<차례>

1. CloneObject 클래스

생성자

public CloneObject()

접근 제어자 : public

사용 목적 : 상속하는 클래스들의 Clone 메서드의 구현을 강제하기 위해서 사용한다.

예시 데이터 : public abstract class Figure : CloneObject, IDrawable

메소드

public virtual object Clone()

입력 : void

출력 : object – 객체의 내용을 복사한 내용을 반환한다.

사용 목적 : 상속하는 클래스들의 Clone() 재정의를 위해 virtual 메서드로 정의된다.

2. MyPoint 클래스

접근 제어자 : public

사용 목적 : 프로젝트에서 사용하는 좌표에 대한 클래스로 구현되었다. x,y좌표를 멤버 필드로 갖는다. CloneObject 클래스를 상속받아 Clone() 메서드를 재정의하고 Equals() 메서드를 재정의한다.

필드

x, y

접근 제어자 : public

타입 : int

사용 목적 : Figure의 좌표 중에서 x, y좌표를 저장하는 필드이다.

예시 데이터 : Rect temp = new Rect(p1.x, p1.y, p2.x, p2.y);

public MyPoint(int x, int y)

입력 : int, int – 순서대로 x,y좌표를 받는다.

사용 목적 : MyPoint 객체를 생성할 때 사용한다.

예시 데이터 : MyPoint point = new MyPoint(10,10);

public override object Clone()

입력 : void

출력 : object – 현재 MyPoint 객체를 새로 생성해서 리턴한다.

사용 목적 : MyPoint 객체를 복사하기 위해서 사용한다. CloneObject 클래스의 Clone 메서드를 재정의한다.

예시데이터 : MyPoint point = Point.Clone();

public override bool Equals(object obj)
입력 : object – 비교할 대상 객체
출력 : bool – 비교한 대상과 같은지 여부 참,거짓
사용 목적 : MyPoint 객체끼리 같은지 여부를 확인하기 위해서 사용한다. 각 객체의 좌표값이 같으면 true 이외의 객체의 형식이 다르거나 좌표값이 다르면 false를 리턴한다.
예시데이터 : if(Point.Equals(OtherPoint))

3. Rect 클래스

접근 제어자 : public
사용 목적 :

필드

left, top, right, bottom
접근 제어자 : private
타입 : int
사용 목적 : Rect 객체의 상하좌우 위치를 저장하기 위한 필드
예시 데이터 :

mergedRatio
접근 제어자 : private
타입 : float[]
사용 목적 : 그린 객체 여러 개를 병합했을 때 확대 축소를 위한 비율 값을 저장하는 멤버 필드
예시 데이터 :

생성자

public Rect()
public Rect(Rect other)
public Rect(MyPoint start, MyPoint end)
public Rect(int left, int top, int right, int bottom)
입력 : void – left,right,bottom,top 필드를 0으로 초기화
Rect – 입력받은 Rect의 필드값으로 현재 객체를 초기화한다.
MyPoint[] - start의 x,y좌표를 left,top, end의 x,y좌표를 right,bottom으로 저장한다.
int[] - 각 값을 left,right,top,bottom 값으로 저장한다.
사용 목적 : Rect 클래스의 생성자

public int getLeft()
public int getRight()
public int getTop()
public int getBottom()
입력 : void

출력 : int – 각 멤버 필드
사용 목적 : private으로 선언된 각 필드의 값을 얻기 위한 메서드

public getRatio(int I)
입력 : int – mergedRatio에서 얻고자 하는 필드의 인덱스
출력 : float – 병합 시 변경되는 비율 값
사용 목적 : private으로 선언된 mergedRatio의 값을 얻기 위한 메서드
public void setLeft(int x)
public void setRight(int x)
public void setTop(int x)
public void getBottom(int x)
입력 : int – 각 필드에 넣고자 하는 값
출력 : void
사용 목적 : private으로 선언된 각 필드의 값을 넣기 위한 메서드

public setRatio(int I, float x)
입력 : int – mergedRatio에 넣고자 하는 필드의 인덱스
float – 변경된 비율 값
출력 : void
사용 목적 : private으로 선언된 mergedRatio 필드에 값을 넣기 위한 메서드

public void normalizeRect()
입력 : void
출력 : void
사용 목적 : Rect가 초기화 되었을 때 left값이 right보다 크거나 top이 bottom보다 큰 경우에 좌표값이 엉킨 상태이기 때문에 이를 normalize하기 위해서 사용한다.
예시 데이터 : tempRect.normalize()

public Rect multipleRect(int x)
public Rect divideRect(int x)
입력 : int – Rect의 모든 멤버필드에 곱하거나 나눌 값
출력 : Rect – Rect객체의 모든 멤버필드에 입력받은 값이 곱하거나 나누어진 새로운 Rect 객체
사용 목적 : Rect객체를 입력받은 값만큼 곱하거나 나누어 새로운 객체를 만들기 위해서 사용된다. 우리 프로젝트에서는 사용되지 않는다.

public void inflateRect(int x, int y)
입력 : int, int – Rect를 가로 방향으로 늘릴 값과 세로 방향으로 늘릴 값
출력 : void
사용 목적 : 현재 Rect 객체의 크기를 늘리기 위해서 사용한다. 마우스로 객체를 선택할 때 클릭한 위치와 객체의 위치의 충돌 여부를 파악하기 위해서 사용된다.

