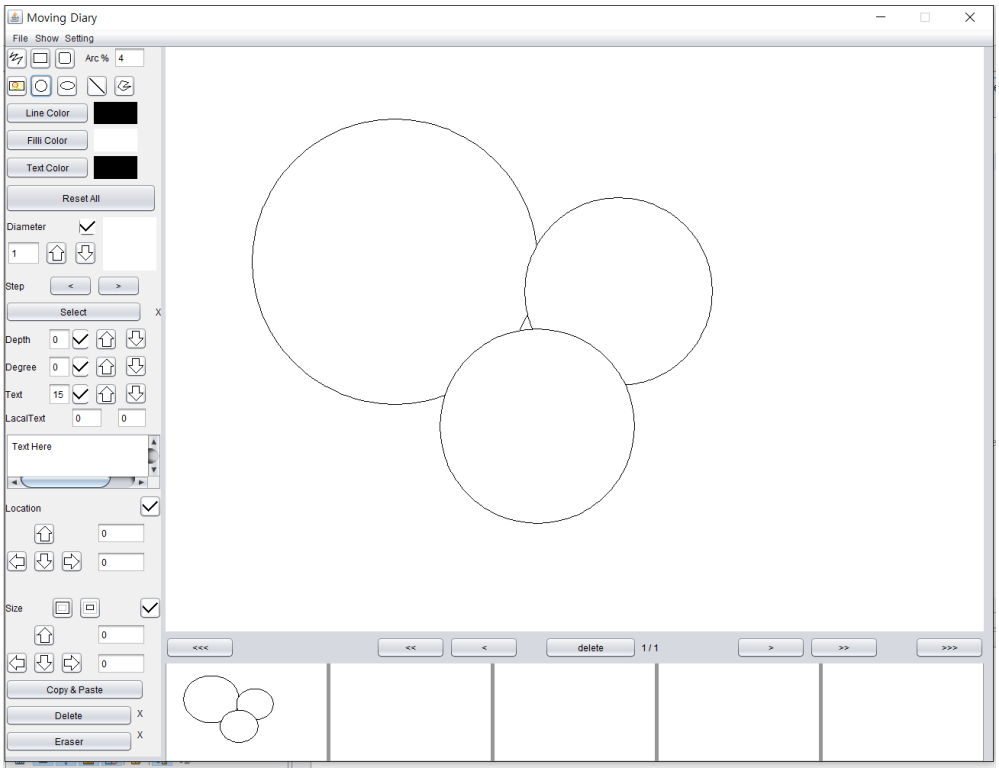


위의 하얀 칸에는 그림을 그릴 수 있는 그림판이다. 아래의 5개의 칸은 애니메이션 슬라이드 이며, 항상 가운데가 현재 슬라이드를 나타내는 칸이다. 생성되지 않은 슬라이드는 하얀 칸으로 표현되며, 현재 슬라이드를 나타내는 번호는 delete 옆쪽에 존재한다.

Show 탭의 play this file 로 현재 만든 파일을 보여주게 하면 start, rotation, stop 버튼이 나오는데, start 로 시작하고, rotation 으로 1번만 진행 할 것인지 무한 반복 할 것인지, stop으로 멈출 것인지를 결정한다. 이때 초록색의 진행 상태바가 얼마나 진행 되었는 지를 알려준다.

5.2 사용 방법 정리

1. 그림을 그립니다. 이때 원, 사각형, 자유 곡선 등 자유롭게 그려도 애니메이션을 그릴 수 있습니다. 주의 할 사항은 타일을 선택 할 때 Select 모드를 들어가야만 선택 할 수 있는 것 입니다. 타일의 색을 바꾸고 싶으면 타일을 선택 한 뒤 Color 를 누르고 Accept 버튼을 눌러야 합니다.

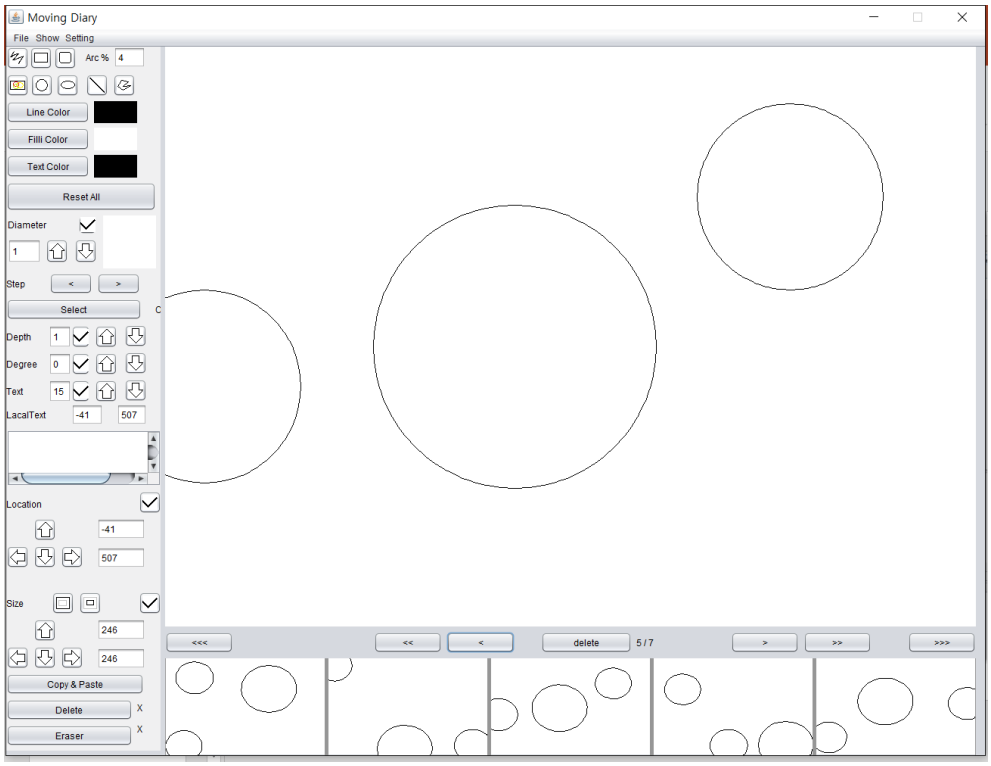


< 원 객체 3개를 생성하였습니다. >

2. Scene 들을 구성합니다. > 나 < 버튼을 눌러서 왼쪽 오른쪽에 화면을 추가합니다. 이 때 Depth 가 같은 Tile은 원 – 원, 혹은 사각형 – 사각형 같이 Tile 의 종류가 같아야 합니다. (원은 타원의 일종이라 상관 없음) 그렇지 않다면 Animation 구성에서 에러가 나게 됩니다.

애니메이션에서 변하는 것은 위치, 크기, 색, 글자 크기, 글자 위치입니다.

또한 Tile 의 종류가 같다면 Depth가 달라지더라도 상관 없습니다.