예시 데이터

public deflateRect(int x, int y)
입력 : int, int – Rect를 가로 방향으로 줄일 값과 세로 방향으로 줄일 값
출력 : void
사용 목적 : 현재 Rect 객체의 크기를 줄이기 위해서 사용된다. 우리 프로젝트에서는 사용되지 않는다.
예시 데이터

public Boolean isRectEmpty()
입력 : void
출력 : Boolean – 객체가 비어있는지 여부를 리턴한다.
사용 목적 : 객체가 크기를 가지는지 여부를 확인하기 위해서 사용된다. 좌우 값이나 상하 값이 같을 경우에 true를 리턴한다.

public Boolean isHitPoint(int x, int y)
입력 : int, int – 사각형에 포함될지 확인할 x,y좌표
출력 : Boolean – 입력받은 좌표가 사각형에 포함되는 지 여부
사용 목적 : 입력으로 받는 좌표가 Rect객체의 내부에 포함되는 지를 파악하기 위해 사용한다.

public Boolean isInArea(Selection f)
입력 : Selection – 마우스 드래그로 생성한 Selection 객체
출력 : Boolean – 현재 Rect객체가 Selection 객체 안에 있는지 여부를 리턴한다.
사용 목적 : Figure가 Selection 객체의 내부에 있는지 확인하는 메서드이다. Selection 객체는 여러 객체를 박스를 묶을 때 사용되는 객체이다.

public MyPoint getSize()
입력 : void
출력 : MyPoint – 폭과 높이를 x,y 좌표로 하는 MyPoint 객체
사용 목적 : 현재 Rect객체의 폭과 높이를 구하기 위해서 사용한다. 우리 프로젝트에서는 사용하지 않는다.

public int getWidth()
입력 : void
출력 : int – Rect객체의 폭
사용 목적 : Rect객체의 폭을 구하기 위해서 사용한다.

public int getHeight()
입력 : void
출력 : int – Rect객체의 높이

사용 목적 : Rect객체의 높이를 구하기 위해서 사용한다.

public void setRect(int left, int top, int right, int bottom)
입력 : int[] - 좌우상하의 좌표값
출력 : void
사용 목적 : 현재 Rect객체에 좌우상하 값을 넣어 위치 혹은 크기를 변경하기 위해서 사용한다.

public void setRect(Rect other)
입력 : Rect – 현재 객체를 조정할 다른 Rect 객체
출력 : void
사용 목적 : 현재 Rect객체를 다른 Rect객체를 참조하여 변경한다.

public Rect intersectRect(Rect other)
입력 : Rect – 현재 Rect객체와 공통되는 영역이 있는지 체크할 Rect 객체
출력 : Rect – 현재 객체와 입력받은 객체의 공통되는 영역, 없으면 null을 리턴한다.
사용 목적 : 현재 Rect객체와 다른 Rect객체의 공통되는 영역이 있는지 체크하고 그 영역을 리턴하기 위해서 사용한다.

4. IDrawable 인터페이스

접근 제어자 : public
형식 : 인터페이스
사용 목적 : paint를 정의하는 인터페이스로 상속하기 위해서 사용한다.

void paint(Graphic g)
입력 : Graphic – paint를 할 때 사용되는 Graphics 객체
출력 : void
사용 목적 : 상속 받는 클래스에서 각 도형들을 그리기 위해서 사용된다.

5. MyArrayList 클래스

접근 제어자 : public
상속 : ArrayList
사용 목적 : 기본 자료형인 ArrayList를 상속받는 클래스이다. Clone과 Equals 메서드를 재정의하기 위해서 정의되었다.

public override object Clone()
입력 : void
출력 : object – 현재 객체를 복사한 새로운 MyArrayList 객체
사용 목적 : 현재 객체를 복사한 객체를 출력으로 받기 위해 Clone 메서드의 재정의

public override bool Equals(object obj)

입력 : object – 현재 객체와 같은지를 비교할 객체
출력 : bool – 현재 객체와 입력받은 객체가 같은지 여부
사용 목적 : 입력 받은 객체와 현재 객체가 동일한 객체인지를 확인하기 위한 Equals 메서드의 재정의 메서드이다. ArrayList에서 각 요소가 모두 같으면 true를 리턴한다.

6. Figure 클래스

접근 제어자 : public
타입 : abstract 클래스
상속 : CloneObject, IDrawable
사용 목적 : 각 도형의 기본이 되는 Figure 클래스이다.

coordinates
접근 제어자 : protected
타입 : MyArrayList
사용 목적 : Figure의 좌표를 저장하기 위해서 사용한다. 기본적으로 두 개의 좌표를 저장하고 Polyline Figure의 경우에는 많은 좌표를 저장한다.
사용 데이터 :

currentColor
접근 제어자 : protected
타입 : Color
사용 목적 : 도형의 색을 저장하기 위해서 사용된다.
사용 데이터 :

bFilled
접근 제어자 : protected
타입 : boolean
사용 목적 : 도형의 색이 채워졌는 지 여부를 저장하기 위해서 사용한다.

bSelected
접근 제어자 : private
타입 : boolean
사용 목적 : 현재 도형이 선택되었는지 여부를 저장하기 위해서 사용된다.

selectType
접근 제어자 : private
타입 : SELECTTPYE
사용 목적 : 객체의 이동 형태를 저장하기 위해서 사용된다.

생성자
Figure()

Figure(Rect rect)
Figure(MyPoint p1, MyPoint p2)
Figure(int left, int top, int right, int bottom)
Figure(MyPoint point)
Figure(int x, int y)

public Figure(Color c)
public Figure(Rect rect, Color c)
public Figure(MyPoint p1, MyPoint P2, Color c)
public Figure(int left, int top, int right, int bottom, Color c)
public Figure(MyPoint point, Color c)
public Figure(int x, int y, Color c)

public int getVectorSize()
입력 : void
출력 : int
사용 목적 : 현재 저장되어 있는 좌표의 개수를 구하기 위해서 사용한다.

public Color getCurrentColor()
입력 : void
출력 : Color
사용 목적 : 현재 설정되어 있는 색상 값을 구하기 위해서 사용한다.

public Object getVectorElementAt(int n)
입력 : int
출력 : Obejct
사용 목적 : 좌표 리스트 중에서 n번째의 좌표값을 구하기 위해서 사용한다.

public SELECTTYPE getSelectType()
입력 : void
출력 : SELECTTYPE
사용 목적 : 현재 사용 중인 선택모드를 구하기 위해서 사용한다.

public ArrayList getCorrdinatesClone()
입력 : void
출력 : ArrayList
사용 목적 : 현재의 좌표 리스트를 복사한 ArrayList를 구하기 위해서 사용한다.

public void setCurrentColor(Color c)
입력 : Color
출력 : void

사용 목적 : 현재 선택 중인 색상을 입력받은 색상으로 설정하기 위해서 사용한다.

public void selected(Boolean b)
입력 : Boolean
출력 : void
사용 목적 : 현재 객체가 선택되어 있는지 여부를 설정하기 위해서 사용한다.