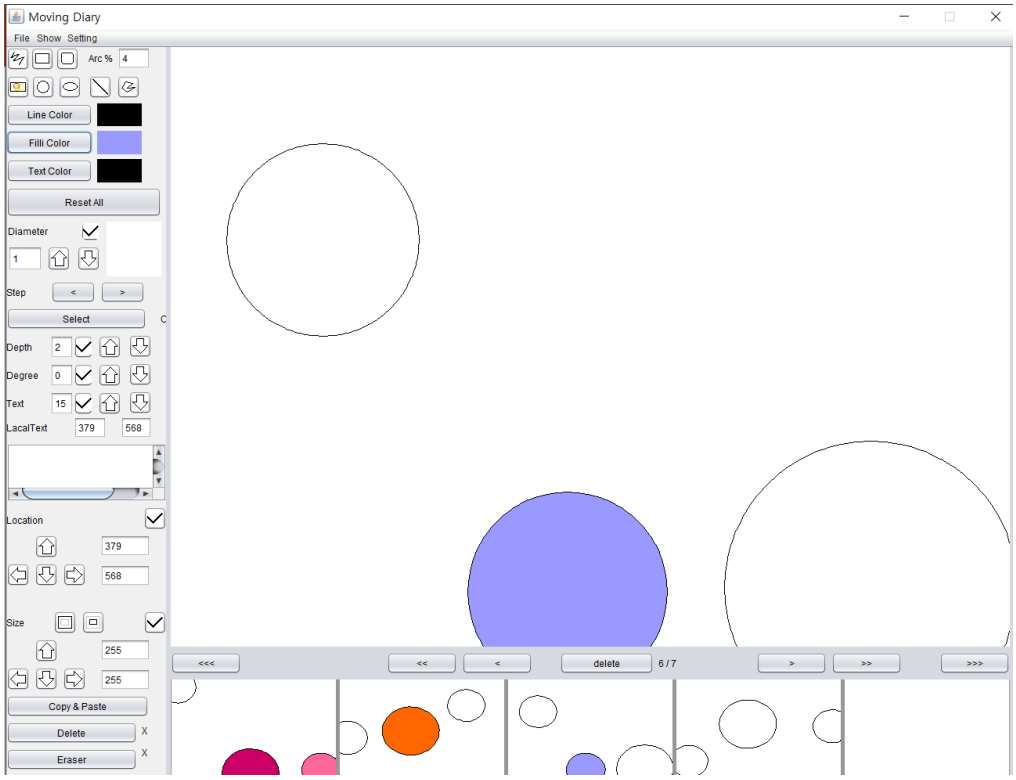


< 원 객체 3개의 위치를 이동시키며 Scene을 구성했습니다. >

3. Scene의 위치는 Delete 버튼 오른쪽에 [현재 위치] / [전체 크기] 로 표현됩니다. Scene 을 삭제하고 싶다면 Delete 버튼을 누르면 됩니다.

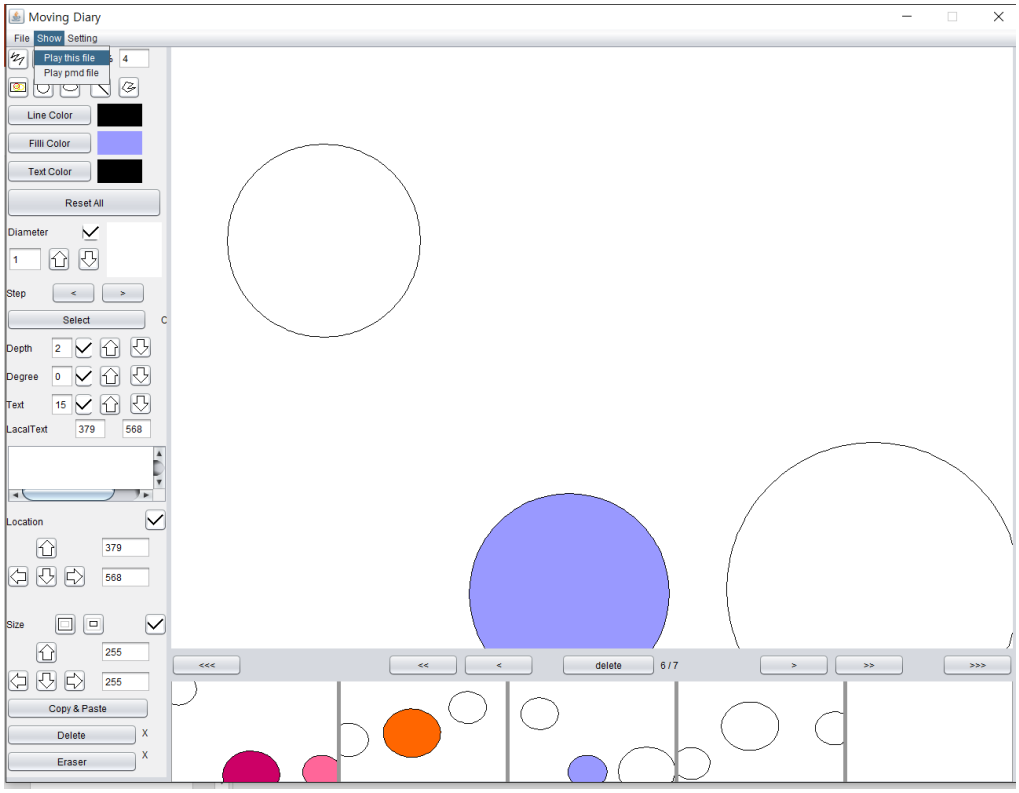
또한 Tile의 개수가 늘어나는 것은 괜찮지만 Tile의 개수가 줄어들게 된다면 오류가 납니다. 이러한 오류 사항들은 후에 고칠 예정입니다.

이미지를 추가해도 괜찮습니다. 이미지의 위치, 이미지의 크기가 변하더라도 애니메이션이 진행 될 수 있습니다.



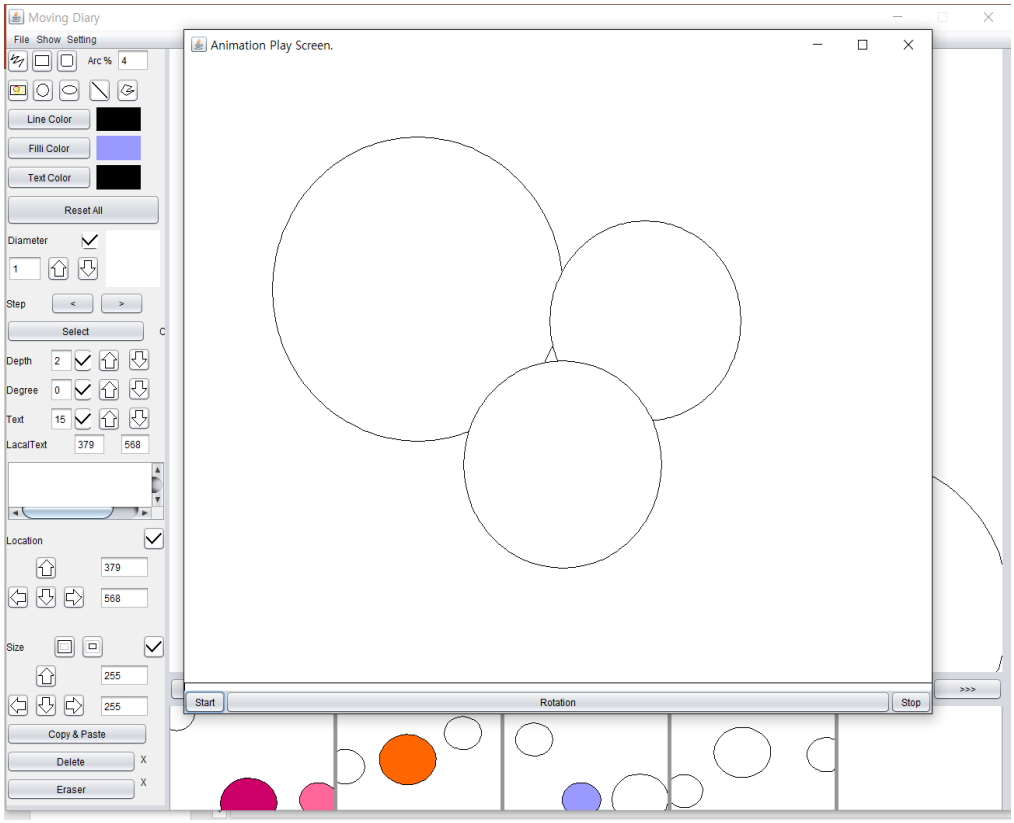
< 원 객체 3개의 색을 변화시켜 보았습니다. >

4. 이제 Show 텝에 있는 Play this file 버튼을 눌러서 진행합니다. Animation의 길이가 많이 길다면 (40 ~ 50장 정도) 5초 정도의 시간이 걸리게 됩니다. 또한 이미지가 들어간 애니메이션은 20장 이상 생성하게 된다면 메모리가 부족하게 됩니다. 이러한 점에 유의해서 애니메이션을 만들어야 합니다.