public void setSelectType(SELECTTYPE type)
입력 : SELECTTYPE
출력 : void
사용 목적 : 현재 사용 중인 선택모드를 설정하기 위해서 사용한다.

public void add(Rect rect)
public void add(MyPoint point)
public void add(int x, int y)
입력 : Rect
MyPoint
int, int
출력 : void
사용 목적 : 현재 객체에 새로운 객체를 추가하기 위해서 사용한다. 주로 생성자에서 사용된다.

public Boolean isSelected()
입력 : void
출력 : Boolean
사용 목적 : 현재 객체가 선택되어 있는지 여부를 확인하기 위해서 사용한다.

public void filled(Boolean b)
입력 : Boolean
출력 : void
사용 목적 : 색 채우기 여부를 설정하기 위해서 사용한다.

public Boolean isFilled()
입력 : void
출력 : Boolean
사용 목적 : 색 채우기 여부를 확인하기 위해서 사용한다.

public abstract Boolean isInArea(Selection s)
입력 : Selection
출력 : Boolean
사용 목적 : 도형이 Selection 내부에 포함되는지를 확인하기 위해서 사용하는 메서드이다.

abstract 메서드로 선언되어서 상속받는 클래스에서 구현된다. 객체의 모든 좌표가 Selection 객체의 내부에 있으면 포함되는 것으로 판단한다.

public abstract Boolean isEmpty()
입력 : void
출력 : Boolean
사용 목적 : 객체가 만들어질 수 있는 지 여부를 확인하기 위해서 사용한다. abstract 메서드로 선언되어 상속받는 클래스에 구현된다. 사각형은 면적이 있어야하고 PolyLine은 점이 3개이상, Line은 시작 좌표와 끝이 좌표가 달라야 한다.

public abstract Boolean isHit(int x, int y)
입력 : int, int
출력 : Boolean
사용 목적 : 객체가 입력받은 x, y 좌표와 충돌하는 지 확인하기 위해서 사용한다. 주로 객체를 선택하려고 할 때 호출된다.

public abstract void move(int x, int y)
입력 : int, int
출력 : void
사용 목적 : 객체를 x, y좌표로 이동하기 위해서 사용된다. 객체가 선택된 상태로 드래그할 때 호출된다.

public abstract void paint(Graphics g)
입력 : Graphics
출력 : void
사용 목적 : 상속 받는 클래스에서 각 도형들을 그리기 위해서 사용된다. IDrawable 인터페이스에서 선언된 메서드이고 구현은 상속받는 클래스에서 이루어 진다.

public override bool Equals(object obj)
입력 : object
출력 : bool
사용 목적 : CloneObject에서 선언된 메서드이다. 입력받은 객체와 현재 객체가 일치하는지 여부를 확인하기 위해서 사용한다. 각 객체의 모든 좌표값이 같을 때 같은 것으로 생각한다.

public override object Clone()
입력 : void
출력 : object
사용 목적 : 객체를 복사하기 위해서 사용된다. 새로운 객체를 생성하고 좌표 값과 색상 값들을 새로 할당하여 새로운 객체를 만들고 그 객체를 리턴한다.

7. Document 클래스

접근 제어자 : public
사용 목적 : View에 그려지는 객체들의 정보를 저장하는 클래스이다.

isMultSel
접근 제어자 : private
타입 : bool
사용 목적 : 현재 선택 모드가 단일 선택인지 멀티 선택인지를 저장하기 위해 사용된다.

currentTool
접근 제어자 : private
타입 : FIGURE
사용 목적 : 현재 선택된 도구를 저장하기 위해 사용된다.

currentColor
접근 제어자 : private
타입 : Color
사용 목적 : 현재 선택된 색상을 저장하기 위해 사용된다.

fileName
접근 제어자 : private
타입 : string
사용 목적 : 현재 View의 이름을 저장하고 파일 저장 시 저장하는 이름이 되는 필드이다.

objList
접근 제어자 : private
타입 : MyArrayList
사용 목적 : 그려진 모든 객체들을 저장하고 있는 리스트이다.

figList
접근 제어자 : private
타입 : MyArrayList
사용 목적 : 현재 선택되거나 그려지고 있는 객체들을 저장하고 있는 리스트이다.

clipboardFigure
접근 제어자 : private
타입 : MyArrayList
사용 목적 : 객체를 복사할 때 저장되는 클립보드 리스트이다.

viewList
접근 제어자 : public

타입 : viewList
사용 목적 : 현재 켜져있는 모든 View의 객체를 저장하고 있는 리스트이다.

stack
접근 제어자 : private
타입 : Stack
사용 목적 : 실행 취소를 하기 위해서 사용한 행동들을 저장한다.

생성자
Document(Form parent)
입력 : Form – View의 부모폼으로 설정될 폼
사용 목적 : Document 클래스의 생성자이다. View를 하나 생성하고 View의 parent로 입력받은 폼을 설정한다. 생성한 View를 viewList에 저장하고 툴, 색상, 다중선택 여부를 초기화한다.

public string getFileName()
입력 : void
출력 : string
사용 목적 : getFile 필드를 얻기 위해서 사용한다.

public FIGURE getCurrentTool()
입력 : void
출력 : FIGURE
사용 목적 : 현재 사용하고 있는 도구를 구하기 위해서 사용된다.

public Color getCurrentColor()
입력 : void
출력 : Color
사용 목적 : 현재 사용하고 있는 색상을 구하기 위해서 사용된다.

public Figure getCurrentFigure()
입력 : void
출력 : Figure
사용 목적 : 현재 사용하고 있는 도형을 구하기 위해서 사용된다.

public Figure getCurrentFigure(int n)
입력 : void
출력 : Figure
사용 목적 : 다중 선택 기능으로 여러개의 객체를 선택했을 때 n번째의 현재 도형을 구하기 위해서 사용된다.

public int getObjectSize()
입력 : void
출력 : int
사용 목적 : 현재 objList의 크기를 구하기 위해서 사용된다.

public Figure getObjectElementAt(int n)
입력 : int
출력 : Figure
사용 목적 : 현재 objList에서 n번째 요소의 도형 종류를 구하기 위해서 사용된다.

public void setFileName(string fn)
입력 : string
출력 : void
사용 목적 : 현재 Document의 파일이름을 설정하기 위해서 사용된다.

public void setCurrentTool(FIGURE f)
입력 : FIGURE
출력 : void
사용 목적 : 현재 사용하고 있는 도구를 설정하기 위해서 사용된다.

public void setCurrentColor(Color c)
입력 : Color
출력 : void
사용 목적 : 현재 사용하고 있는 색상을 설정하기 위해서 사용된다.

public void setCurrentFigure(Figure f)
입력 : Figure
출력 : void
사용 목적 : 현재 선택한 도형 종류를 설정하기 위해서 사용된다.

public Figure mergeFigure()
입력 : void
출력 : Figure – 선택된 도형들을 모두 합친 mergeFigure를 리턴한다.
사용 목적 : 여러 개의 객체들을 모두 합치기 위해서 사용한다. 선택된 객체가 저장된 figList에서 객체들을 선택해서 selected 필드를 false로 변경하고 figList에서 삭제한다. 그리고 mergedFigure 객체를 새로 생성해서 선택되었던 객체들을 모두 포함하는 Figure를 생성한다.

public void setRatio(MergedFigure merged)
입력 : MergedFigure 병합된 상태에서 크기를 조정할 때 비율을 제대로 조정하기 위해서 사용

출력 : void
사용 목적 : 병합된 상태에서 각 Figure의 위치의 비율값을 각 Figure에 저장한다. 병합된 상태에서 확대 축소를 올바르게 작동시키기 위해 사용한다. 각 Figure들에 대해서 현재 MergeFigure에 대한 각 객체의 상대적인 위치를 저장해서 확대 축소 시 그 값을 사용한다.

public void setIsMultSel(bool a)
입력 : bool
출력 : void
사용 목적 : 다중 선택 여부를 설정하기 위해서 사용된다.

public bool getIsMultSel()
입력 : void
출력 : bool
사용 목적 : 현재 Document의 다중 선택 여부를 구하기 위해서 사용된다.

public void removeFigList(Figure f)
입력 : Figure
출력 : void
사용 목적 : 현재 FigList에서 f 도형을 제거하기 위해서 사용된다. 다중 선택 중에 하나의 도형을 제외할 때 사용한다.

public void addObject(Figure figure)
입력 : Figure figure
출력 : void
사용 목적 : Document의 objList에 새로운 도형을 추가하기 위해서 사용된다.

public Figure findFigureByPoint(int x, int y)
입력 : int, int
출력 : Figure
사용 목적 : x, y 좌표에 있는 도형이 무엇인지 구하기 위해서 사용된다. 각 객체에 isHit 메서드를 실행하여 선택되는 figure를 리턴한다. 없으면 null을 리턴한다.

public void allFigureNotSelected()
입력 : void
출력 : void
사용 목적 : 모든 객체들을 선택하지 않은 상태로 만들기 위해서 사용된다.

public void UpdateAllViews()
입력 : void
출력 : void
사용 목적 : Document를 참조하는 모든 View를 갱신시키기 위해서 사용된다.

public void pushStack()
입력 : void
출력 : void
사용 목적 : 실행 취소 기능을 구현하기 위해서 objList를 stack에 푸시한다.

public void popStack()
입력 : void
출력 : void
사용 목적 : stack에 들어있던 objList를 Document객체의 objList에 넣는다.

public void undo()
입력 : void
출력 : void
사용 목적 : 실행 취소 기능을 사용하기 위해서 사용된다. 스택에 있는 objList를 pop해서 구현된다.

public void copy()
입력 : void
출력 : void
사용 목적 : 복사 기능을 사용하기 위해서 사용된다. 선택되어 있는 객체들을 클립보드리스트에서 복사한다.

public void cut()
입력 : void
출력 : void
사용 목적 : 잘라내기 기능을 사용하기 위해서 사용된다. 선택되어 있는 객체들을 현재 Document에서 지우고 클립보드리스트에 넣는다.

public void paste()
입력 : void
출력 : void
사용 목적 : 붙여넣기 기능을 사용하기 위해서 사용된다. 클립보드리스트에 있는 객체들을 현재 Document에 붙여 넣는다.

public void SaveDocument(string fileName)
입력 : string
출력 : void
사용 목적 : 입력으로 받은 파일이름으로 이미지 형식이 아닌 바이트스트림으로 저장되는 파일로 저장한다. 도형이 잘리지 않도록 하기 위해서 View의 크기에 맞게 잘라서 저장된다.

public void SavePNG(string fileName)
입력 : string – 저장할 파일이름
출력 : void
사용 목적 : 현재 그린 객체를 PNG 형식 이미지 파일로 저장하기 위해서 사용한다. view에 있는 pictureBox를 들고와서 pictureBox를 Bitmap형식으로 저장한 다음에 객체만 저장하기 위해서 객체의 크기로 자른다. 자른 후에 PNG형식으로 저장한다.

public Point[] findPos(Bitmap bitmap)
입력 : Bitmap – 이미지를 자를 Bitmap
출력 : Point[] - 이미지를 자르고 난 후의 이미지의 위치 정보
사용 목적 : 이미지를 저장할 때 객체의 크기에 맞게 잘라서 저장하기 위해서 사용한다. 이미지에서 색을 비교해서 white가 아니면 객체의 일부로 인식하고 이를 바탕으로 이미지의 크기를 줄인 값을 point[]로 리턴한다.

public Bitmap CropBitmap(Bitmap bitmap, int cropX, int cropY, int cropWidth, int cropHeight)
입력 : Bitmap – 이미지를 자를 Bitmap
int, int – 이미지의 시작 위치
int, int – 이미지의 너비와 높이
출력 : Bitmap – 크기에 맞게 자른 이미지
사용 목적 : findPos 메서드를 통해 이미지의 자른 위치를 Point[]로 구하고 이 값을 바탕으로 새로운 Bitmap 객체를 생성하기 위해서 사용한다.

public void OpenDocument(string fileName)
입력 : string – 불러올 sof 파일이름
출력 : void
사용 목적 : sof 형식의 document 파일을 open할 때 사용한다. document 파일을 불러와서 저장되어 있는 view와 objList를 설정한다.

public void OpenPNG(string fileName)
입력 : string – 불러올 PNG 파일이름
출력 : void
사용 목적 : PNG 형식의 이미지 파일을 open할 때 사용한다. 이미지를 불러와서 Bitmap 객체를 생성하고, ImageFigure를 새로 생성하여 객체리스트에 추가한다.
예시 데이터 : 파일을 open할 때 open한 파일 이름이 png일 때 openPNG 메서드를 불러와서 ImageFigure객체를 생성한다.

public void selectFigures(Selection s)
입력 : Selection – 선택하고자 하는 영역의 크기를 가진 Selection 객체
출력 : void
사용 목적 : Selection 모드에서 객체들을 선택하려고 할 때 Selection 객체를 생성하는 데

선택한 객체들을 figList에 넣기 위해서 사용한다. 이 때 선택되는 객체들은 Figure의 모든 점이 Selection 객체 안에 있는 것으로 판단한다.