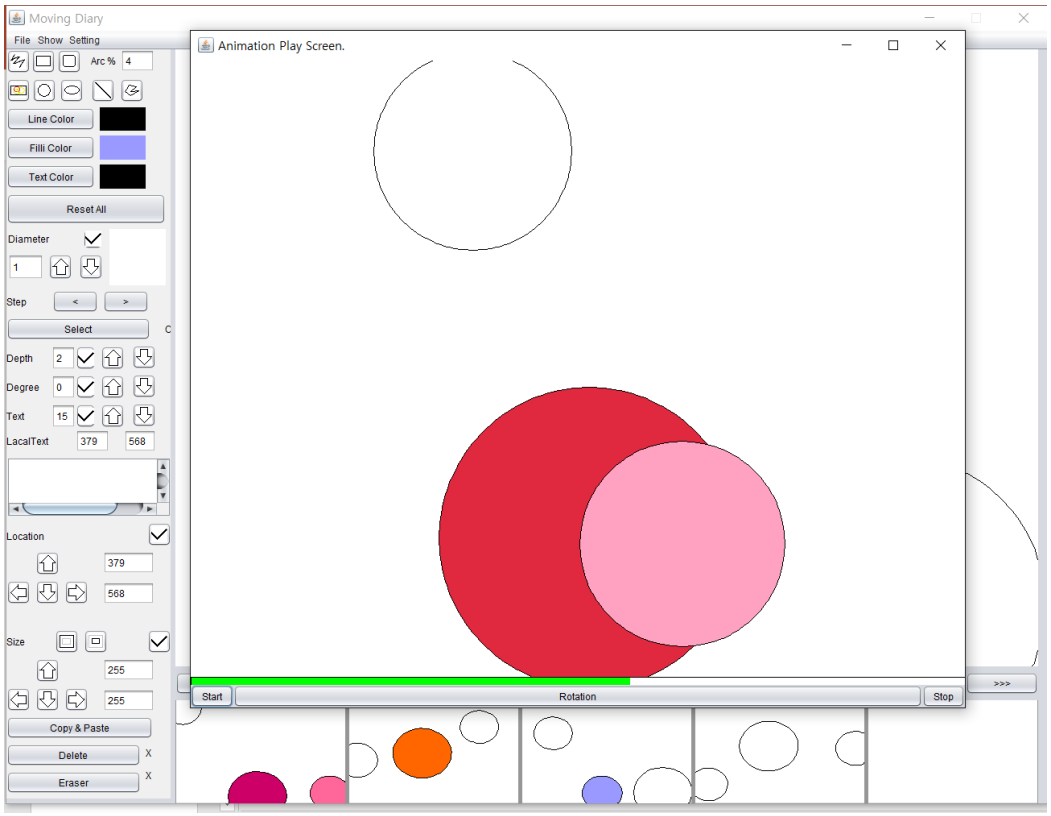
< Play Playable file 탭은 pmd 형식으로 저장된 파일을 실행합니다. 이는 속도가 더 빠릅니다. >

5. Animation Play Screen (Animation Viewer) 가 뜨면 잘 생성 된 것입니다. 혹시 뜨지 않는다면 이미지가 포함된 20장 이상의 애니메이션인지, 혹은 180여장 이상의 애니메이션인지, 혹은 다른 종류의 Tile 이 같은 Depth를 가지고 있는지 (클릭을 통해서 알 수 있습니다) 확인 해 보아야 합니다. 이제 Start 버튼을 눌러서 실행 할 수 있습니다.



< 애니메이션이 잘 완성 된 모습 >

6. 잘 실행이 된다면 아래에 초록색으로 현재 진행 상태를 나타내어 주는 바가 이동하게 됩니다. 한번만 진행 하고 싶으면 Rotation을, 중간에 멈추고 싶으면 Stop 을 누르면 됩니다.



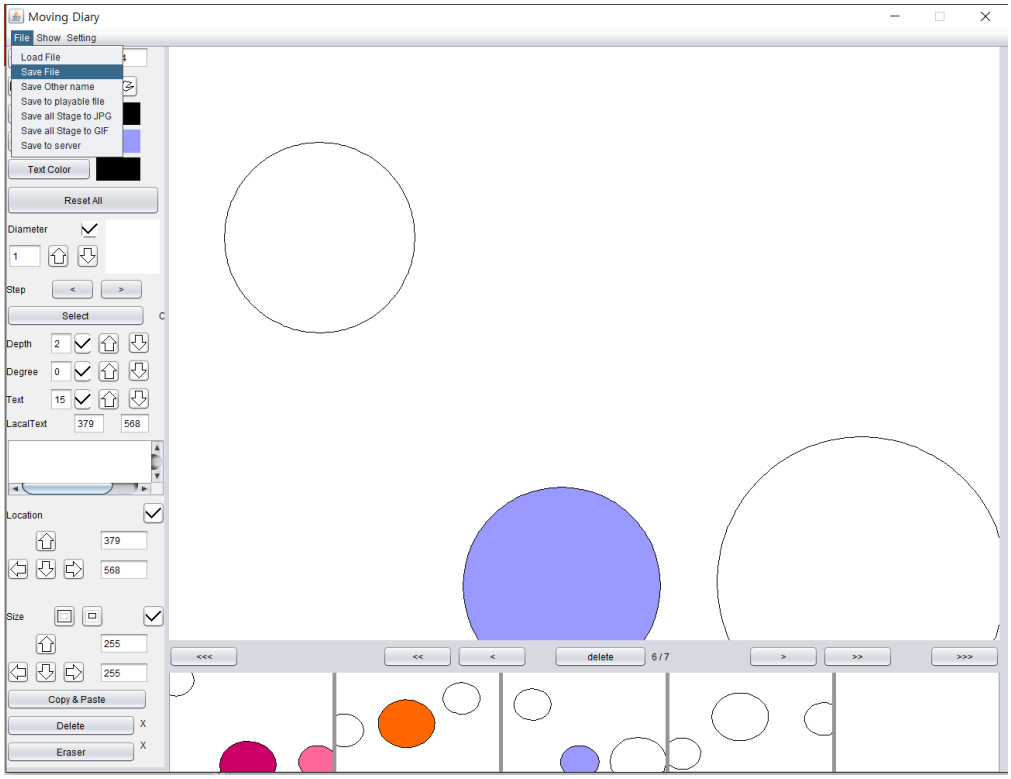
< Stop 으로 중간에 멈춘 애니메이션 >

7. 파일을 만들었으면 저장 해야 합니다. 이미지가 포함 된 애니메이션 같은 경우는 BufferedImage의 문제 때문에 저장 할 수 없습니다. 그러나 이미지가 포함된 애니메이션도 JPG 로 저장 할 수 있습니다.

Save File 버튼을 누르면 .md 파일의 확장자로 저장 할 수 있습니다. 이는 현재 Scene 들의 정보를 불러 올 수 있는 파일입니다.

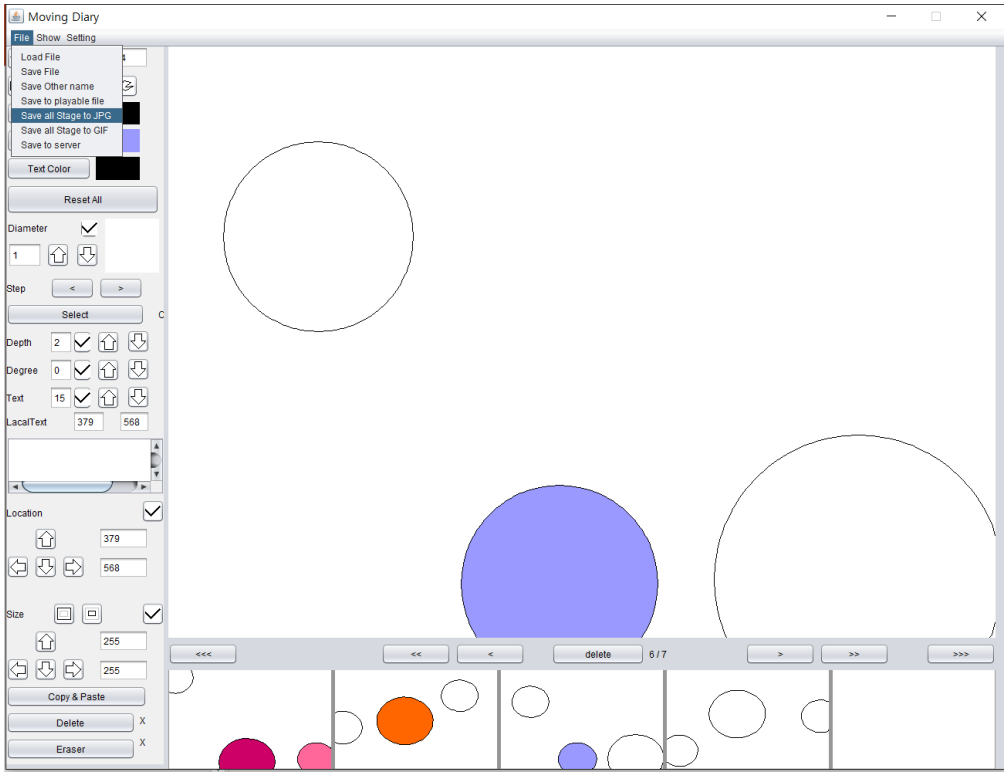
Save Other Name 은 다른 이름으로 저장 하는 탭 입니다. Save to Playable file 은 현재 Scene 을 애니메이션에 필요한 정보로 바꾸어 저장하여 후에 바로 연산 없이 바로 재생 할 수 있도록 하는 .pmd 파일로 저장하게 됩니다.