8. MyGraphics 클래스

접근 제어자 : public
사용 목적 : 도형을 클릭했을 때 클릭한 도형 주위에 크기를 조절하는 핸들을 그리기 위해 사용하는 클래스

public static void drawRectTracker(Graphics g, int x, int y)
입력 : Graphics – 핸들을 그리기 위해서 사용하는 Graphics 객체
int, int – 사각형을 그릴 x, y 좌표
출력 : void
사용 목적 : 도형을 클릭 했을 때 클릭한 도형 주위에 핸들을 그린다. 입력 받은 위치 좌표를 참조해서 그 곳에 사각형을 그린다.

9. Selection 클래스

접근 제어자 : public
상속 : Figure
사용 목적 : 선택 모드일 때 여러개의 도형을 한꺼번에 선택하는 기능을 구현하기 위해서 사용한다. 점선으로 사각형을 그리고 그 안에 있는 도형들을 선택하는 역할을 한다.

생성자
public Selection(int x1, int y1, int x2, int y2)
입력 : int, int ,int, int 사각형의 왼쪽 위 좌표와 오른쪽 아래 좌표
사용 목적 : Selection 객체를 생성한다. 부모 클래스의 생성자 중에 Figure(int, int, int ,int)를 호출한다.

public override void paint(Graphics g)
입력 : Graphics – 마우스로 드래그 할 때 사각형을 그리기 위한 Graphics 클래스
출력 : void
사용 목적 : 마우스로 드래그 할 때 새로운 사각형을 그리기 위한 메서드이다. 다른 객체들을 선택하는 역할이므로 회색의 점선으로 그려진다.

10. MyMath 클래스

접근 제어자 : public
사용 목적 : 객체를 클릭하려고 할 때 마우스가 클릭한 위치와 객체가 충돌하는 지 여부를 계산하기 위해서 사용한다.

public static Boolean isLineHit(Rect r, int px, int py)
public static Boolean isLineHit(int x1, int y1, int x2, int y2, int px, int py)
입력 : Rect – 충돌하는 지 확인하고자 하는 객체

int[4] - 충돌하는 지 확인하고자 하는 객체의 네 좌표
int, int – 마우스의 클릭 위치
출력 : Boolean – 충돌하는 지 여부
사용 목적 : View에서 마우스 클릭 이벤트가 발생할 때 마우스가 클릭한 위치와 객체의 위치를 비교해서 마우스가 클릭하는 객체가 어떤 객체인지를 확인하기 위해서 사용된다. 위치를 확인할 때는 객체의 크기를 조금 늘린 다음에 비교해서 선택이 원활하게 이루어지도록 한다.

11. View 클래스

필드

doc
접근 제어자 : private
타입 : Document
사용 목적 : 현재 View가 참조하고 있는 document 객체를 저장하기 위해 사용한다.

drag_off_x, drag_off_y
접근 제어자 : private
타입 : int
사용 목적 : 드래그 되는 동안 변한 값을 저장하기 위해 사용한다.

pictureBox
접근 제어자 : private
타입 : PictureBox
사용 목적 : View에서 객체가 그려지고 저장할 때 사용하는 pictureBox를 정의한다.

isChanged
접근 제어자 : private
타입 : bool
사용 목적 : View 에서 어떤 변화가 일어났을 때 변화가 일어난 지 여부를 저장하기 위해서 사용한다. 이동하거나 크기가 변경되었을 때 변경된다.

생성자

public View(Form parent, Document doc)
입력 : Form – View의 부모 폼이 되는 폼
Document – View가 참조하는 Document 객체
사용 목적 : View 객체를 생성하기 위해서 사용한다. View 객체가 생성될 때 부모 폼과 참조하는 Document 객체를 설정하고 변경 여부는 false로 저장한다.

메서드

public Document GetDocument()

입력 : void
출력 : Document – 현재 View 객체가 참조하는 Document 객체
사용 목적 : 현재 View 객체가 참조하고 있는 Document 객체를 받기 위해서 사용한다.

public PictureBox getPictureBox()
입력 : void
출력 : PictureBox – 현재 View 객체가 가지고 있는 pictureBox 객체를 받기 위해서 사용한다.

이벤트
protected override void Dispose(bool disposing)
입력 : disposing – 현재 객체가 dispose 되고 있는지 여부를 나타낸다.
출력 : void
사용 목적 : 객체가 dispose될 때 사용한 component들을 모두 dispose하기 위해서 사용한다.

private void View_Load(obejct sender, EventArgs e)
protected override void OnPaintBackground(PaintEventArgs e)
protected void PaintHandler(object sender, PaintEventArgs e)
protected void MouseDoubleClick(object sender, EventArgs e)
protected void MouseUphandler(object sender, MouseEventArgs e)
protected void MouseMovehandler(object sender, MouseEventArgs e)

protected void MouseDownhandler(object sender, MouseEventArgs e)
private void View_KeyDown(object sender, KeyEventArgs e)

12. Form1 클래스

접근 제어자 : public
상속 : RibbonForm
사용 목적 : 프로그램의 메인 화면의 폼으로 사용한다.

필드

Obj
접근 제어자 : private
타입 : ObjectList
사용 목적 : 시뮬레이션 모델과 객체를 보여주는 ObjectList 폼의 객체로 사용한다. 메인화면에서 오른쪽 패널에 보여주게 된다.