Save to JPG는 모든 Scene 을 JPG 파일 형식으로 저장합니다. Save to GIF 기능은 아직 구현하지 못했습니다. Save to Server 기능도 아직 구현하지 못했습니다.

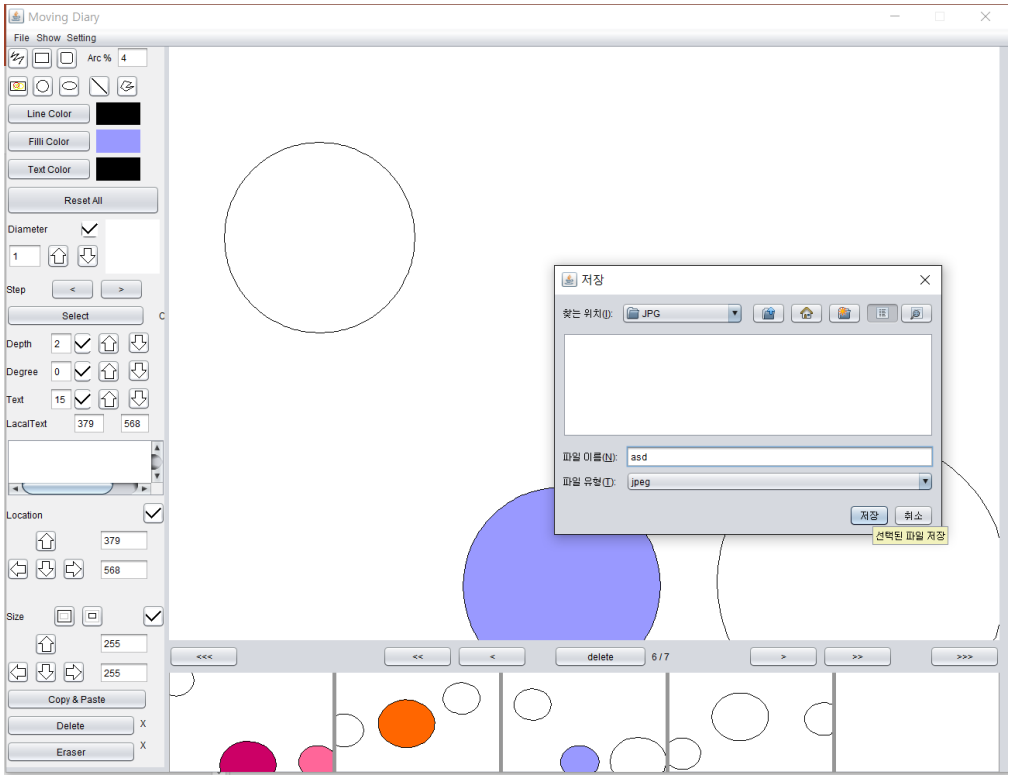


< save file >

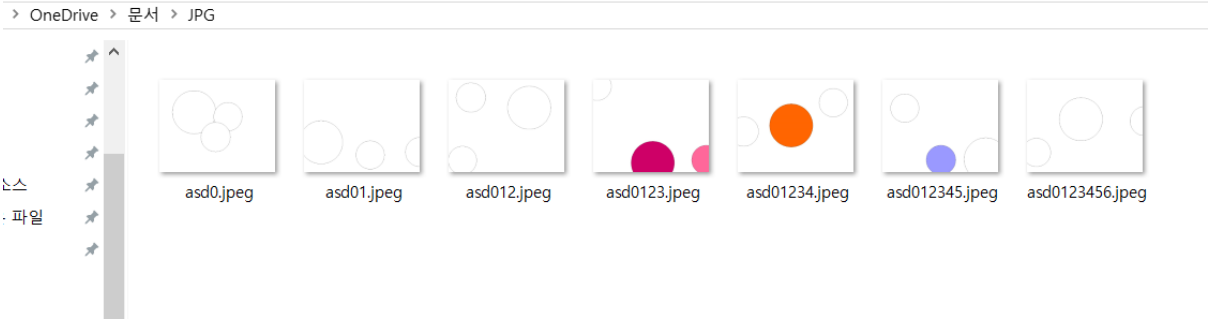
8. JPG로 바꾸어 저장하게 되면 확장성 있게 자신이 쓴 일기를 확인할 수 있습니다. 