Ani
접근 제어자 : private
타입 : Animator
사용 목적 : 애니메이션을 진행하는 Animator 폼의 객체로 사용한다. 탭으로 이동했을 때 Animator 폼을 보여준다.

objflag
접근 제어자 : private
타입 : int
사용 목적 : 오브젝트 리스트를 보이거나 숨기기를 저장하는 플래그로 사용한다.

setting_path
접근 제어자 : private
타입 : string
사용 목적 : setting.conf 파일의 경로를 저장한다.

TapPage2
접근 제어자 : public for get, private for set
타입 : object

생성자

Form1()
접근 제어자 : public
입력 : void
사용 목적 : Form1 객체를 생성하기 위해서 사용한다. Form1 객체를 생성할 때 폼에 대한 초기화를 진행한다. Animation 관련 리본 메뉴는 숨기고 오브젝트 리스트를 초기화 한다.

Editor 탭에는 새로운 Document 객체와 View 객체를 생성하고 Animator 객체도 생성하여 멤버 필드에 할당한다.

메서드

private Document CreateDocument()

입력 : void

출력 : Document

사용 목적 : 현재 폼을 인자로 받는 Document 객체를 생성하여 리턴한다.

private Document GetActiveDocument()

입력 : void

출력 : Document

사용 목적 : 현재 활성화 되어 있는 View에서 참조하는 Document를 불러온다.

private View GetActiveView()

입력 : void

출력 : View

사용 목적 : 현재 활성화 되어있는 View 레퍼런스를 구하기 위해서 사용한다.

private void Open()

private void Save()

private void SaveAsPNG()

private void Exit()

이벤트

private void OrbMenuHandler(object sender, MouseEventArgs e)
private void smallButtonHandler(object sender, MouseEventArgs e)
public void RibbonGeneralHandler(object sender, MouseEventArgs e)
public void RibbonGeneralHandler2(object sender, MouseEventArgs e)
private void colorPicBox_Click(object sender, MouseEventArgs e)
private tabControl1_SelectedIndexChanged(object sender, EventArgs e)
private void Start_Click(object sender, EventArgs e)
private void Stop_Click(object sender, EventArgs e)

13. msg 클래스

접근 제어자 : default

사용 목적 : IPC로 받은 스트링 값들을 쉽게 사용하기 위해 쓰이는 클래스이다.

필드

msgOrig

접근 제어자 : public

타입 : string[]

사용 목적 : 주어진 문자열 배열을 좀 더 쉽게 내부 변수들에 넣기 위한 프로퍼티이다.

modelN, seqNo, X, Y, R, T

접근 제어자 : public

타입 : int

사용 목적 : 메시지의 각 항목들을 나타낸다. 차례대로 모델번호, 시퀀스 넘버, 도착 x,y좌표, radius, 지속시간이다.

14. MessageReceiver 클래스

접근 제어자 : public

사용 목적 : IPC로 메시지를 수신하기 위해 사용한다. 스레드의 일종인 BackgroundWorker가 사용된다.

필드

worker

접근 제어자 : private

타입 : BackgroundWorker

사용 목적 : 연결된 파이프에서 수신하는 메시지를 메시지 큐에 저장하는 역할을 한다.

messageQ

접근 제어자 : private

타입 : Queue<string>

사용 목적 : worker가 받은 메시지를 저장하는데 사용한다.

key

접근 제어자 : private

타입 : object

사용 목적 : messageQ에 enque/dequeue간 동기화를 위한 키로 쓰인다.

생성자

public MessageReceiver()

입력 : void

출력 :

사용 목적 : MessageReceiver의 생성자이다. 사용하는 worker와 관련된 기본 설정들을 수행한다.

메소드

public void RunWorkerAsync()

입력 : void

출력 : void

사용 목적 : worker를 시작한다.

static void getMsg(object sender, DoWorkEventArgs e)

입력 : 이벤트의 기본 인자이다. 사용하지 않는다.

출력 : void

사용 목적 : IPC 연결이 수립되어 있다면 전송하는 메시지를 받아 큐에 저장한다. 연결되어 있지 않다면 연결을 기다린다.

public string[] checkMsg(out string msg)

입력 : msg – 이후 출력을 위해 메시지 자체를 전달한다.

출력 : string[] - 항목별로 쪼개진 메시지

사용 목적 : 큐에 메시지가 있는지 확인하고, 있다면 이를 쪼개서 반환한다.

15. AnimatorHandler 클래스

접근 제어자 : public

사용 목적 : 모델들의 애니메이션을 제어해주는 클래스이다. 모델의 애니메이션을 시작하고, 그들의 좌표를 업데이트해주는 메소드를 가지고 있다.

필드

modelList

접근 제어자: private

타입 : Dictionary<int, ModelAnimator>

사용 목적 : 모델-애니메이터쌍을 저장하기 위한 해시테이블이다. AnimatorHandler 생성시 모델-애니메이터 쌍을 저장한다. 메시지를 받게 되면 여기서 해당 모델의 애니메이터를 찾아 애니메이션을 실행한다.

예시 데이터 : modelList[m.modelN].startAnim(...)

생성자

public AnimatorHandler(int[] models, Bitmap[] img, Graphics g, PictureBox b)

입력 : int[] - 모델번호들,Bitmap[] - 모델과 매핑 된 이미지들, Graphics – 그려줄 그래픽객체, Picturebox – 배경이 될 picturebox

출력 :

사용 목적 : AnimatorHandler의 생성자이다. 모델번호, 이미지를 받아 모델-애니메이터쌍을 만들어 해시테이블에 저장한다.

메소드

public void startAnim(msg m)

입력 : msg 애니메이션을 진행할 메시지 객체
출력 : void
사용 목적 : 모델의 애니메이션을 시작하는 메소드이다. 메시지를 전달 받아 해당 모델의 애니메이션을 시작한다. 두 가지 경우를 막아 놓았는데 첫째, 시퀀스 넘버가 다르면 메시지를 전송하지 않고 기다린다. 둘째, 해당 모델이 애니메이션 중이면 전송하지 않고 기다린다.

private bool canDoSeq(int seqNo)
입력 : int 메시지의 시퀀스넘버
출력 : bool 시퀀스 넘버의 메시지를 진행해도 되는지 여부
사용 목적 : 이전 시퀀스 넘버의 메시지 처리가 끝난 경우에만 다음 시퀀스 넘버로 진행할 수 있다. 이를 위해 먼저 확인하는 메소드이다. 시퀀스 넘버가 다르면서 애니메이션이 진행중이라면 false를 반환한다.

public void updatePos(Graphics g)
입력 : Graphics 그려줄 그래픽 객체
출력 : void
사용 목적 : 업데이트 된 좌표에 새로 모델들을 그려줄 때 사용하는 메소드이다. picturebox의 paint메소드에서 호출되어 사용된다.

16. ModelAnimator 클래스

접근 제어자 : public
사용 목적 : 모델들의 애니메이션을 실제 진행하는 클래스이다. 도착좌표, 시간 등을 인자로 받아 C#의 타이머를 이용해 애니메이션을 진행한다. 현재는 원운동과 선운동이 가능하다.

필드

timerInterval
접근 제어자: private
타입 : static int
사용 목적 : 타이머의 호출 빈도를 나타낸다. 자주 호출할 수 록 더 부드러운 애니메이션을 얻을 수 있으나 더 부정확해질 수 있다.

t
접근 제어자 : private
타입 : System.Timers.Timer
사용 목적 : c#의 스레드의 일종이다. 주기적으로 실행되어 설정된 이벤트를 수행한다. ModelAnimator에선 모델의 좌표를 업데이트 해주기 위해 사용하고 있다.

background
접근 제어자 : private

타입 : System.Windows.Forms.PictureBox
사용 목적 : Timer의 이벤트에서 모델의 좌표를 업데이트 해주고 난 뒤 invalidate() 메소드를 호출하여야 Paint이벤트가 호출되어 새로운 좌표에 모델들을 그려줄 수 있다. 이 invalidate() 메소드 호출을 위해 사용된다.

img
접근 제어자 : public
타입 : Bitmap
사용 목적 : 모델과 연결된 오브젝트의 이미지 파일을 저장하기 위한 프로퍼티이다.

startX, startY
접근 제어자 : public
타입 : float
사용 목적 : 모델의 x,y좌표를 나타내기 위해 쓰이는 프로퍼티이다.

isDoingAnim
접근 제어자 : public
타입 : bool
사용 목적 : modelAnimator가 현재 애니메이션을 진행 중인지 나타내는 프로퍼티이다. 메시지를 전달하기 전 이를 통해 애니메이션이 진행 중 인지 확인한다.

seqNo
접근 제어자 : public
타입 : int
사용 목적 : 현재 진행중인 애니메이션의 시퀀스 넘버를 나타내는 프로퍼티이다. 메시지 전달 전 이를 확인하여 대기 여부를 판단한다. 시작시 seqNo은 –1이다.

time
접근 제어자 : private
타입 : int
사용 목적 : 반복횟수를 의미하는 변수이다. Timer의 interval과, 건네받은 duration을 통해 반복횟수를 계산한다.

perMovex, perMovey
접근 제어자 : private
타입 : float
사용 목적 : 1회 반복당 얼마나 움직이는지 담고 있는 변수이다. Timer의 이벤트가 호출될 때 이 변수들 값만큼 startX,Y를 업데이트한다.

inx, iny
접근 제어자 : private

타입 : double
사용 목적 : 시작시 x,y각도를 나타낸다. 원운동에 사용된다.

xPi, yPi
접근 제어자 : private
타입 : double
사용 목적 : 1회 반복당 얼마나 회전하는지 담고 있는 변수이다. Timer이벤트 호출 시 이 변수들 값만큼 inx,y를 증가시킨다.

cx, cy
접근 제어자 : private
타입 : float
사용 목적 : 원의 중심을 나타내는 값이다.

radius
접근 제어자 : private
타입 : double
사용 목적 : 원의 반지름을 나타낸다.

생성자

public ModelAnimator(Bitmap img,System.Windows.Forms.PictureBox pic)
입력 : Bitmap – 매핑된 이미지, PictureBox – 배경이 될 PictureBox
출력 :
사용 목적 : ModelAnimator의 생성자이다. 타이머 생성, seqNo, startX,Y의 초기화를 진행한다.

메소드

public void startAnim(int seqNo,float ang, float x, float y, int duration)
입력 : int – 시퀀스 넘버, float – 회전 각, float,float – 목표 좌표, int - 지속시간
출력 : void
사용 목적 : 애니메이션을 시작하는 메소드이다. 위의 인자들을 받아 반복 횟수,회당 이동거리(각도)를 계산하고 원운동인 경우 중심좌표, 시작각도등도 계산한다. 타이머를 시작한다. ang가 0 이면 선운동을 의미하며 아닌 경우 원운동을 의미한다. duration이 0 이면 반복횟수는 1로 취급한다.

private void OnTimedEventLine(Object source, System.Timers.ElapsedEventArgs e)
입력 : object,System.Timers.ElapsedEventArgs – 이벤트 인자들이다.
출력 : void
사용 목적 : 선운동을 진행하는 Timer 이벤트이다. 앞서 계산한 perMove만큼 좌표를 업데이트 해주고 picturebox의 invalidate메소드를 호출한다.

private void OnTimedEventCircle(Object source, System.Timers.ElapsedEventArgs e)
입력 : object,System.Timers.ElapsedEventArgs – 이벤트 인자들이다.
출력 : void
사용 목적 : 원운동을 진행하는 Timer 이벤트이다. 앞서 계산한 x,yPi만큼 각도를 업데이트 하여 새로운 좌표를 계산한다. picturebox의 invalidate메소드를 호출한다.

17. ImageFigure 클래스

접근 제어자 : public
사용 목적 : 이미지를 불러서 조작할 때 사용하기 위한 Figure의 자식 클래스이다.

필드

image
접근 제어자 : private
타입 : Bitmap
사용 목적 : 사용자가 불러온 이미지를 저장하고 있는다. 이동, 확대, 축소와 같은 작업 시 사용하는 원본 이미지이다.

생성자

public ImageFigure(Bitmap image) : base()
입력 : Bitmap – 사용할 이미지이다.
출력 :
사용 목적 : ImageFigure의 생성자이다. 이미지의 Bitmap클래스를 받아 자신의 Rect를 생성한다.