그러나 현재 폴더에 바로 JPG를 모든 Scene을 바꾸어 저장하기 때문에 새로운 폴더를 만들어서 저장하기를 권장합니다.



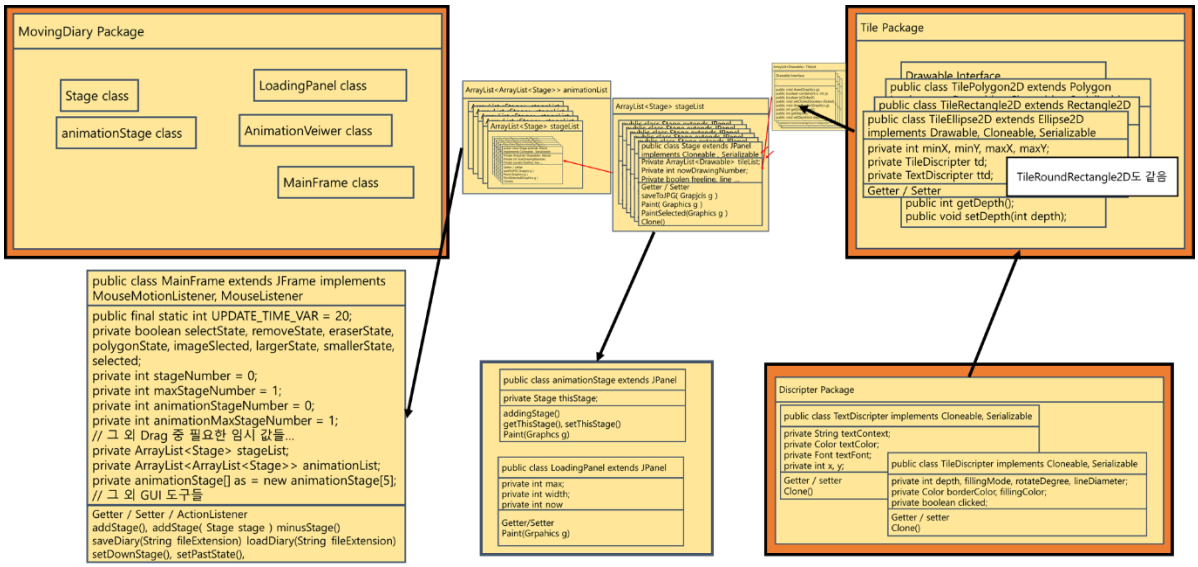
9. 새로운 폴더에 저장하는 모습입니다. .md 파일이나 .pmd 파일도 이와 같은 방식으로 저장하게 됩니다.



10. 잘 저장 된 모습입니다. 이와 같이 저장 된 것을 불러오는 기능도 추가 계획에 있습니다.



5.3 UML 정리



자세한 사항은 PPT에 참조하시면 됩니다.