메소드

public override void paint(Graphics g)
입력 : Graphics – 그려줄 그래픽 객체
출력 : void
사용 목적 : Figure를 그려줄 때 호출되는 메소드이다. 현재 Rect크기에 맞게 이미지를 그려주도록 되어 있다. 선택된 상태라면 외곽에 작은 점들도 그려준다.

public Bitmap getImage()
입력 : void
출력 : Bitmap – 현재 사용 중인 이미지이다.
사용 목적 : 현재 사용 중인 이미지를 반환한다.

18. MergedFigure 클래스

접근 제어자 : public
사용 목적 : 여러 Figure를 병합한 Figure를 나타내는 데 쓰인다. 우리의 프로그램은 도형의 Grouping 기능을 제공하는데 Grouping하게 되면 기존 도형들을 MergedFigure 하나로 합치게 된다.

필드

figList

접근 제어자 : private

타입 : MyArrayList

사용 목적 : 현재 Grouping된 Figure들을 저장하는 리스트이다. UnGrouping도 지원하기 때문에 리스트에 저장해둔다. MergedFigure에 행해지는 연산은 이 figList에 저장된 도형들에 행해진다.

생성자

public MergedFigure(MyArrayList figList)

입력 : MyArrayList – 묶을 Figure목록이다.

출력 :

사용 목적 : MergedFigure의 생성자이다. 묶을 도형들을 받아 figList에 저장한다. 각 도형의 Rect사이즈를 통해 MergedFigure의 Rect사이즈를 구하는 연산도 진행한다.

메소드

public override void paint(Graphics g)

입력 : Graphics – 그려줄 그래픽 객체

출력 : void

사용 목적 : 도형을 그려줄 때 사용한다. figList에 저장된 도형들의 paint 메소드를 호출하는 방식으로 진행된다. 선택된 상태라면 외곽에 작은 점들도 그려준다.

public override void move(int x, int y)

입력 : int,int – 이동할 x,y좌표이다.

출력 : void

사용 목적 : 확대, 축소, 이동시에 호출되는 메소드이다. 확대, 축소 시에 각 도형들의 상대적 위치와 크기는 유지되어야 한다. 따라서 이를 위해 먼저 각각 차지하는 비율을 계산하고 저장한다. 그 후 확대/축소가 일어나면 변화량에 계산해둔 비율을 곱하는 방식을 통해 비율을 유지하도록 한다.

public override void filled(bool b)

입력 : bool – 채우기 여부

출력 : void

사용 목적 : 도형의 색채우기 여부를 변경할 때 쓰인다. mergedfigure는 내부에 여러개의 도형을 담고 있기 때문에 모든 도형의 채우기 여부를 변경해주도록 오버라이드 하였다.

public override void setCurrentColor(Color c)

입력 : color – 채울 색

출력 : void

사용 목적 : 도형에 채울 색을 설정한다. filled와 마찬가지로 mergedfigure는 내부에 여러

개의 도형을 담고 있기 때문에 모든 도형의 채울 색을 변경해주도록 오버라이드 하였다.

19. Animator 클래스

접근 제어자 : public
사용 목적 : Animator 폼을 구현하기 위한 폼 클래스이다.

필드

rsv
접근 제어자 : private
타입 : MessageReceiver
사용 목적 : 시뮬레이터에서 생성된 데이터를 전송 받기위한 MessageReceiver 객체

animHandler
접근 제어자 : private
타입 : AnimatorHandler
사용 목적 : 애니메이션의 구현을 담당하는 AnimatorHandler 객체

g
접근 제어자 : private
타입 : Graphics
사용 목적 : 애니메이션 내용을 그리기 위해서 사용되는 Grphics 객체

myTimer
접근 제어자 : private
타입 : Timer
사용 목적 : 시뮬레이터로부터 수신된 데이터를 큐로부터 확인하고 메시지 창에 띄우기 위한 타이머 객체

생성자

Animator()
사용 목적 : Animator 폼 클래스의 생성자이다. MessageReceiver 객체를 생성하고 깜빡임 방지를 위한 더블버퍼링 설정을 한다.

이벤트

private void pictureBox1_paint(object sender, PaintEventArgs e)
private void button1_Click(object sender, EventArgs e)
private void OnTimerEvent(Object source, ElapsedEventArgs e)

20. ObjectList 클래스

접근제어자 : public

사용 목적 : 생성하는 오브젝트와 DEVS-ObjectC에서 제공하는 모델들을 매칭하기 위한 클래스이다.

변수

model_path

접근제어자 : private

타입 : string

사용목적 : 프로그램 종료 후 재시작을 할 때 모델 파일이 있는 폴더 경로 정보를 가진 파일로부터 해당경로를 불러와서 세팅상태가 유지되도록 한다.

imgList

접근제어자 : public

타입 : static

사용목적 : 오브젝트 리스트에서 불러온 오브젝트들을 Animator에서 사용하기 위해서 오브젝트의 정보를 저장한다.

메소드

public void Set_model_path(string path)

입력 : string-경로 정보를 저장

출력 : void

사용 목적 : 설정 파일에 모델 경로를 model_path 변수에 할당하기 위한 함수이다.

public void Object_Load(FileInfo[] files)

입력 : FileInfo- 파일 정보를 가지고 있는 클래스이다.

출력 : void

사용 목적 : OpenFileDialog를 사용하여 불러온 오브젝트의 정보를 저장하고 있는 FileInfo 클래스의 객체로부터 오브젝트의 이름, 경로 등을 불러와서 오브젝트 리스트에 저장하고 해당오브젝와 매칭된 모델파일을 모델 리스트에 저장한다.

private void SetupDataGrid()

입력 :

출력 : void

사용 목적 : 모델 상태변수를 보여주는 grid를 초기화 하는 함수이다.

private void listView2_Click(object sender, EventArgs e)

입력 : object , EventArgs

출력 : void

사용 목적 : 모델 리스트에서 모델을 클릭하여 해당 모델의 상태 변수와 그 값을 parsing 하여 grid에 출력 시킨다. 모델 파일을 줄 단위로 읽어서 상태 변수를 추출하고 다시 한번 줄단위로 읽어서 상태 변수의 값을 추출해낸다.